

# **Learning Out-of-Vocabulary Words in Automatic Speech Recognition**

Long Qin

CMU-LTI-13-014

Language Technologies Institute  
School of Computer Science  
Carnegie Mellon University  
5000 Forbes Ave., Pittsburgh, PA 15213  
[www.lti.cs.cmu.edu](http://www.lti.cs.cmu.edu)

## **Thesis Committee:**

Alexander I Rudnicky, CMU (Chair)  
Alan W Black, CMU  
Florian Metze, CMU  
Mark Dredze, JHU

*Submitted in partial fulfillment of the requirements  
for the degree of Doctor of Philosophy  
in Language and Information Technologies*

© 2013 Long Qin

**Keywords:** speech recognition, out-of-vocabulary (OOV) words, hybrid system, sub-lexical units, OOV word detection, system combination, recurrent OOV words, OOV word recovery

*This thesis is dedicated to my wife and my parents, who always support me during my study toward the doctoral degree.*



## Abstract

Out-of-vocabulary (OOV) words are unknown words that appear in the testing speech but not in the recognition vocabulary. They are usually important content words such as names and locations which contain information crucial to the success of many speech recognition tasks. However, most speech recognition systems are closed-vocabulary recognizers that only recognize words in a fixed finite vocabulary. When there are OOV words in the testing speech, such systems cannot identify OOV words, but misrecognize them as in-vocabulary (IV) words. Furthermore, the errors made on OOV words also affect the recognition accuracy of their surrounding IV words. Therefore, speech recognition systems in which OOV words can be detected and recovered are of great interest.

As simply applying a large vocabulary in a recognizer cannot solve the OOV word problem, several alternative approaches had been proposed. One is to use a hybrid lexicon and hybrid language model which incorporate both word and sub-lexical units during decoding. Another popular OOV word detection method is to locate where the word decoding and the phone decoding results are in disagreement. Some other methods involve with a classification process to find possible OOV words using confidence scores and other evidence. For OOV word recovery, the phoneme-to-grapheme (P2G) conversion is usually applied to predict the written form of an OOV word.

Current OOV research focuses on detecting the presence of OOV words in the testing speech. There is only limited work about how to convert OOV words into IV words of a recognizer. In this thesis, we therefore investigated learning OOV words in speech recognition. We showed that it is feasible for a recognizer to automatically learn new words and operate on an open vocabulary. Specifically, we built an OOV word learning framework which consists of three major components. The first component is OOV word detection, where we built hybrid systems using different sub-lexical units to detect OOV words during decoding. We also studied to improve the hybrid system performance using system combination and OOV word classification techniques. Since OOV words can appear more than once in a conversation or over a period of time, in the OOV word clustering component, we worked on finding multiple instances of the same OOV word. At last, in OOV word recovery, we explored how to integrate identified OOV words into the recognizer's lexicon and language model. The proposed work was tested on tasks with different speaking styles and recording conditions including the Wall Street Journal (WSJ), Broadcast News (BN), and Switchboard (SWB) datasets. Our experimental results show that we are able to detect and recover up to 40% OOV words using the proposed OOV word learning framework. Finally, a self-learning speech recognition system will be more robust and has broader applications in real life.



## **Acknowledgments**

First and foremost, I would like to thank my advisor, Prof. Alexander I. Rudnicky, for his support and advice throughout my journey towards the doctoral degree at Carnegie Mellon. In these years, Alex has taught me how to understand a problem and how to solve it step by step. He always supported me to explore unconventional ideas and gave me the time and resource to work on various topics.

I also thank my thesis committee members for their precious advice and encouragement on my thesis work. Prof. Alan W Black is very knowledgeable about every aspect of spoken language processing, who broadened my research horizon. Prof. Florian Metze has worked with me in his speech recognition class, in the BABEL project, as well as in my thesis. He constantly advised me to conduct thorough analysis on experiments to better understand research results. Prof. Mark Dredze is an expert of both natural language and speech processing, who offered valuable suggestions on research directions.

Carnegie Mellon is a great place for studying. It has the best atmosphere and the smartest people. I feel fortunate to know so many close friends here. With my gratitude to all my friends, I specially thank Dr. David Huggins Daines and Matthew Marge. David was a senior students in our lab who taught me many fundamental speech recognition concepts and tools. Matt is a very kind colleague. He introduced me American cultures and helped to improve my English speaking and writing skills. I also want to thank the City of Pittsburgh for not causing too much trouble in the past six long winters.

Above all, I am deeply grateful for my family for their endless love, understanding, support and encouragement. My parents did their best to give me a good education. They encouraged me to play with complex problems since I was a kid. At last, I want to thank my wife for her love, support, tolerance and sacrifice through this whole process. You made many hard days easier and better. This dissertation is as much yours as it is mine.





# Contents

<b>1</b>	<b>Introduction</b>	<b>1</b>
1.1	The OOV word problem . . . . .	1
1.2	Thesis statement . . . . .	2
1.3	Summary of thesis contributions . . . . .	2
1.4	Thesis organization . . . . .	3
<b>2</b>	<b>Automatic Speech Recognition</b>	<b>5</b>
2.1	The general ASR framework . . . . .	5
2.2	Feature extraction . . . . .	6
2.3	Recognition . . . . .	6
2.3.1	Acoustic model . . . . .	6
2.3.2	Language model . . . . .	7
2.3.3	Dictionary . . . . .	8
2.4	Word lattice and confidence score . . . . .	10
2.4.1	Word lattice . . . . .	10
2.4.2	Confidence score . . . . .	10
2.5	Summary . . . . .	11
<b>3</b>	<b>Related Work</b>	<b>13</b>
3.1	OOV word detection . . . . .	13
3.1.1	Finding the mismatch between phone and word recognition results . . . . .	13
3.1.2	Considering OOV word detection as a binary classification problem . . . . .	14
3.1.3	Using hybrid models to explicitly model OOV words . . . . .	15
3.1.4	Combining hybrid model and classification . . . . .	17
3.2	OOV word recovery . . . . .	17
3.3	Summary . . . . .	17
<b>4</b>	<b>OOV Word Detection</b>	<b>19</b>
4.1	OOV word detection using hybrid systems and OOV word classifier . . . . .	19
4.1.1	Hybrid systems . . . . .	19
4.1.2	Sub-lexical units . . . . .	21
4.1.3	The OOV word classifier . . . . .	22
4.2	System combination for OOV word detection . . . . .	22
4.2.1	Combining multiple hybrid systems' outputs . . . . .	22

4.2.2	Combining multiple types of sub-lexical units . . . . .	25
4.3	Experiment setup and datasets . . . . .	27
4.3.1	Experiment setup . . . . .	27
4.3.2	Datasets . . . . .	28
4.4	Experimental results . . . . .	30
4.4.1	Baseline word recognition results . . . . .	30
4.4.2	Baseline OOV word detection results . . . . .	31
4.4.3	OOV word classifier results . . . . .	39
4.4.4	System combination results . . . . .	40
4.5	Summary . . . . .	45
<b>5</b>	<b>Finding Recurrent OOV Words</b>	<b>47</b>
5.1	Bottom-up clustering . . . . .	47
5.2	Measuring the distance between OOV candidates . . . . .	48
5.2.1	Phonetic distance . . . . .	48
5.2.2	Acoustic distance . . . . .	49
5.2.3	Contextual distance . . . . .	50
5.3	Experiment setup . . . . .	51
5.3.1	The hybrid system and dataset . . . . .	51
5.3.2	Evaluation metrics . . . . .	52
5.4	Experimental results . . . . .	53
5.4.1	The intra-cluster and inter-cluster distances . . . . .	53
5.4.2	The bottom-up clustering results . . . . .	54
5.5	Summary . . . . .	56
<b>6</b>	<b>OOV Word Recovery</b>	<b>57</b>
6.1	The baseline OOV word recovery approach . . . . .	57
6.1.1	Estimating the written form of an OOV word . . . . .	57
6.1.2	Estimating the language model scores of an OOV word . . . . .	59
6.2	OOV word recovery through filtering . . . . .	61
6.3	Recovering recurrent OOV words . . . . .	62
6.4	Experiment setup . . . . .	63
6.5	Experimental results . . . . .	65
6.5.1	The baseline OOV word recovery results . . . . .	65
6.5.2	The results of OOV word recovery through filtering . . . . .	77
6.5.3	The results of recovering recurrent OOV words . . . . .	82
6.6	Summary . . . . .	84
<b>7</b>	<b>Conclusion and Future Work</b>	<b>85</b>
7.1	Summary of results and contributions . . . . .	85
7.1.1	Comparing the results on different datasets . . . . .	88
7.2	Future work . . . . .	90
	<b>Bibliography</b>	<b>93</b>

# List of Figures

1.1	The overall framework of our OOV word learning system. . . . .	2
2.1	The general ASR framework. . . . .	5
2.2	A 3-state left-to-right HMM. . . . .	7
2.3	An example of the SPHINX word lattice. . . . .	10
3.1	An example of detecting OOV word by locating the mismatch between phone and word recognition results. . . . .	14
3.2	An example of detecting OOV word through classification. . . . .	15
3.3	The hybrid ASR framework. . . . .	16
4.1	The training process of the hierarchical hybrid LM. . . . .	20
4.2	The training process of the flat hybrid LM. . . . .	20
4.3	The ROVER framework. . . . .	23
4.4	The ROVER combination process. . . . .	24
4.5	An example of modeling OOV words in the mix-hier system. . . . .	25
4.6	An example of modeling OOV words in the mix-flat system. . . . .	26
4.7	The OOV word detection performance of hierarchical hybrid systems with different number of subword units and different grapheme lengths. . . . .	32
4.8	The OOV word detection performance of flat hybrid systems with different number of subword units and different grapheme lengths. . . . .	32
4.9	The OOV word detection performance of hierarchical hybrid systems with different sub-lexical units. . . . .	33
4.10	The OOV word detection performance of flat hybrid systems with different sub-lexical units. . . . .	34
4.11	The OOV word detection performance of hierarchical and flat hybrid systems. . .	35
4.12	The OOV word detection results using bigram and trigram flat hybrid models. . .	37
4.13	The OOV word detection results using hybrid systems trained from small and large dictionaries. . . . .	38
4.14	The OOV word detection results using the hybrid system and OOV word classifier. .	40
4.15	The OOV word detection results of individual and combined systems. . . . .	42
4.16	The OOV word detection results of combining multiple types of sub-lexical units in one system. . . . .	43
4.17	The OOV word detection results of combining multiple systems' outputs and combining multiple types of sub-lexical units. . . . .	44

5.1	Comparison of the average distance between candidates from the same OOV word (intra-cluster) with the average distance between candidates from different OOV words (inter-cluster). . . . .	53
5.2	The performance of bottom-up clustering using one feature. . . . .	54
5.3	The performance of bottom-up clustering using the combined feature. . . . .	55
5.4	The ARI and AMI scores calculated from all clusters and only from clusters with more than one candidate. . . . .	56
6.1	An example of the P2G conversion from an incorrect OOV word pronunciation. . . . .	59
6.2	An example of the POS labels of decoded OOV and IV words. . . . .	60
6.3	The OOV rate, WER and run time of the WSJ Eval set when increasing the vocabulary size of the recognizer. . . . .	62
6.4	The OOV word recovery results using trigram flat hybrid models with different sub-lexical units. . . . .	66
6.5	The OOV word recovery results using hybrid systems and P2G models built with small and large dictionaries. . . . .	68
6.6	The pronunciation accuracy and recovery rate of hybrid systems with the baseline and better P2G conversion. . . . .	70
6.7	The OOV word recovery results with the baseline and better P2G conversion. . . . .	71
6.8	The vocabulary size and OOV rate on the Eval and Test data of the baseline and expanded vocabulary. . . . .	73
6.9	The number of recognized OOV words when integrating OOV $n$ -grams into the language model. . . . .	74
6.10	The recognition performance when integrating OOV $n$ -grams into the language model. . . . .	75
6.11	The number of recognized OOV words when integrating unseen OOV $n$ -grams into the language model. . . . .	76
6.12	The recognition performance when integrating unseen OOV $n$ -grams into the language model. . . . .	77
6.13	The OOV rate of different size recognition vocabularies produced by choosing upon word frequency and filtering a large lexicon. . . . .	79
6.14	The OOV rate of recognition systems built from different vocabularies. . . . .	79
6.15	The WER of recognition systems built from different vocabularies. . . . .	80
6.16	The OOV rate of the recovered and filtered expanded vocabularies in the Eval and Test data. . . . .	81
6.17	The recognition performance when using the recovered and filtered expanded vocabularies on the Eval and Test data. . . . .	81
6.18	The OOV words recognized ratio when using the recovered and filtered expanded vocabularies on the Eval and Test data. . . . .	82
6.19	The recognition performance of ASR systems built from baseline recovered expanded vocabulary and recurrent recovered expanded vocabulary. . . . .	84

# List of Tables

1.1	The recognition result of an utterance with one OOV word - “ <b>AIRCOA</b> ”. . . . .	1
2.1	An example of the dictionary used in our ASR system. . . . .	9
4.1	The OOV rate of the Dev and Eval data. . . . .	29
4.2	The percentage of Dev and Eval OOV words in the training speech and text data. . . . .	30
4.3	The baseline word recognition results. . . . .	30
4.4	The number of words used for training sub-lexical units. . . . .	31
4.5	The optimal number of subword units and grapheme length. . . . .	31
4.6	The number of sub-lexical units in different hierarchical hybrid systems. . . . .	33
4.7	The number of sub-lexical units in different flat hybrid systems. . . . .	33
4.8	The weighted length of sub-lexical units in different hierarchical hybrid systems. . . . .	34
4.9	The weighted length of sub-lexical units in different flat hybrid systems. . . . .	35
4.10	The optimal number of subword units and grapheme length used in flat hybrid models. . . . .	36
4.11	The optimal number of subword units and grapheme length used in hybrid systems trained from small and large dictionaries. . . . .	38
4.12	The distribution of common detection errors among the syllable, subword, and grapheme hybrid systems. . . . .	41
4.13	The distribution of different detection errors. . . . .	41
5.1	Examples of the phonetic, acoustic and contextual features of an OOV candidate. . . . .	47
5.2	Examples of the local and global contextual features of OOV candidates. . . . .	51
5.3	The OOV word detection performance on the Eval data of selected hybrid systems. . . . .	52
5.4	The number of instances an OOV word has in the hybrid system’s output. . . . .	52
5.5	Details of the OOV word clustering result. . . . .	56
6.1	Examples of the OOV word clustering results. . . . .	63
6.2	The OOV rate and WER of the new Test data. . . . .	64
6.3	The pronunciation accuracy of different hybrid systems. . . . .	65
6.4	The recovery rate of different hybrid systems. . . . .	65
6.5	The pronunciation accuracy of hybrid systems built with the large dictionary. . . . .	67
6.6	The recovery rate of hybrid systems built with the large dictionary. . . . .	67
6.7	The statistics on OOV words in the training speech. . . . .	69
6.8	The OOV word detection performance on the training speech. . . . .	69

6.9	The number of alignments and unique OOV words in the hybrid decoding output of the training speech. . . . .	69
6.10	The POS label accuracy on OOV words in the Eval data. . . . .	72
6.11	The distribution of POS labels of OOV words in the Eval data. . . . .	72
6.12	The number of recovered OOV words and their POS label accuracy. . . . .	73
6.13	The number of hypothesized OOV words and OOV words eliminated in the Eval and Test data. . . . .	73
6.14	The number of $n$ -gram scores added into the language model. . . . .	74
6.15	The number of new OOV $n$ -grams added into the language model. . . . .	76
6.16	The number of oracle OOV $n$ -grams added into the language model. . . . .	76
6.17	The vocabulary size and OOV rate on the Eval data of different systems. . . . .	78
6.18	The statistics of the hybrid decoding output in each task. . . . .	78
6.19	The size of the recovered and filtered expanded vocabularies. . . . .	80
6.20	The pronunciation accuracy of recurrent OOV words before and after combination. . . . .	83
6.21	The recovery rate of recovered recurrent OOV words with and without combination. . . . .	83
6.22	The POS label accuracy of recurrent OOV words before and after combination. . . . .	83

# Chapter 1

## Introduction

In this Chapter, we first discuss the out-of-vocabulary (OOV) word problem in automatic speech recognition. Following that, the thesis statement, a summary of thesis contributions, and thesis organization are presented.

### 1.1 The OOV word problem

OOV words are unknown words that appear in the testing speech but not in the recognition vocabulary. Most automatic speech recognition (ASR) systems can only recognize words that belong to a fixed finite vocabulary. When encountering an OOV word, the recognizer will incorrectly recognize the OOV word with one or more similar sounding in-vocabulary (IV) words. In addition, OOV words also affect the recognition performance of their surrounding IV words. Table 1.1 shows the recognition result of an utterance with one OOV word - “**AIRCOA**”. We can find that instead of just one substitution error, it contains four recognition errors. Because of the presence of OOV word, the preceding two words “KNOWN AS” of the OOV word “**AIRCOA**” are also not recognized. On average, one OOV word introduces 1.2 word errors [Rosenfeld, 1995]. Furthermore, OOV words are usually important content words, such as names, locations, etc., which incorporate crucial information for understanding the recognition result. For instance, in a voice search system, many OOV words are business names, without successfully identifying OOV words, such system cannot find the information that a user looks for.

Table 1.1: The recognition result of an utterance with one OOV word - “**AIRCOA**”.

associated	inns	KNOWN	AS	***	<b>AIRCOA</b>
associated	inns	AND	IS	A	TELE

Is it possible to predefine a lexicon containing all words that may be encountered by a recognizer? The answer is **NO**. Language is constantly growing and changing. New words always appear in a language. Furthermore, one language may borrow a large number of words from other languages. A major source of OOV words is words from foreign languages. Take English as an example, we “imported” résumé from French, kung fu from Chinese, kawaii from Japanese, etc. Those words were initially OOV words before they were adopted into English.

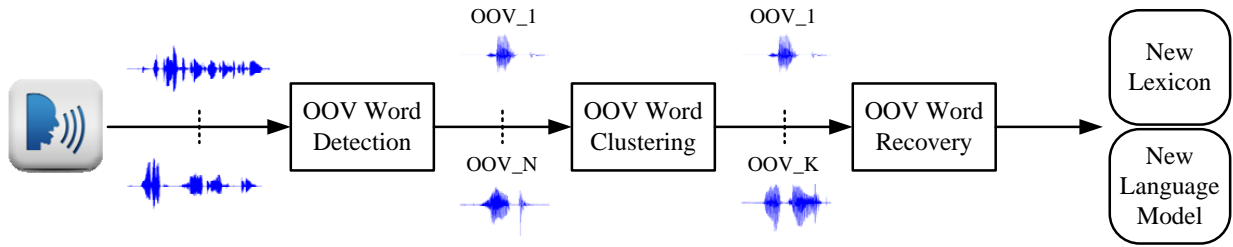


Figure 1.1: The overall framework of our OOV word learning system.

And not mention that there is an infinite number of unknown foreign names that may come up in the testing speech.

## 1.2 Thesis statement

It is necessary for an ASR system to be able to detect the presence of OOV words in the testing speech. But more important, an ASR system should also be capable of learning new words - converting OOV words into IV words, so that it can recognize those new words correctly in the future. In this thesis, we investigated learning OOV words in automatic speech recognition. We showed that it is feasible for a recognizer to automatically learn new words and operate on an open vocabulary. Precisely, to learn new words or to convert OOV words into IV words, we have to first detect the OOV word and then estimate the written form and  $n$ -gram language model scores of that word. Therefore, we build an OOV word learning system which can detect, cluster and recover OOV words. We adopted the hybrid system for OOV word detection. Then, we performed OOV word clustering to identify recurrent OOV words. And at last, we integrated OOV words into the recognizer's lexicon and language model in OOV word recovery. With the ability of learning new words, the speech recognition system will be more robust and has broader applications in real life.

## 1.3 Summary of thesis contributions

This thesis investigates a fundamental problem in speech recognition – how does an ASR system automatically learn new words? Overall, the most important contribution is that we proposed an OOV word learning framework and we showed that an ASR system is capable of learning new words and operating on an open vocabulary. Within this framework, several contributions had been made:

- We compared different training schemes and different types of sub-lexical units for building the hybrid system for OOV word detection.
- We demonstrated that system combination and OOV word classification techniques can improve the OOV word detection performance.
- We successfully identified recurrent OOV words through a bottom-up clustering process. We also showed that multiple instances of the same OOV word are valuable for improving



the OOV word recovery performance.

- We improved the phoneme-to-grapheme (P2G) conversion performance by training a better P2G model from the decoding result of training speech. We also proposed to estimate  $n$ -gram language model scores for OOV words from syntactic and semantic similar IV words.
- We studied OOV word recovery through filtering when a large dictionary and extra text data is available.

## 1.4 Thesis organization

The remainder of this thesis is organized as follows:

- Chapter 2 introduces the general framework of an automatic speech recognition system. It also explains some basic concepts in speech recognition, which will help us understand the OOV word problem and our thesis work better.
- Chapter 3 describes the details of current OOV word detection and recovery research. It also discusses the advantages and limitations of each technique.
- Chapter 4 presents our work on OOV word detection. Specifically, We explored two different approaches for training hybrid systems, the hierarchical and flat hybrid systems. We also compared building hybrid systems using different types of sub-lexical units, such as the phone, syllable, subword and grapheme units. Then, the OOV word classifier was applied to reduce false detections. Furthermore, we demonstrated two system combination techniques to improve the OOV word detection performance.
- Chapter 5 discusses how to find recurrent OOV words through a bottom-up clustering process. The phonetic, acoustic and contextual features were collected and used to measure the similarity between OOV candidates.
- Chapter 6 writes about our OOV word recovery work. The details of how we estimated the written form and  $n$ -gram language model scores for an OOV word are provided. In the second part of this chapter, it describes an alternative method for OOV word recovery when a large dictionary and extra text data are available. It also demonstrates that the clustered recurrent OOV words can be utilized to improve the OOV word recovery performance.
- Chapter 7 summaries thesis results and contributions. It also discuss future research on learning OOV words in speech recognition.



# Chapter 2

## Automatic Speech Recognition

In this chapter, we first present the general framework of an ASR system. Then, we briefly discuss a few major components within this framework. And finally, some important concepts in ASR are discussed.

### 2.1 The general ASR framework

Automatic speech recognition, hereafter referred as ASR, converts spoken words into text. In the past decade, many algorithms had been studied and developed to improve the performance of ASR systems. Popular applications of ASR, such as voice search, voice control and spoken dialog system, etc., had also been widely used.

An ASR system generally includes two major components: the front-end and the decoder. As shown in Figure 2.1, the front-end extracts feature observations  $O$  from the input speech signal  $S$ , so as to obtain an appropriate representation of speech. While the decoder utilizes the predefined acoustic model, language model and dictionary to recover words  $W$  from the feature observations  $O$ .

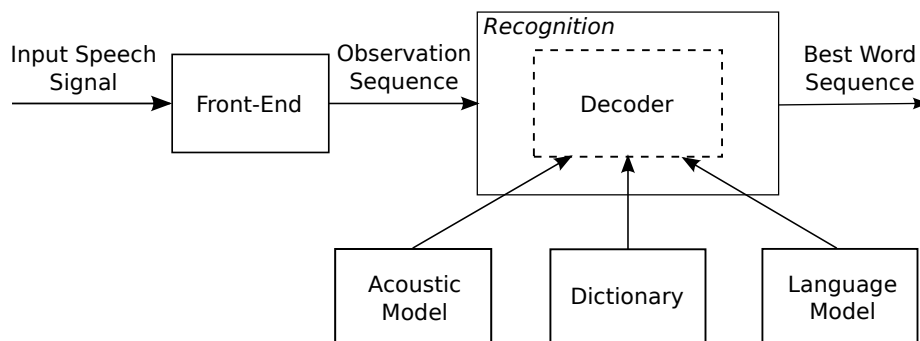


Figure 2.1: The general ASR framework.

## 2.2 Feature extraction

The input speech signal  $S$  is usually time-domain sampled speech waveform. However, human hearing is based on the characteristics of speech sounds in the frequency-domain, thus a spectral representation of speech signal is more useful for speech recognition. Since speech signal is a time varying signal, which is stationary within a short period of time but changes over a longer time [Rabiner & Juang, 1993]. When extracting features, we need to segment the input speech signal into small frames, then process each frame separately. The frame length is normally 25 msec. It is short enough to capture the rapid transitions in speech and good enough to achieve sufficient time-domain resolution. As the mel-scale approximates the human auditory response better, the mel-frequency cepstral coefficients (MFCCs) is one of the most popular feature representations in speech recognition [Davis & Mermelstein, 1980].

## 2.3 Recognition

Following feature extraction, the recognition component decodes the most probable word sequence  $W$  from the observation sequence  $O$ . This recognition process can be represented by the following equation:

$$\begin{aligned}\hat{W} &= \arg \max_W P(W|O) \\ &= \arg \max_W \frac{P(W)P(O|W)}{P(O)},\end{aligned}\tag{2.1}$$

where  $P(W)$  is the prior probability of the word sequence  $W$ ,  $P(O|W)$  is the likelihood of the observation sequence  $O$  given the word sequence  $W$ , and  $P(O)$  is the probability of observing  $O$ . Since  $P(O)$  is not a variable of  $W$ , Equation 2.1 can be written as

$$\hat{W} = \arg \max_W P(W)P(O|W).\tag{2.2}$$

Although we don't know the true distribution of  $P(O|W)$  and  $P(W)$ , those probabilities can be estimated from the predefined acoustic model and language model.

### 2.3.1 Acoustic model

Most ASR systems adopt the hidden Markov models (HMMs) [Baum & Petrie, 1966; Baum & Egon, 1967] to capture the acoustic characteristics of speech sounds. Figure 2.2 shows the typical topology of HMMs used in speech recognition. The model has three hidden states concatenated from left to right. After entering into a state, a sample can either remain in that state for a while or transit to the next state. The observation sequence  $O$  is generated by each state and only depends on that state.

To train a HMM, we need to estimate the initial state distribution  $\pi = \{\pi_i = P(q_1 = S_i)\}$ , the transition probability matrix  $A = \{a_{ij} = P(q_{t+1} = S_j | q_t = S_i)\}$  and the observation distribution  $B = \{b_i(O_t) = P(O_t | q_t = S_i)\}$ , where  $O = \{O_1, O_2, \dots, O_T\}$  is a  $T$ -frame

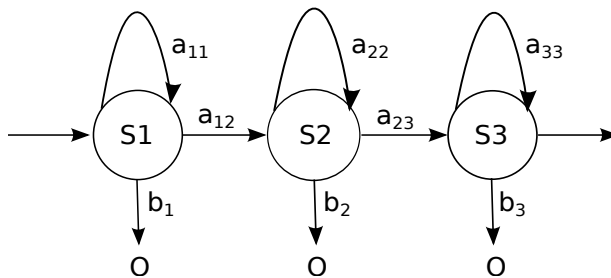


Figure 2.2: A 3-state left-to-right HMM.

observation sequence and  $Q = \{q_1, q_2, \dots, q_T\}$  is the underlying state sequence. The Gaussian mixture model (GMM) is usually used to approximate the observation distribution  $B$ , hence the likelihood  $P(O|W)$  of the observation sequence  $O$  given  $W$  can be calculated as

$$\begin{aligned}
 P(O|W) &= \sum_{all Q} P(O|Q, \lambda) P(Q|\lambda) \\
 &= \sum_{q_1, q_2, \dots, q_T} \pi_{q_1} b_{q_1}(O_1) a_{q_1 q_2} b_{q_2}(O_2) \cdots a_{q_{T-1} q_T} b_{q_T}(O_T). \quad (2.3)
 \end{aligned}$$

These HMM parameters  $\lambda = (\pi, A, B)$  can be estimated using the well known Baum-Welch (BW) algorithm [Baum et al., 1970], a special case of the classical Expectation-Maximization (EM) algorithm [Dempster et al., 1977].

HMMs can be trained on different units, such as phones, syllables, words, etc. As there are fewer unique phones than words in a language, training phone HMMs requires much less training data than training word HMMs. On the other hand, because co-articulations often appear in continuous speech, the speech signal of a phone can be heavily influenced by surrounding phones. Simply training a HMM for each phone is not sufficient to model the acoustic properties of speech sounds in different contexts. As a result, in speech recognition, HMMs are normally trained on triphone, which is a phone unit preceded and followed by specific phones. However, even just training triphone HMMs, there are still too many triphone units to work with. For example, in our English ASR system, there are only 39 phones, but up to  $39^3 = 59319$  unique triphones. Therefore, to reduce the amount of training data, we cluster triphone HMM states or Gaussian mixtures into groups and use the data from each group for training [Hwang, 1993; Huang, 1989].

### 2.3.2 Language model

Language model is used to calculate the prior probability  $P(W)$  of observing the word sequence  $W$  in a language. In speech recognition, language model is very helpful to discriminate acoustic ambiguous speech sounds and reduce the search space during decoding. For example, it is very difficult to discriminate the following two utterances, “I OWE YOU TOO” and “EYE O U TWO”, using acoustic properties. But from our prior knowledge of English, we know that the first utterance is more likely to hear than the second utterance in real life. Mathematically,  $P(W)$

can be decomposed as

$$\begin{aligned}
P(W) &= P(w_1, w_2, \dots, w_n) \\
&= P(w_1)P(w_2|w_1) \cdots P(w_n|w_1, w_2, \dots, w_{n-1}) \\
&= \prod_{i=1}^n P(w_i|w_1, w_2, \dots, w_{i-1}),
\end{aligned} \tag{2.4}$$

which is a product of the probabilities of observing word  $w_i$  given its history.

In reality, the probability  $P(w_i|w_1, w_2, \dots, w_{i-1})$  is impossible to calculate for even a moderate length of history, since most word sequences  $w_1, w_2, \dots, w_{i-1}, w_i$  only occur a few times. Instead, we estimate  $P(w_i|w_1, w_2, \dots, w_{i-1})$  based on several previous words  $w_{i-N+1}, \dots, w_{i-1}$ . This produces the widely used  $n$ -gram language model. Based on the length of word history, there are different complexity  $n$ -gram models, such as the unigram  $P(w_i)$ , bigram  $P(w_i|w_{i-1})$ , trigram  $P(w_i|w_{i-2}, w_{i-1})$ , etc. Take the trigram as an example, the trigram probability  $P(w_i|w_{i-2}, w_{i-1})$  can be estimated using the count of the word pair  $C(w_{i-2}, w_{i-1})$  and the triplet  $C(w_{i-2}, w_{i-1}, w_i)$ ,

$$P(w_i|w_{i-2}, w_{i-1}) = \frac{C(w_{i-2}, w_{i-1}, w_i)}{C(w_{i-2}, w_{i-1})}. \tag{2.5}$$

Similarly, we can compute other  $n$ -gram probabilities in the same way. However, if the word sequence is not seen in the training data,  $P(W)$  will be assigned zero probability, although it may appear in some other places. Various smoothing techniques had been proposed to mitigate this problem, e.g., the Good-Turing smoothing, the Katz smoothing and the modified Kneser-Ney smoothing [Good, 1953; Katz, 1978; Kneser & Ney, 1995; Chen & Goodman, 1998].

Besides using the recognition word error rate (WER) to evaluate the quality of a language model, a more direct way is to measure the probability of the testing word sequences through the language model, which is the perplexity of a model on a data set. The perplexity is simply the cross-entropy between the model and the data set,

$$PP(W) = 2^{H(W)}, \tag{2.6}$$

where

$$H(W) = -\frac{1}{N_W} \log_2 P(W), \tag{2.7}$$

and  $N_W$  is the length of the testing word sequence. The perplexity can be roughly interpreted as the average branching factor of the testing data to the language model. It is generally true that lower perplexity correlates to better recognition performance. As the lower the perplexity, the less branches the speech recognizer needs to consider during decoding [Huang et al., 2001].

### 2.3.3 Dictionary

We had discussed the acoustic model and language model in previous paragraphs. As shown in Figure 2.1, there is another module in the decoder, the dictionary. Acoustic model measures the acoustic properties of speech sounds. Language model estimates the prior probability of word sequences in a language. While dictionary bridges the gap between acoustic model and language

model with the lexical knowledge. Dictionary provides pronunciations of words, so decoder knows which HMMs to use for a certain word. Dictionary also provides a list of words to limit the language model complexity and the decoder’s search space. As a result, an ASR system can only recognize a limited number of words presented in the dictionary, which is normally known as the closed-vocabulary speech recognition. Table 2.1 shows part of the dictionary used in our ASR system. We can find that for some words, such as the word “A”, multiple pronunciations are given in the dictionary, as there are normally a few different ways to pronounce those words.

Table 2.1: An example of the dictionary used in our ASR system.

A	AH
A(2)	EY
ABANDON	AH B AE N D AH N
⋮	⋮
INK	IH NG K
⋮	⋮
ZURICH	Z UH R IH K

It is not easy to generate a dictionary from scratch. To obtain a dictionary specifically for speech recognition, it usually involves with multiple linguists manually write rules and check individual pronunciations. This process can be very costly and time consuming. Not mention that many linguists may not agree with each other and a linguist may not be consistent over a long period of time. Researchers had investigated to predict pronunciations of new words with models trained from existing dictionaries [Chen, 2003; Bisani & Ney, 2008]. There are also some work on refining an existing dictionary with spoken examples [Bahl et al., 1991; Maison, 2003]. However, most dictionaries used in ASR systems still require human intervention.

The size of a dictionary, i.e., the number of unique words it contains, is an important parameter for an ASR system. For some domain-specific applications, a 5k-word dictionary may be enough. For a large vocabulary continuous speech recognition (LVCSR) system, a 64k-word or larger dictionaries are usually applied. While for voice search systems, it is very common to apply a dictionary with more than 100k words. A very large dictionary may cause several issues to an ASR system. First, it requires more data for training the acoustic model and language model, which will produce larger models with more parameters. As a result, the decoder will consume more memory to load those models during decoding. Second, a larger dictionary tends to increase the perplexity of the language model to the testing data, which will affect the speed and accuracy of the recognizer, since it increases the size of the search space during decoding. Therefore, we cannot always use a very large dictionary for all speech recognition applications.

## 2.4 Word lattice and confidence score

### 2.4.1 Word lattice

The word lattice is a by-product of the recognition process, which is usually considered as a compact representation of the most possible recognition hypotheses. It contains a set of word hypotheses with boundary times and transitions between different hypotheses [Ortmanns & Ney, 1997]. Figure 2.3 provides an example of a word lattice generated by the SPHINX-3 decoder [Lee et al., 1990]. In this word lattice, each node is associated with a word and its entry time, while an arc is the transition from one node to another. This lattice corresponds to an utterance with only one word “YES”. It can be found that such word lattice usually contains a large number of word hypotheses including both the true hypotheses and the competing hypotheses. Because of the rich information embedded in the word lattice, nowadays, it had been involved in various stages of the speech recognition process, such as the discriminative acoustic modeling and the multi-pass decoding.

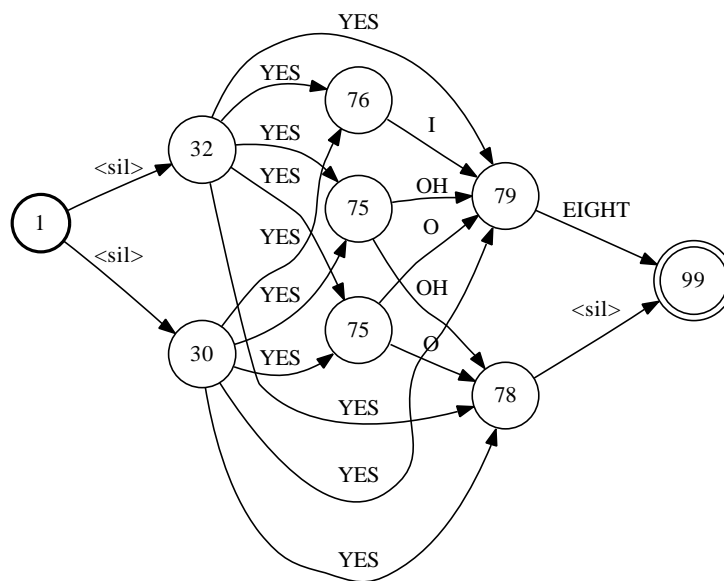


Figure 2.3: An example of the SPHINX word lattice.

### 2.4.2 Confidence score

The confidence score is an estimate of how reliable the recognition output is, which is usually used to automatically spot possible recognition errors. Some researchers calculated the confidence score from a large number of heuristic features, such as the number of times a back-off in the language model occurs or the log of the number of phones in a word. A more popular confidence score is the word posterior probability derived from the word lattice through the lattice-based forward-backward algorithm [Wessel et al., 2001]. In this case, the confidence score is the posterior probability of a word appearing at a specific time given the whole utterance.



## 2.5 Summary

In this chapter, we discussed the general framework and some key components of an ASR system including the acoustic model, language model, word lattice, etc. To learn more details about speech recognition techniques, please refer to textbooks such as [Rabiner & Juang, 1993] and [Huang et al., 2001].



# Chapter 3

## Related Work

In this chapter, we discuss the details of current solutions for the OOV word problem in speech recognition. For OOV word detection, we describe three different approaches as well as some work on combining those techniques to improve the detection performance. For OOV word recovery, we present the commonly used phoneme-to-grapheme (P2G) conversion method and some recent studies. At last, we discuss the limitations of existing techniques.

### 3.1 OOV word detection

#### 3.1.1 Finding the mismatch between phone and word recognition results

In this method, both the weakly constrained phone recognition and the strongly constrained word recognition are performed on the testing speech. Then the two recognition results are aligned and the region where the bad alignment occurs is considered as an OOV word region. The assumption here is that the phone recognition and the word recognition results should be equivalent if there is no OOV word. Otherwise, if there is an OOV word, the word recognizer will substitute the OOV word with another IV word. As a result, the word recognition result will not be able to align with the phone recognition result. Therefore, we can detect the OOV word by finding the mismatch between the phone and word recognition results. Figure 3.1 provides an example of detecting OOV words by finding the mismatch between the phone and word recognition results. We can see that the OOV word “AIRCOA” was successfully located. However, because of phone recognition errors, the system also incorrectly recognized two IV words as OOV words.

This approach was first proposed in [Hayamizu et al., 1993], where the 1-best word recognition hypothesis was converted into a sequence of phones and then aligned with the 1-best phone recognition hypothesis to find the possible OOV word region. [White et al., 2008] extended the work to also measure the alignment between the word recognition hypothesis and the output of transducing the phone recognition hypothesis into words. In [Lin et al., 2007], instead of using the 1-best recognition hypotheses, word lattices produced by the phone and word recognizers were aligned. Since the word lattice provides a concise representation of all possible recognition hypotheses, the alignment between such lattices attempted to find the best alignment between all possible phone recognition and word recognition hypotheses. [Burget et al., 2008; Kombrink

<b>WORD HYP</b>	associated	inns	and	is	a	tele
<b>WORD HYP</b>	AH S OW S IY EY T AH D	IH N Z	AH N D	IH Z	AH T EH L AH	
<b>PHONE HYP</b>	AH S OW IY EY D	IH N Z	N OW D	AE Z	EH R OW AH	
<b>HYP</b>	OOV	IV	OOV	IV	OOV	
<b>REF</b>	associated	inns	known	as	AIRCOA	

Figure 3.1: An example of detecting OOV word by locating the mismatch between phone and word recognition results.

et al., 2009] studied to align not the recognition hypotheses but the frame-based posterior scores from the phone and word recognizers.

This method is simple to implement. However, phone recognition is not very reliable, the accuracy sometimes could be lower than 50%. As there are many errors in the phone recognition result which also cause the mismatch between the phone and word recognition hypotheses, a large portion of the reported OOV words are in fact recognition errors. As a result, the OOV word detection performance is normally poor.

### 3.1.2 Considering OOV word detection as a binary classification problem

The OOV word detection task can be considered as a binary classification problem, where each word in the recognition result is classified as an OOV or IV word according to the collected evidence. Various features and different classifiers had been studied in this framework. For example, in [Sun et al., 2003], multiple evidence of a word, such as the acoustic features including various likelihood scores, the number of paths, the number of similar hypotheses and the contextual features including the acoustic confidence scores of surrounding words, were measured. Then, those features were combined and reduced to one confidence score using the Fisher linear discriminate analysis (FLDA), which was finally used for OOV word detection. Besides the acoustic features and the contextual features, [Lecouteux et al., 2009] also introduced some linguistic features derived from the language model and graph features from the confusion network. In their work, the adaptive boosting (AdaBoost) was applied to find the optimal combination of all features. In [Stouten et al., 2009], rather than only calculating features from the word recognition result, researchers also collected evidence from the phone recognition result. Among all different features

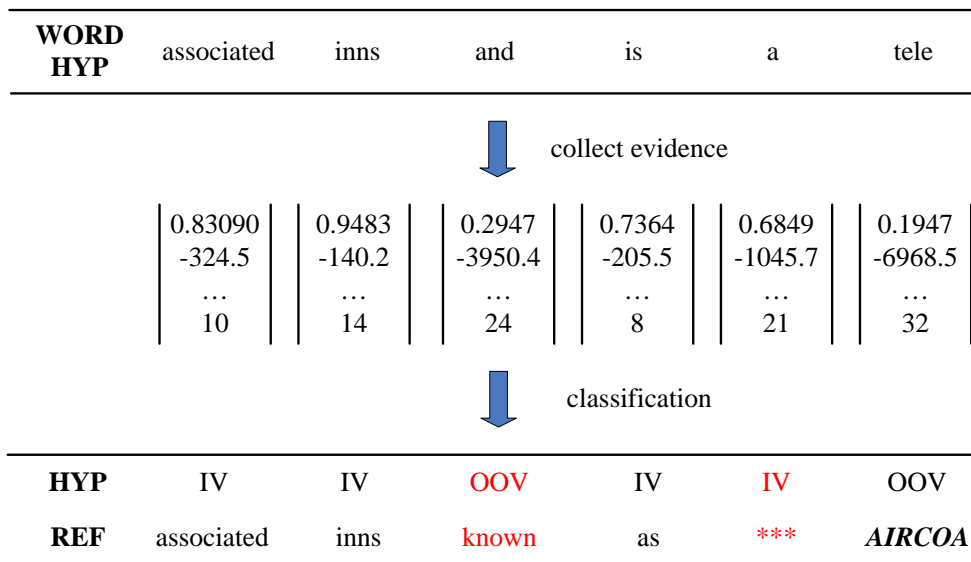


Figure 3.2: An example of detecting OOV word through classification.

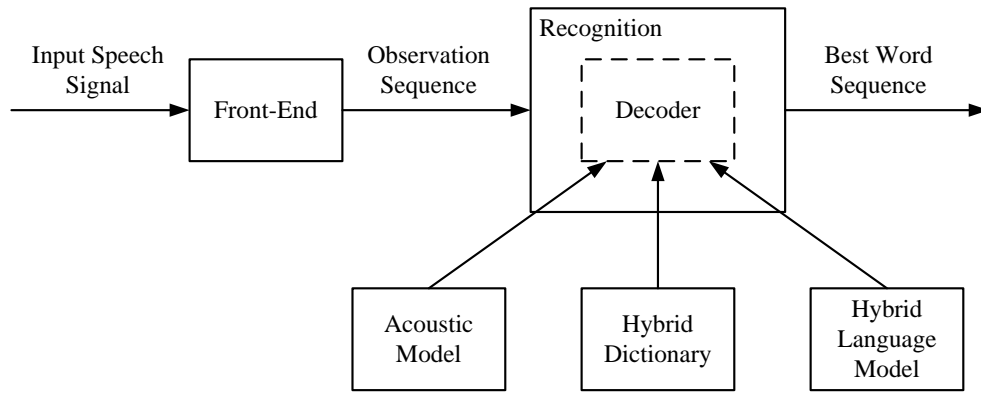
presented in those literatures, the acoustic and linguistic features are the most effective ones. And in most cases, systems using more types of features have a better performance than systems with fewer types of features.

In this OOV word classification framework, various features can be easily applied to detect OOV words. But the whole word units are not good to represent OOV words, especially for short OOV words. As shown in Figure 3.2, the system correctly detected the OOV word in this utterance. However, because it applied classification to all words in the recognition hypothesis, the system also produced an inserted IV word. Furthermore, within this framework, even we can locate the OOV regions, it is still not easy to retrieve the OOV pronunciations from the IV words units. Finally, it has the same problem as the previous method that sometimes the identified OOV regions are in fact recognition errors.

### 3.1.3 Using hybrid models to explicitly model OOV words

Different from the previous two approaches, a more direct way is to explicitly model and represent the OOV word using sub-lexical units, such as phones, syllables, subwords or graphemes, etc. Figure 3.3 presents the framework of a hybrid recognition system. It can be seen that the hybrid system is essentially the same as the general ASR system except that a hybrid lexicon and hybrid language model consisting of both words and sub-lexical units are used during decoding. As a result, OOV words in the testing speech will be recognized as sub-lexical unit sequences instead of IV words.

Depending on how the hybrid language model is trained, there are two different hybrid systems: the hierarchical hybrid system and the flat hybrid system. The hierarchical hybrid system was originally proposed in [Bazzi & Glass, 2000], where a generic word model and a base word language model were trained separately and then combined into a single hybrid language model. In their system, the generic word model was basically a phone language model, which is aimed



<b>HYBRID HYP</b>	associated	EH N Z	known	as	EH R K OW AH
↓					
<b>HYP</b>	IV	OOV	IV	IV	OOV
<b>REF</b>	associated	inns	known	as	<i>AIRCOA</i>

Figure 3.3: The hybrid ASR framework.

to absorb and recognize OOV words into phone sequences. [Klakow et al., 1999] first introduced the work of using a flat hybrid system with words and subword units for OOV word detection. In their system, they selected the top 5k words as IV words and considered all the rest words in the training data as “artificial” OOV words. The pronunciations of those “artificial” OOV words were then used as data for training the subword units. After that, the appearances of OOV words in the training text were substituted by corresponding subword units. At last, a flat hybrid language model was directly trained from the hybrid text data. In [Schaaf, 2001], a different head-tail hybrid model was presented, where a word was divided into a head unit and a base word unit. Then, the recognition was conducted using a language model incorporating both head and tail units. Graphones - the grapheme-phoneme pairs of letters and phones were proposed as another type of sub-lexical units in [Galescu, 2003]. The advantage of using a graphone hybrid system is that the written form of an OOV word can be obtained by simply concatenating the letter sequence of the decoded graphone units.

When the hybrid language model is trained, the information about which word categories OOV words are and where they appear in a sentence is implicitly embedded in the model. During decoding, the optimal path is searched by calculating the overall acoustic and language model scores of the whole utterance instead of each individual word. Therefore, more context evidence are measured in the hybrid method than the mismatch and classification methods. But in this framework, it is hard to utilize evidence other than just the acoustic and language model features during decoding to achieve further improvement.

### 3.1.4 Combining hybrid model and classification

Some researchers had proposed to combine the OOV word classifier and the hybrid system so as to improve the OOV word detection performance. Specifically, the hybrid system was first applied to decode the testing speech. Then, the reported OOV words in the decoding result were classified as true or false positives to remove possible detection errors. In [Yazgan & Saraclar, 2004], the OOV word candidates were filtered according to the confidence scores derived from the word lattice. In [Rastrow et al., 2009a], more evidence were tested in the classification step to obtain further improvement. Within the same framework, [Parada et al., 2010a] formulated the OOV word detection task as a sequential labeling problem, where the conditional random field (CRF) was applied to predict IV/OOV labels for a sequence of words.

In this method, OOV words are still explicitly represented using the hybrid model. Meanwhile, multiple evidence such as the acoustic features and lexical features can be easily applied to reduce the errors made by the hybrid system.

## 3.2 OOV word recovery

The most common way to recover the written form of an OOV word is through the phoneme-to-grapheme (P2G) conversion using a joint-sequence model trained from the recognition dictionary. In [Bisani & Ney, 2005, 2008], different lengths of graphemes were tested to find the optimal joint-sequence model for the P2G conversion. [Vertanen, 2008] presented that using the best path derived from the confusion network was better than using the 1-best hypothesis for OOV word recovery. A different recovery method was presented in [Rastrow et al., 2009b; Hanemann et al., 2010], where a lexical finite state transducer (FST) built from a large dictionary was used to translate the decoded sub-lexical units into words.

[Parada et al., 2010b] introduced another interesting approach to OOV word recovery that involved with an information retrieval (IR) system and a spoken term detection (STD) system. They used the surrounding words of a possible OOV word as search query to search the web and find sentences containing those query words. Then candidate words appearing at the same locations as OOV words in each sentence were selected as key words of the STD system. Finally, testing speech was input into the STD system, if there was an OOV word, the STD system would spot that word from its key word list.

## 3.3 Summary

In above paragraphs, we discussed the current solutions to the OOV word problem in speech recognition. Different techniques had been investigated to solve this problem from different directions. But to finally convert an OOV word into an IV word, so that it can be recognized when encountered by the ASR system in the future, we have to estimate  $n$ -gram language model scores for the OOV word. Moreover, the same OOV word can appear more than once in a conversation or over a period of time. Such multiple instances of the same OOV word provide valuable information for estimating the pronunciation or the language model score of the word. However, there is very limited work on those two topics in speech recognition.





# Chapter 4

## OOV Word Detection

In this chapter, we study the OOV word detection problem, which is detecting the location of an OOV word in the testing speech. We explored two different approaches for training hybrid systems, the hierarchical and flat hybrid systems. We also compared building hybrid systems using different types of sub-lexical units, such as the phone, syllable, subword and grapheme units. Then, the OOV word classifier was presented to remove false detections. Furthermore, we demonstrated two system combination techniques to improve the OOV word detection performance. And at last, the experiment setup and experiment results are provided.

### 4.1 OOV word detection using hybrid systems and OOV word classifier

#### 4.1.1 Hybrid systems

In a hybrid system, a hybrid lexicon and hybrid language model (LM) consisting of both words and sub-lexical units were applied during decoding to detect the presence of OOV words. The hybrid lexicon was obtained by incorporating the sub-lexical units and their pronunciations into the word lexicon, while the hybrid LM could be trained in two different manners.

##### The hierarchical hybrid LM

As shown in Figure 4.1, the hierarchical hybrid LM was generated by merging an open-vocabulary word LM trained from a large text corpus and a closed-vocabulary sub-lexical LM trained from the pronunciations of IV words. When training the word LM, all OOV words were matched to the same unknown token “ $\langle unk \rangle$ ”. Then by combining the word LM with the sub-lexical LM, a single sub-lexical word hybrid LM was produced. For example, the unigram probability of a sub-lexical unit in the hybrid LM was calculated as

$$P_H(f_i) = P_W(\langle unk \rangle) \cdot P_S(f_i) \cdot C_{OOV}, \quad (4.1)$$

where  $P_W(\langle unk \rangle)$  is the unigram probability of the unknown token  $\langle unk \rangle$  in the word LM,  $P_S(f_i)$  is the unigram probability of the sub-lexical unit  $f_i$  in the sub-lexical LM, and  $C_{OOV}$  is the

cost of entering an OOV word during decoding. Similarly, we can compute  $n$ -gram probabilities in the hierarchical hybrid LM.

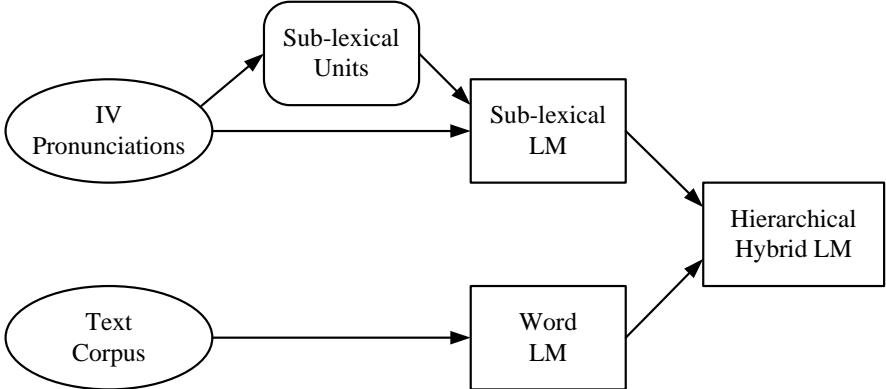


Figure 4.1: The training process of the hierarchical hybrid LM.

**The flat hybrid LM**

As shown in Figure 4.2, to train a flat hybrid LM, rather than separately training the word LM and the sub-lexical LM then merging them into one hybrid LM, we directly built a flat hybrid LM from the hybrid text corpus. Such hybrid text data was produced by substituting OOV words in the training text with corresponding sub-lexical units derived from the OOV pronunciations. In our system, the pronunciations of OOV words were estimated through the grapheme-to-phoneme (G2P) conversion. We could also obtain OOV pronunciations by looking up larger dictionaries. Again, when training the flat hybrid LM, we assigned a OOV cost  $C_{OOV}$  to control how likely to encounter an OOV word during decoding. By tuning  $C_{OOV}$ , we can find an optimal configuration of our system to achieve the target OOV word detection performance.

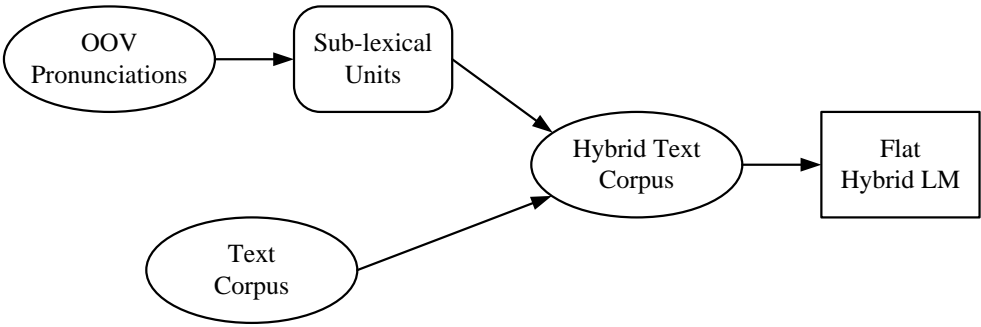


Figure 4.2: The training process of the flat hybrid LM.

## Word boundaries

When training the hybrid LM, sometimes two or more OOV words might appear consecutively in the training data. After replacing OOV words with their corresponding sub-lexical units, the word boundary between two OOV words was lost. Hence, a sequence of sub-lexical units in the OOV word detection output might actually match to multiple consecutive OOV words in the testing speech. To solve this problem, we added two more symbols into the sub-lexical units of each OOV word, which were the word start “^” and word end “\$”. By doing this, the sub-lexical units at the word boundary and sub-lexical units within a word were treated differently during training. As a result, the context information about the sub-lexical unit position was also preserved in the hybrid model.

### 4.1.2 Sub-lexical units

Four different types of sub-lexical units, phone, syllable, subword, and grapheme units, were investigated for OOV word detection [Qin et al., 2011]. Among those units, phone, syllable and subword units only model the phonetic level of a word, while grapheme units also consider the orthography level.

#### Phone

In the phone hybrid system, the hybrid lexicon and hybrid LM were composed with both whole word entries and monophone entries. Then during decoding, OOV words were absorbed and represented by phone sequences. For example, our system recognized the OOV word “ashland” as “^AE SH AH N\$”. The phone hybrid system was simple to build. However, as we modeled OOV words with a very small number of phones and the phone recognition was not reliable, the OOV word detection performance was usually not satisfied. Therefore, we explored to build hybrid systems with more complex sub-lexical units.

#### Syllable

Syllables are often considered as the phonological “building blocks” of words, which can influence the rhythm, prosody and stress of a word. The general structure of a syllable consists of three components: the onset, nucleus, and coda. The nucleus is normally a vowel or a diphthong, while the onset and coda are usually optional consonants. For instance, the word “water” can be split into two syllables: “^W\_AO” and “T\_ER\$”. In our system, we segmented OOV pronunciations into syllables using the Festival lexicon tools [Black et al., 1997].

#### Subword

Similar to syllable units, subwords, such as “^AH\_N” and “EY\_SH\_AH\_N\$”, are iteratively trained phone sequences with variable lengths [Klakow et al., 1999]. First, we initialized the subword inventory with all phones to ensure the full coverage of all possible OOV words. In each iteration, the most frequent subword bigram was merged and added to the subword inventory. Its occurrences in the training data were also concatenated into one single entry. This transformed

training data was then used in the next iteration. The training ended when a target number of subword units was reached.

## Graphone

A graphone is a grapheme-phoneme pair of English letters and phones. For example, one possible graphone representation of the word “speech” is

$$\text{speech} = \begin{pmatrix} s \\ \wedge S \end{pmatrix} \begin{pmatrix} pee \\ P\_IY \end{pmatrix} \begin{pmatrix} ch \\ CH\$ \end{pmatrix}.$$

To find graphone units, a trigram joint-sequence model was trained and then used to segment words into grapheme-phoneme pairs [Bisani & Ney, 2008]. A graphone can have a minimum and maximum number of letters and phones. Here, we used the same range for both letters and phones, where the minimum was set to 1 and the maximum was varied from 2 to 5.

### 4.1.3 The OOV word classifier

As mentioned in Chapter 3, it is much easier to utilize various evidence for OOV word detection using the OOV word classifier. Therefore, we integrated the OOV word classification component into our hybrid system framework to improve the OOV word detection performance. Specifically, the hybrid decoding was first performed to find possible OOV words. From the hybrid decoding results, we collected four types of evidence for each OOV hypothesis, which are the acoustic, lexical, lattice, and contextual features. Those features were then inputted into a classifier to classify whether the OOV hypothesis was a true or false positive. In this thesis, we trained LogitBoost classifiers with feature selection using Weka on the hybrid decoding result of the development data [Hall et al.].

- *Acoustic Features*: the number of phones, the duration per phone, the acoustic score, the posterior score
- *Lexical Features*: the LM score, the unigram LM score, the LM back-off activity
- *Lattice Features*: the number of competing hypotheses, the number of competing OOV hypotheses, the number of filler hypotheses, the sum of posterior probability of OOV hypotheses, the sum of posterior probability of filler hypotheses
- *Contextual Features*: the acoustic, lexical and lattice features of preceding and succeeding words

## 4.2 System combination for OOV word detection

### 4.2.1 Combining multiple hybrid systems’ outputs

In previous sections, we introduced hybrid systems with different sub-lexical units for solving the OOV word problem. Here we present a method to improve the OOV word detection performance by utilizing multiple hybrid systems [Qin et al., 2012]. Specifically, the recognizer output voting

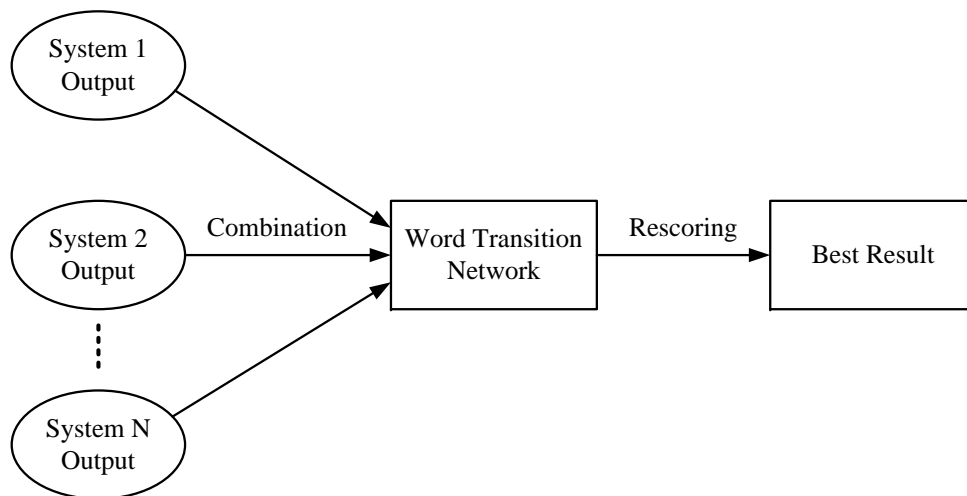


Figure 4.3: The ROVER framework.

error reduction (ROVER) was applied to combine different systems' outputs [Fiscus, 1997]. Two combination metrics, i.e., voting by word frequency and voting by both word frequency and word confidence, were explored.

## ROVER

ROVER was originally developed at the National Institute of Standards and Technology (NIST) to produce composite speech recognition system output when multiple recognizers' results are available. In many cases, the composite recognition output has lower word error rate (WER) than any of the individual recognizers. In ROVER, the multiple recognizers' outputs are first combined into a single, minimal cost word transition network (WTN) via iterative applications of dynamic programming (DP) alignments. Then, the resulting WTN is re-scored and searched to find the optimal word sequence. Figure 4.3 shows the general ROVER framework.

A commonly used re-scoring formula in ROVER is

$$Score(w_i) = \alpha \cdot \frac{N(w_i)}{\sum_w N(w_i)} + (1 - \alpha) \cdot C(w_i), \quad (4.2)$$

where  $N(w_i)$  is the count of word  $w$  at the  $i$ -th alignment in the WTN,  $C(w_i)$  is the word confidence score of  $w_i$ , and  $\alpha$  is the weight used to balance the word frequency and the word confidence score. Another parameter  $C(@)$  is used to set the confidence score of the NULL transition arc. For details of ROVER, please refer to [Fiscus, 1997].

### ROVER for OOV word detection

Two ROVER systems with different re-scoring modules were investigated in this work. In the baseline system,  $\alpha$  in Eqn. 4.2 was set to 1, the optimal word sequence was found by only considering the frequency of word occurrences in each alignment in the WTN. In the second ROVER system,  $\alpha$  was set to a value between 0 and 1, so that both the word occurrences and word confidence scores were measured when re-scoring the WTN.

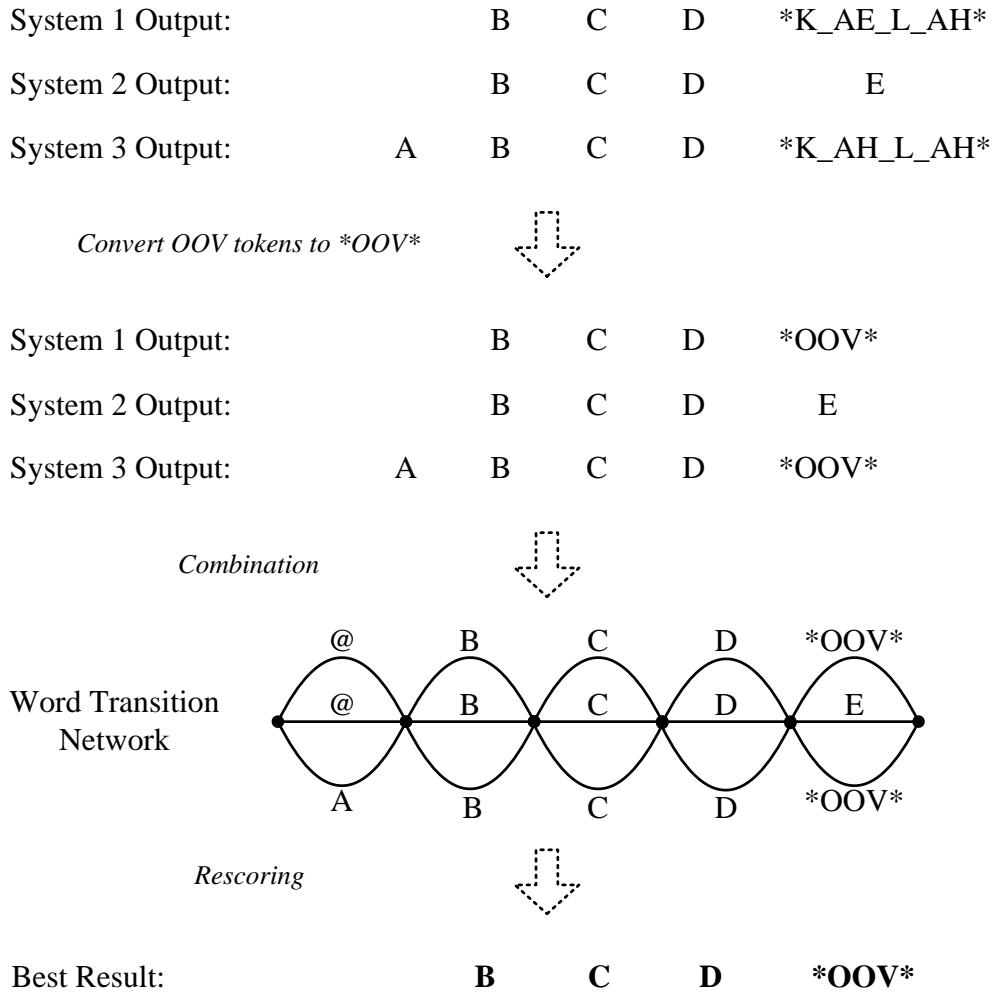


Figure 4.4: The ROVER combination process.

Because we used ROVER for OOV word detection instead of speech recognition, we were more concerned about where the OOV word occurred in the testing speech than what was the correct pronunciation of that word. When calculating the confidence score  $C(w_i)$  in Eqn. 4.2 for  $w_i$  in the lattice of the  $j$ -th hybrid system, we did not measure the confidence of that single word. Alternatively, depending on whether  $w_i$  is an IV or OOV word, we summed over the posterior probabilities of all IV or OOV words in that region.

$$Conf_j(w_i) = \sum_{k \in [s_i, e_i]} \begin{cases} P(IV_k) & w_i \text{ is IV} \\ P(OOV_k) & w_i \text{ is OOV,} \end{cases} \quad (4.3)$$

where  $s_i$  and  $e_i$  are the start and end time of  $w_i$ , and  $P(IV_k)$  is the posterior probability of an IV word in that region, while  $P(OOV_k)$  is the posterior probability of an OOV word.  $Conf_j(w_i)$  is then normalized by the sum of posterior probabilities of all words in that alignment to make sure  $Conf_j(w_i) \in [0, 1]$ . There are two ways to compute  $C(w_i)$  from the individual confidence score  $Conf_j(w_i)$ , i.e., the average and the maximum of individual scores. In our system, we found that

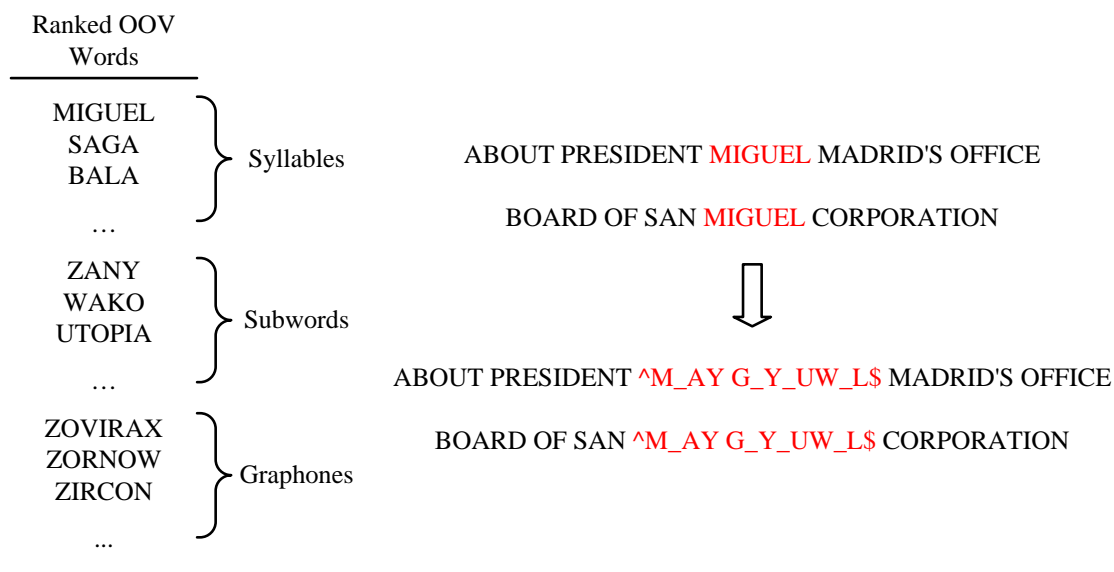


Figure 4.5: An example of modeling OOV words in the mix-hier system.

the performance of those two methods was similar. The optimal values of  $\alpha$  and  $C(@)$  in Eqn. 4.2 were searched from 0 to 1 with a step size of 0.2 using the grid search. And multiple outputs of individual systems were always aligned to the one with the best performance.

Since different hybrid systems usually produce different sub-lexical sequences for the same OOV word, ROVER cannot be directly applied to the OOV word detection results. Therefore we converted all sub-lexical sequences in multiple recognizers' outputs to the same OOV token "**\*OOV\***". IV words were not changed so as to have a better alignment when building the transition network. As presented in Figure 4.4, both *System 1* and *System 3* reported an OOV word in their output. However, *System 1* recognized the OOV word as "K\_AE\_L\_AH", while *System 3* recognized it as "K\_AH.L\_AH". Therefore in our ROVER system, the first step was to convert different sub-lexical sequences into the same token "**\*OOV\***". Then ROVER was applied to combine and find the best result from multiple systems' outputs. If there are OOV words in the ROVER result, we can trace back to individual systems' outputs, then combine different sub-lexical sequences to obtain a better phonetic representation of the OOV word.

## 4.2.2 Combining multiple types of sub-lexical units

Different types of sub-lexical units have their own advantages and problems. For example, sub-words are simple robust units, yet lack linguistic considerations. Syllables maintain phonetic restrictions to form words, but occasionally produce problematic long rare units. Graphones model both the written form and pronunciation of a word, however, the number of graphone units explodes when allowing longer letter or phone sequences. Therefore, besides combining multiple hybrid systems' outputs, we investigated system combination from another direction by utilizing multiple types of sub-lexical units in one system, so that different units can complement each other [Qin & Rudnicky, 2012].

ABOUT PRESIDENT MIGUEL MADRID'S OFFICE

BOARD OF SAN MIGUEL CORPORATION



For each appearance of MIGUEL,  
stochastically select one type of units

ABOUT PRESIDENT ^M\_AY G\_Y\_UW\_L\$ MADRID'S OFFICE

BOARD OF SAN ^MI:M\_AY G:G UEL\$:Y\_UW\_L CORPORATION

Figure 4.6: An example of modeling OOV words in the mix-flat system.

We implemented two methods to use mixed types of sub-lexical units in our hybrid system. In the first method, hereafter referred as “*mix-hier*”, OOV words were divided into three groups and separately modeled by the syllable, subword, and grapheme units. This is similar to [Shaik et al., 2011], in which researchers used the whole word units together with syllable or morpheme units to model IV words and grapheme units for OOV words in an open vocabulary German speech recognition system. In the second method, hereafter referred as “*mix-flat*”, each occurrence of OOV words was modeled by one type of unit stochastically selected from three types of sub-lexical units. Compared with the first method, where each OOV word is only modeled by one type of sub-lexical unit, in the second method, an OOV word could be modeled by multiple types of sub-lexical units if it occurs more than once in the training data.

### The mix-hier combination method

In this method, different types of sub-lexical units were combined in a hierarchical way, where we divided OOV words into three groups and used one type of sub-lexical unit for each group. Because the syllable system usually performed better than the subword and grapheme systems, we modeled the most frequent OOV words using syllable units and the less frequent OOV words using subword and grapheme units. In particular, we first ranked all OOV words based on their frequencies in the training text. Then, we used syllable units to model the top  $x$  percent of OOV words, subword units to model the following  $y$  percent of OOV words, and grapheme units for the remaining OOV words. The value of  $x$  and  $y$  were tuned over the development data to get the best OOV word detection performance. For example, as shown in Figure 4.5, those two sentences both contain the OOV word “MIGUEL”. Since we selected to model “MIGUEL” using syllables, all occurrences of the word in the training data were represented using the same sequence of syllable units.

### The mix-flat combination method

In this method, different types of sub-lexical units were combined in a flat way, where each occurrence of OOV words was modeled by one type of sub-lexical unit stochastically selected from syllables, subwords, and graphemes. We trained each type of sub-lexical unit using the estimated OOV pronunciations. Then, for each OOV occurrence in the training text, we represented it with



the sub-lexical units from the stochastically selected unit type. As a result, different from the mix-hier method, here, one OOV word can be modeled by multiple types of sub-lexical units, if it occurs more than once in the training data. For instance, in Figure 4.6, we have the same example as in the mix-hier system. But here, the first “MIGUEL” was modeled using syllable units, while the second one was modeled using grapheme units. The mix-flat system could therefore have a better coverage of OOV words from all three types of sub-lexical units. Furthermore, this mix-flat method doesn’t require any development data for tuning.

## 4.3 Experiment setup and datasets

### 4.3.1 Experiment setup

#### CMU SPHINX speech recognition system

In this thesis, all experiments were conducted using the CMU SPHINX speech recognition system [Lee et al., 1990]. SPHINX is an open source toolkit for speaker-independent large vocabulary continuous speech recognition [sph]. It comes with an acoustic model trainer - Sphinx-Train and a number of decoders including Sphinx3, Sphinx4, PocketSphinx and MultiSphinx [Huggins-Daines, 2011]. Among all those decoders, the Sphinx3 decoder is the most accurate one, thus we used it for our experiments. To train the acoustic model, 13-order mel-scale frequency cepstral coefficients (MFCCs) including  $c_0$  were extracted from the input speech, and then the delta and delta-delta coefficients were calculated. The feature dimension was reduced to 29 after applying the linear discriminant analysis (LDA) [Haeb-Umbach & Ney, 1992] and the maximum likelihood linear transform (MLLT) [Gales, 1998]. Finally, 3-state left-to-right HMMs were trained under the maximum likelihood (ML) criterion and the maximum mutual information (MMI) criterion [Qin & Rudnicky, 2010a,b].

#### Hybrid model training

For the hybrid LM training, we selected the most frequent words from the LM training text data as vocabulary. Then the recognition lexicon was produced by looking up pronunciations for iv-vocabulary (IV) words from the CMUdict (v.0.7.a) [Rudnicky, 2007]. To build the hierarchical hybrid model, we combined an open-vocabulary word bigram LM trained from the LM training data and a closed-vocabulary sub-lexical bigram LM trained from the IV pronunciations. To build the flat hybrid model, we directly trained a bigram model from the hybrid text data by substituting OOV words with corresponding sub-lexical units derived from the OOV pronunciations. In our experiments, we adjusted the OOV cost  $C_{OOV}$  from 0 to 2.5 with a step size of 0.25 to control the penalty of entering an OOV word during decoding. By doing this, we could find the optimal  $C_{OOV}$  for different applications, such as applications in favor of finding as many OOV words as possible or applications in favor of not misrecognizing IV words as OOV words.

#### Evaluation metrics

We used the following metrics to evaluate the OOV word detection performance:

- *Miss Rate (MR)*, which measures how many OOV words are missed in the OOV word detection output. *MR* can be calculated as

$$MR = \frac{\#OOVs \text{ in reference} - \#OOVs \text{ detected}}{\#OOVs \text{ in reference}} \times 100\%. \quad (4.4)$$

- *False Alarm Rate (FAR)*, which measures how many IV words are falsely reported as OOV words. *FAR* can be calculated as

$$FAR = \frac{\#OOVs \text{ reported} - \#OOVs \text{ detected}}{\#IVs \text{ in reference}} \times 100\%. \quad (4.5)$$

- *Recall*, which measures how many OOV words are detected. *Recall* can be calculated as

$$Recall = \frac{\#OOVs \text{ detected}}{\#OOVs \text{ in reference}} \times 100\% = 100\% - MR. \quad (4.6)$$

- *Precision*, which measures how many reported OOV words are correct detections. *Precision* can be calculated as

$$Precision = \frac{\#OOVs \text{ detected}}{\#OOVs \text{ reported}} \times 100\%. \quad (4.7)$$

- *F1*, which measures both the *Recall* and *Precision* of the OOV word detection output. *F1* can be calculated as

$$F1 = \frac{2 \cdot Precision \cdot Recall}{Precision + Recall}. \quad (4.8)$$

We calculated *MR*, *FAR*, *Recall*, *Precision* and *F1* at the word level, which measure both the presence and positions of OOV words in an utterance. This is because that in practical applications, knowing where OOV words are located is more valuable than simply knowing the fact that OOV word(s) exist in an utterance.

### 4.3.2 Datasets

As we did not have data specifically designed for evaluating OOV word detection, the proposed work was tested on three conventional ASR corpora, the Wall Street Journal (WSJ), the Broadcast News (BN) and the Switchboard (SWB).

- The *WSJ* corpus contains clean read speech recordings of the WSJ text materials. We used the WSJ0 and WSJ1 data [Garofalo et al., 1993, 1994] with a total of 80 hours speech for acoustic model training, and the 1987-1989 WSJ text for language model training [Graff et al., 1995]. We selected the top 20k words to build the vocabulary. The development data was the WSJ 1992 and 1993 20k-word speaker-independent Dev sets, while the evaluation data was the WSJ 1992 20k-word and 1993 64k-word speaker-independent Eval sets.
- The *BN* corpus contains speech recordings from televisions and radio networks. We used a subset of the 1996 and 1997 BN data [Graff et al., 1997; Fiscus et al., 1998] with a total

of 140 hours speech for acoustic model training, and the 1992-1997 BN transcripts for language model training. Again, we selected the top 20k words to build the vocabulary. The development and evaluation data were the F0 and F1 tasks from the 1996 HUB4 Dev and Eval sets.

- The *SWB* corpus contains spontaneous telephone conversation recordings. We used the SWB1 phrase 1 data [Godfrey & Holliman, 1997] with a total of 180 hours speech for acoustic model training, and the SWB1 and SWBCELL transcripts together with the CALL-HOME and FISHER data for language model training. Here, we selected the top 10k words to build the vocabulary. The development and evaluation data were randomly selected sessions from the SWB1 phrase 2 recordings.

The above three datasets are all commonly used corpora in speech recognition. They are different from each other with regard to content materials, recording conditions and speaking styles. The WSJ data is read speech recorded in quiet conditions, which is the simplest recognition task among three tasks. The BN data is the recordings of television and broadcast news including some portion of conversations and sometimes background noise. Thus it is harder to recognize than the WSJ speech. While the SWB corpus contains recordings of spontaneous telephone conversations, which is normally more difficult than read speech or broadcast news speech. In addition, to handle the deletions and repetitions in the spontaneous speech, we added many word fragment units into the SWB recognition lexicon. Most word fragment units are very similar to the sub-lexical units, so OOV words in the testing speech might be instead recognized as word fragments.

Table 4.1: The OOV rate of the Dev and Eval data.

	Dev Data		Eval Data		Common OOVs in Dev and Eval
	NO. OOV	OOV RATE	NO. OOV	OOV RATE	
WSJ	313	2.10%	200	2.20%	6
BN	204	2.02%	255	2.04%	5
SWB	204	1.74%	209	1.69%	4

From our initial experimental results, we found that some OOV words in the development and evaluation data were morphological changes of IV words, such as TANKS and LUMBERING. In practice, we can prevent this kind of OOV words by adding all morphology forms of a word into the lexicon. As such OOV words are not the new words our system expects to detect, we manually examined OOV words in the development and evaluation data and added those fake OOV words into the lexicon. We also removed a few less frequent words from the lexicon to maintain the vocabulary size. The final OOV rate of each task is given in Table 4.1. It can be found that the WSJ and BN tasks have an OOV rate around 2%, while the SWB task has a slightly lower OOV rate. The OOV rate is similar to many practical LVCSR applications, such as the *voice search* system on mobile phones. From Table 4.1, we can also find that there are only very few OOV words appearing in both the Dev and Eval data. Therefore, by tuning OOV word detection performance on the Dev data, we only learned parameters of the hybrid system but not about OOV words in the Eval data. We also measured the percentage of Dev and Eval OOV words in the training speech and text corpora. As shown in Table 4.2, a small portion of

Dev and Eval OOV words appear in the training speech, while a large portion of OOV words appear in the training text in the WSJ and BN tasks. In the SWB task, all Dev and Eval OOV words are in the training data.

Table 4.2: The percentage of Dev and Eval OOV words in the training speech and text data.

	Dev Data		Eval Data	
	In Speech	In Text	In Speech	In Text
WSJ	13%	98%	17%	98%
BN	50%	88%	33%	83%
SWB	100%	100%	100%	100%

## 4.4 Experimental results

In this proposal, the OOV word detection performance of different systems were compared using the FAR-MR and F1 curves. To draw those curves, we adjusted the OOV cost  $C_{OOV}$  when training the hybrid systems. In general, the OOV cost  $C_{OOV}$  was adjusted from 0 to 2.5 with a step size of 0.25.

### 4.4.1 Baseline word recognition results

As we discussed in 4.3.2, these three recognition tasks are quite different from each other in terms of channels, recording conditions and speaking styles, etc. Table 4.3 presents the baseline word recognition results using the bigram and trigram LMs. We can find that the WSJ task has the best recognition accuracy, the BN task gets a higher WER around 30%, and the SWB task has a WER up to 40%. To be noticed, these results are speaker independent decoding results without adaptation. Furthermore, in this thesis, we didn't focus on building the most accurate ASR systems. However, we did find that better OOV word detection performance was achieved with a more accurate recognizer.

Table 4.3: The baseline word recognition results.

WER (%)	Bigram LM		Trigram LM	
	Dev Data	Eval Data	Dev Data	Eval Data
WSJ	15.70	13.77	13.48	12.04
BN	29.00	31.23	25.93	28.28
SWB	39.13	39.27	32.58	33.54

## 4.4.2 Baseline OOV word detection results

### Finding the optimal number of subword units and optimal grapheme length

To determine the optimal number of sub-lexical units when building the subword and grapheme hybrid systems, we simply built multiple hybrid systems with different number of subword units or different grapheme lengths and chose the system with the best performance on the development data. If two systems performed similarly, we preferred the slightly sub-optimal system to prevent the over training problem. In those experiments, to save computations, we only tested hybrid models with an OOV cost  $C_{OOV}$  of 0.25, 0.75, 1.25 and 1.75.

From the results given in Figure 4.7 and Figure 4.8, we can find that the optimal number of subword units and grapheme lengths are different for each task. More subword units and longer grapheme units are used in the flat hybrid systems than in the hierarchical hybrid systems. This is because, as provided in Table 4.4, the subword and grapheme units were trained from the pronunciations of IV words in the hierarchical hybrid system. But in the flat hybrid system, we trained sub-lexical units from the pronunciations of a large number of OOV words. As a result, there were more data for training the sub-lexical units in the flat hybrid system, so that it could afford more subword units and longer grapheme lengths. The optimal number of subword units and grapheme length in each task is summarized in Table 4.5.

Table 4.4: The number of words used for training sub-lexical units.

	WSJ	BN	SWB
Hierarchical System	20k	20k	10k
Flat System	150k	240k	50k

Table 4.5: The optimal number of subword units and grapheme length.

	Number of Subword Units			Grapheme length		
	WSJ	BN	SWB	WSJ	BN	SWB
Hierarchical System	1000	500	500	3	2	4
Flat System	7000	3000	4500	4	3	4

As “^” and “\$” were combined with the sub-lexical units at the word boundaries, there were a large number of sub-lexical units in our hybrid systems. The total number of sub-lexical units in the phone, syllable, subword and grapheme hybrid systems are given in Table 4.6 and Table 4.7. In the grapheme hybrid system, the number of graphemes is determined by the grapheme length. If only allowing short graphemes, OOV words will be represented by a sequence of short grapheme units that are hard to accurately recognize. On the other hand, if allowing longer graphemes, OOV words will be composed by fewer long grapheme units that are more robust for recognition. But in this case, the number of different grapheme units will also dramatically increase. In the subword hybrid systems, the number of subword units were fully controlled in training. And there are much fewer sub-lexical units in the subword system than in the syllable and grapheme systems.

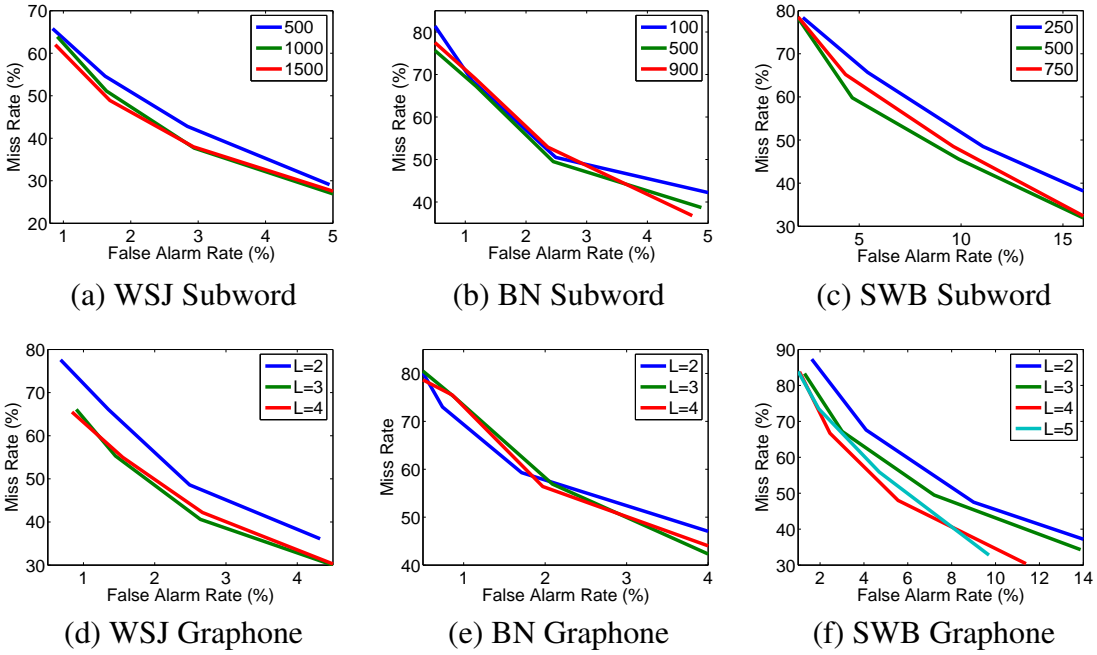


Figure 4.7: The OOV word detection performance of hierarchical hybrid systems with different number of subword units and different graphone lengths.

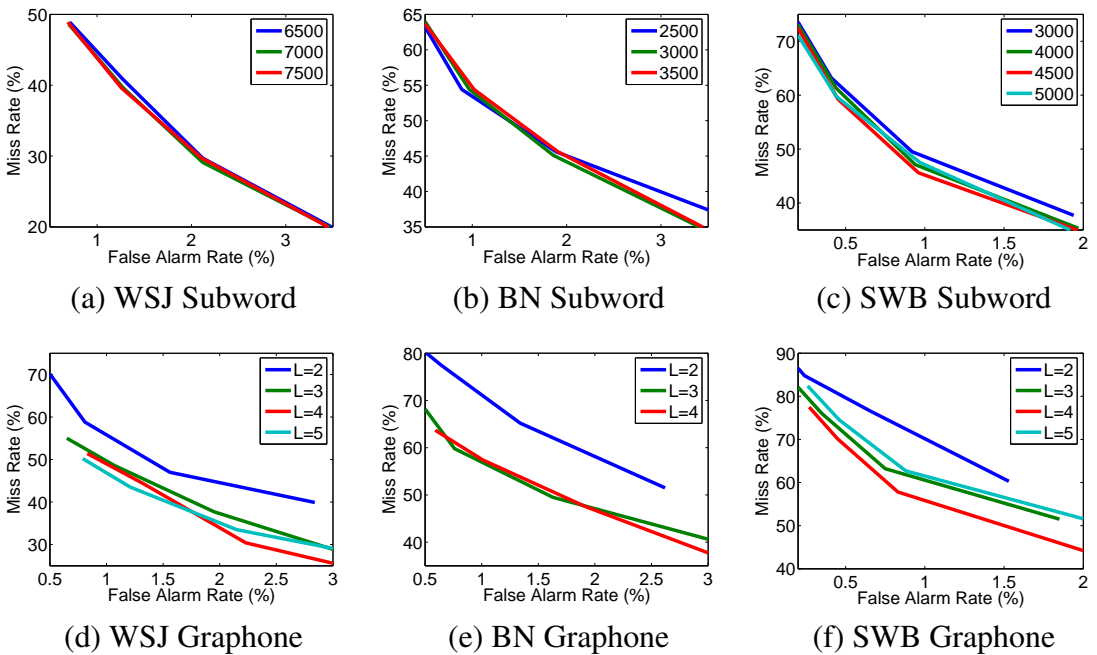


Figure 4.8: The OOV word detection performance of flat hybrid systems with different number of subword units and different graphone lengths.

Table 4.6: The number of sub-lexical units in different hierarchical hybrid systems.

	Phone	Syllable	Subword	Graphone
WSJ	118	8703	1000	9151
BN	124	8915	500	3002
SWB	149	5989	500	8938

Table 4.7: The number of sub-lexical units in different flat hybrid systems.

	Phone	Syllable	Subword	Graphone
WSJ	151	26855	7000	19579
BN	152	34297	3000	13616
SWB	152	15264	4500	10298

### Comparison of hybrid systems with different sub-lexical units

First, we compare the OOV word detection performance of hierarchical hybrid systems using different sub-lexical units. As provided in Figure 4.9, we can find that the syllable, subword and graphone hybrid systems are usually better than the phone hybrid system. The reason is that those systems can utilize a longer history of phones for modeling OOV words. For instance, in the syllable system, a sub-lexical unit “AE\_N\_T” already incorporates a history of two phones. However, in the phone hybrid system, any phone bigram can only rely on the previous phone. Among three tasks, the performance difference between different hybrid systems is quite small in the WSJ and BN tasks, but larger in the SWB task.

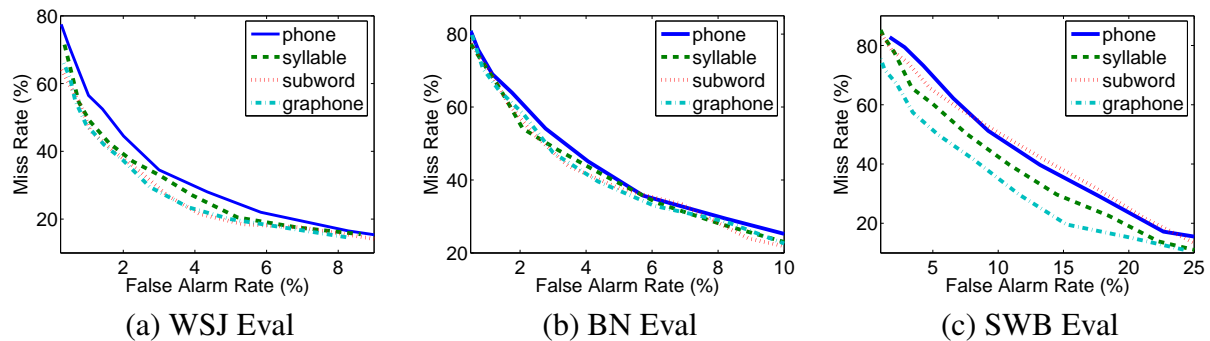


Figure 4.9: The OOV word detection performance of hierarchical hybrid systems with different sub-lexical units.

The OOV word detection results of flat hybrid systems with different sub-lexical units are provided in Figure 4.10. Similar to the hierarchical results, the syllable, subword and graphone systems are all better than the phone system. In the flat hybrid system, the difference between the phone system and the other three systems is much larger, which is probably because much more complex sub-lexical units were applied in the flat hybrid system. Among the syllable, subword and graphone systems, the syllable system is slightly better than the other two systems.

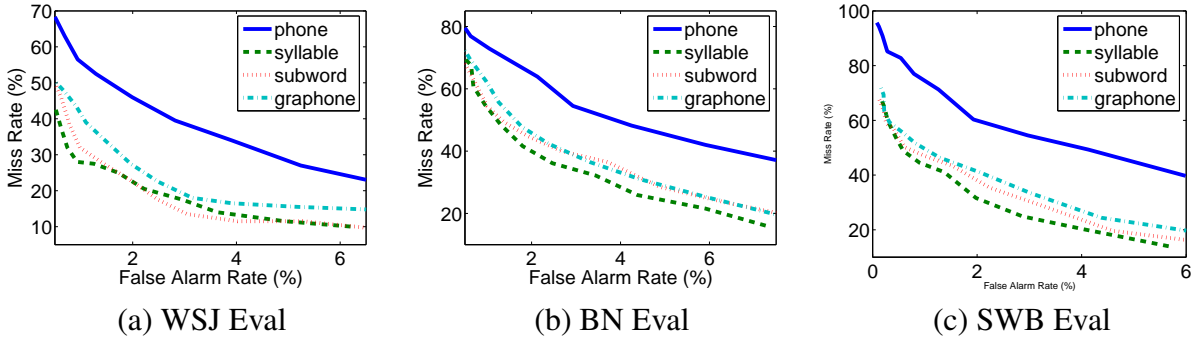


Figure 4.10: The OOV word detection performance of flat hybrid systems with different sub-lexical units.

We also calculated the weighted length of sub-lexical units in each system. The length of a sub-lexical unit is defined as the number of phones it contains. A longer sub-lexical unit contains more phones and is thus easier and more robust for recognition. On the other hand, if we represent OOV words only using longer sub-lexical units, we will end up with a very large number of units, which then require more data for training. To account for the distribution of sub-lexical units, the length of a unit is weighted by its frequency in the training data. As shown in Table 4.8 and Table 4.9, the length of the phone unit is always 1 and is the smallest. This is one reason why the phone system is usually worse than the other hybrid systems. In the hierarchical systems, although the syllable unit has the largest length, the performance of the syllable system is not better than the subword and graphone systems. However, in the flat systems, the length of the syllable unit is the largest and the syllable system also has the best performance among all hybrid systems. As mentioned in the previous paragraphs, there were usually a very large number of sub-lexical units used in the syllable system. Therefore, it required more data for training the hybrid LM using those units. In the hierarchical system, the sub-lexical units were trained from the IV pronunciations, which is a very small dataset. But in the flat system, the training data was the large number of OOV words in the LM training text, which could afford more longer sub-lexical units. We can also find that in the flat systems, the graphone system has a weighted length smaller than the syllable and subword units, and its performance is also worse than the syllable and subword systems. The performance of the graphone system may be better, if there are more data for training longer graphone units. Basically, there is a correlation between the weighted length of sub-lexical units and the OOV word detection performance in the flat hybrid systems.

Table 4.8: The weighted length of sub-lexical units in different hierarchical hybrid systems.

	Phone	Syllable	Subword	Graphone
WSJ	1.00	2.64	2.00	2.12
BN	1.00	2.64	1.75	1.50
SWB	1.00	2.66	1.71	2.66



Table 4.9: The weighted length of sub-lexical units in different flat hybrid systems.

	Phone	Syllable	Subword	Graphphone
WSJ	1.00	2.54	2.58	2.27
BN	1.00	2.50	2.21	1.94
SWB	1.00	2.59	2.48	2.21

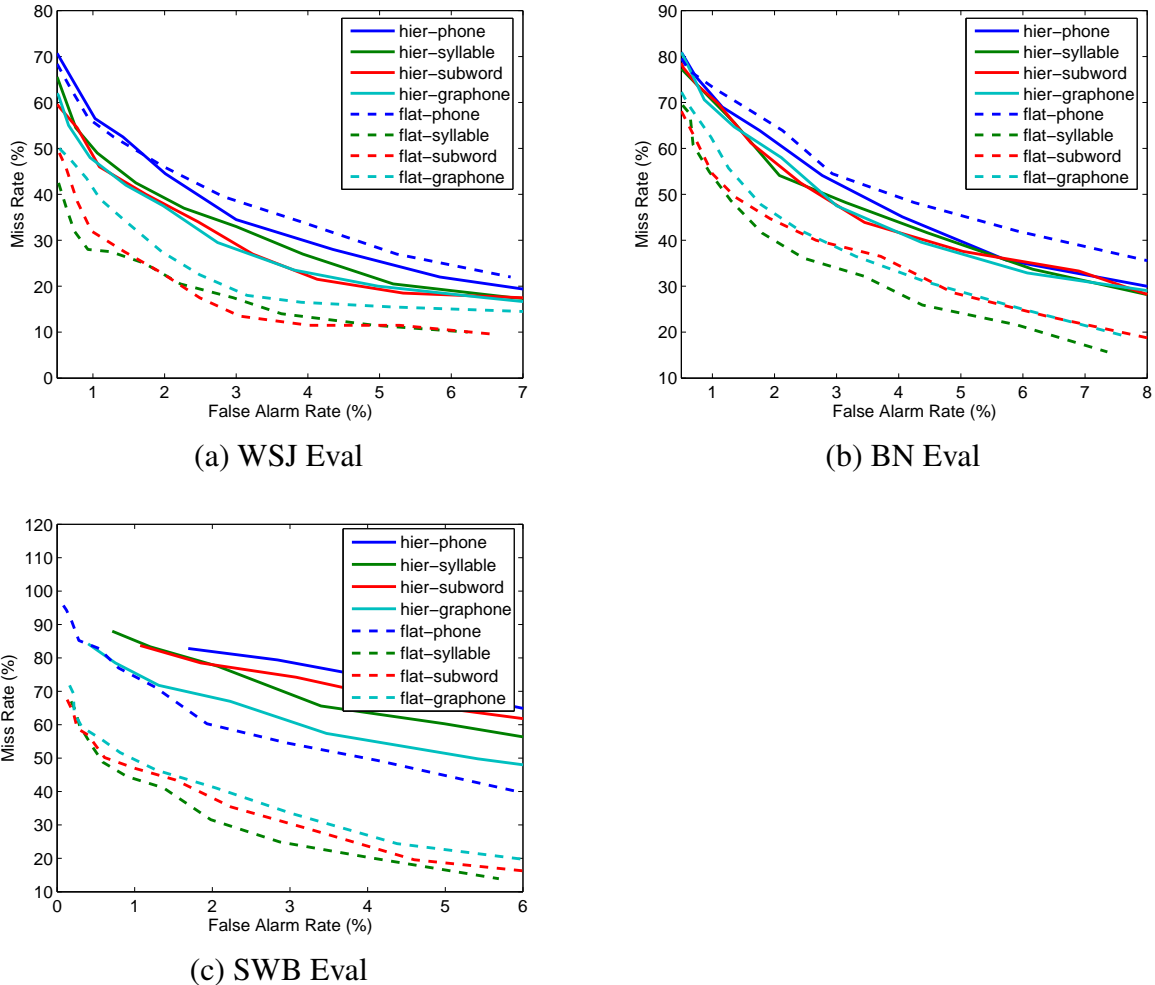


Figure 4.11: The OOV word detection performance of hierarchical and flat hybrid systems.

### Comparison of the hierarchical hybrid system and flat hybrid system

In the previous section, we have discussed the OOV word detection performance of hybrid systems using different sub-lexical units in the hierarchical hybrid framework and flat hybrid framework, respectively. Here, we compare those two types of hybrid systems. Figure 4.11 presents the OOV word detection performance of hierarchical and flat hybrid systems, where the solid lines are the results of hierarchical systems and the dotted lines are the results of flat systems.

We can find that the flat hybrid systems are significantly better than the hierarchical hybrid systems, except for the phone flat hybrid systems in the WSJ and BN tasks. Across different systems and tasks, the flat hybrid system obtains 10% to 30% absolute improvement on the *Miss Rate* over the hierarchical hybrid system. In the flat hybrid system, the transitional probabilities between words and sub-lexical units capture valuable information about which OOV word appears in what context. While in the hierarchical system, the sub-lexical LM was trained separately, the context dependencies between words and sub-lexical units were therefore lost. Furthermore, OOV words were represented with more complex sub-lexical units in the flat hybrid system than in the hierarchical hybrid system. As a result, flat hybrid systems usually perform better than hierarchical hybrid systems. Since flat hybrid systems are more preferable, in the following experiments, we will focus on the flat hybrid framework.

### OOV word detection using a higher order hybrid model

Bigram hybrid models had been tested for the OOV word detection task in previous paragraphs. Now, we investigate the use of higher order hybrid models for finding OOV words. Specifically, trigram flat hybrid models with different sub-lexical units were trained and compared with bigram flat hybrid models. Again, to find the optimal setting, we varied the number of subword units and the grapheme length when building the trigram subword and grapheme systems. Then, the best parameters were selected based on the evaluation results on the Dev data. The optimal number of subword units and grapheme length used in the bigram and trigram flat hybrid models is presented in Table 4.10. We can find that the number of subword units is smaller than that in the bigram model. This is because the same text training data can only afford fewer tokens when estimating higher order  $n$ -gram language model.

Table 4.10: The optimal number of subword units and grapheme length used in flat hybrid models.

	Number of Subword Units			Grapheme length		
	WSJ	BN	SWB	WSJ	BN	SWB
Bigram Flat System	7000	3000	4500	4	3	4
Trigram Flat Model	5000	3000	3000	4	3	4

The OOV word detection results using bigram and trigram hybrid models are provided in Figure 4.12, where the solid lines correspond to the results of using bigram models and the dotted lines correspond to the results of using trigram models. It can be seen that trigram hybrid models perform much better than bigram hybrid models. Generally, there is a 5% to 15% absolute improvement across different tasks. Compared to bigram models, LM scores are calculated from longer history of words in the trigram hybrid model. Therefore a broader context is utilized for OOV word detection. Similar to the bigram results, among individual hybrid systems, the phone hybrid system is still the worst, while the subword hybrid system performs better than the grapheme hybrid system and the syllable hybrid system is slightly better than the other systems.

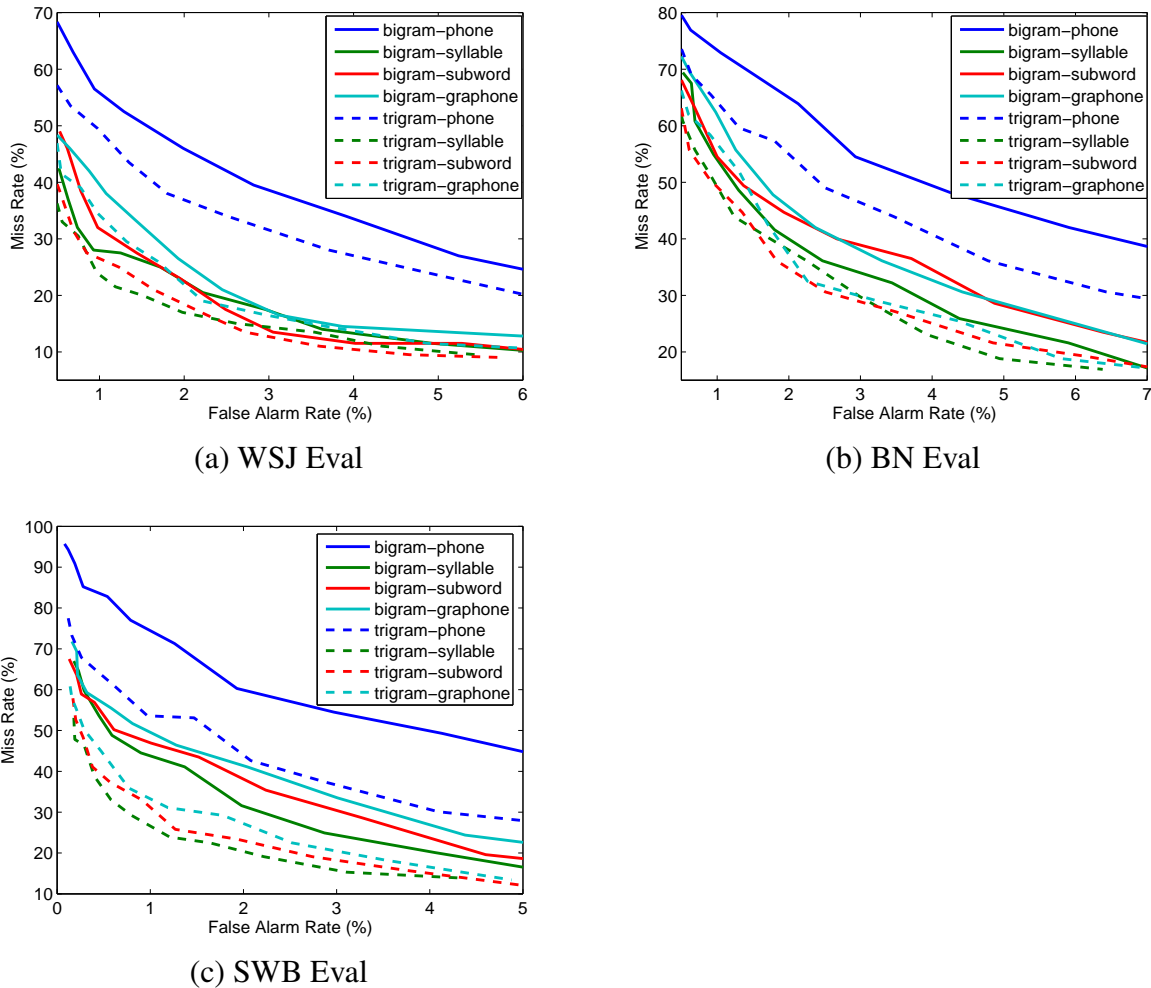


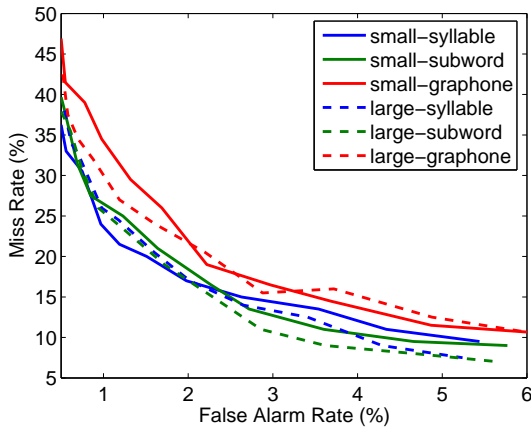
Figure 4.12: The OOV word detection results using bigram and trigram flat hybrid models.

### Utilizing a larger dictionary in a hybrid system

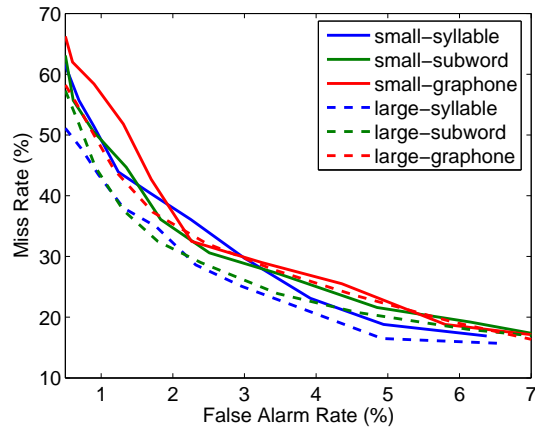
As discussed in previous chapters, it is not good to always apply a very large dictionary in a recognizer. In most cases, a small domain specific dictionary is used. For example, in a dialog system designed for inquiring bus information, a small dictionary including many street and location names is applied. Even for LVCSR applications, some systems can only afford dictionaries with up to 60k words. For OOV word detection, can we obtain any benefit from the large dictionary, if we still apply a small IV dictionary in decoding? The answer is YES. We can utilize the large dictionary in different phases when building a hybrid system. Specifically, when training a flat hybrid LM, we need to estimate the pronunciations of OOV words through the grapheme-to-phoneme (G2P) conversion. The G2P model is normally built from the small IV dictionary. But now, we can estimate a better G2P model using the large dictionary. We can even directly search correct pronunciations for OOV words, if they appear in the large dictionary. Because of that, we now have a better model of OOV words from the more accurate OOV pronunciations. And

Table 4.11: The optimal number of subword units and graphone length used in hybrid systems trained from small and large dictionaries.

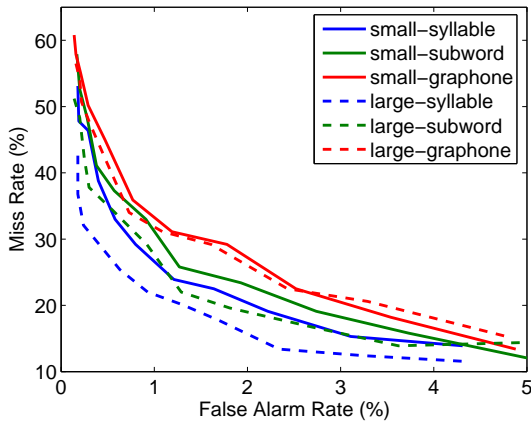
	Number of Subword Units			Graphone length		
	WSJ	BN	SWB	WSJ	BN	SWB
Small Dict	5000	3000	3000	4	3	4
Large Dict	5500	4000	3000	4	4	4



(a) WSJ Eval



(b) BN Eval



(c) SWB Eval

Figure 4.13: The OOV word detection results using hybrid systems trained from small and large dictionaries.

as a result, we are able to train a better flat hybrid model. In this thesis, the CMUdict (v.0.7.a) [Rudnicky, 2007], which is a pronunciation dictionary for North American English containing over 125k words, was introduced as the large dictionary.

We can find the optimal number of subword units and graphone length used in hybrid systems trained from the small IV dictionary and the large CMUdict from Table 4.11. When using the

large CMUdict to train hybrid model, we have more accurate phonetic representation of OOV words. As a result, we can utilize more subword units and longer grapheme units in the hybrid system. The OOV word detection results using hybrid systems trained from small and large dictionaries are given in Figure 4.13, where the solid lines correspond to the results of using hybrid systems trained from the small dictionary and the dotted lines correspond to the results of using hybrid systems trained from the large CMUdict. It can be seen that by using better hybrid models trained from the very large CMUdict, we can further improve the OOV word detection performance by about 5%. To be noticed, the improvement obtained in the grapheme hybrid system is not as large as in the syllable and subword hybrid systems. Because we still used the joint sequence model built from the small IV dictionary to segment OOV words into grapheme units when training the grapheme hybrid system. We tried to build joint sequence models from the large CMUdict for segmentation. However, the training of high order long length joint sequence models require very large amount of computation. Therefore, we only used the not so accurate model built from the small dictionary. We believe the grapheme system could perform better if the grapheme units are produced using a better model trained from the large dictionary. For the best performance, when the *False Alarm Rate* is 1%, the hybrid system can detect more than 75% OOV words in the WSJ and SWB tasks and up to 60% OOV words in the BN task.

### 4.4.3 OOV word classifier results

To evaluate the OOV word classifier performance, we performed classification on OOV hypotheses in the hybrid system’s output. We selected the best hybrid system in each task, which is the subword system in the WSJ and BN tasks and the syllable system in the SWB task. As described in Section 4.1.3, we first trained the LogitBoost classifier using the OOV word detection output of the Dev data. After that, we collected acoustic, lexical, lattice and contextual features from the OOV word detection output of the Eval data and classified each detection hypothesis as a true or false positive. Figure 4.14 presents the OOV word detection results using the hybrid system and OOV word classifier, where the dotted line is the detection result of the hybrid system and the solid line is the result after OOV word classification. First, when  $C_{OOV}$  is small, we cannot get much help from the OOV word classifier. This is because OOV word detection was very accurate when  $C_{OOV}$  was small, there were not many false alarm errors in the detection output. As a result, we cannot observe improvement from the OOV word classifier. Then, when we gradually increase  $C_{OOV}$ , the hybrid system started to report more OOV hypotheses, which produced more false alarm errors. In this case, the OOV word classifier was able to identify more false positive OOV hypotheses, so that the OOV word detection performance was improved. Furthermore, the OOV word classifier did not work well in the BN task. This may be because the BN task is generally more difficult than the WSJ and SWB tasks. The features collected from the BN data were more noisy than features in the WSJ and SWB tasks, which could cause a poor classification performance. Moreover, we trained the OOV word classifier from the hybrid decoding result of the Dev data. But there are only a few hundred OOV words in the Dev data. If we could train the classifier on a larger data set, such as the hybrid decoding result of the training data, the classification performance may be further improved. As the OOV word classifier involved extra computation and the performance was not consistent across different tasks, we didn’t perform OOV word classification in following experiments.

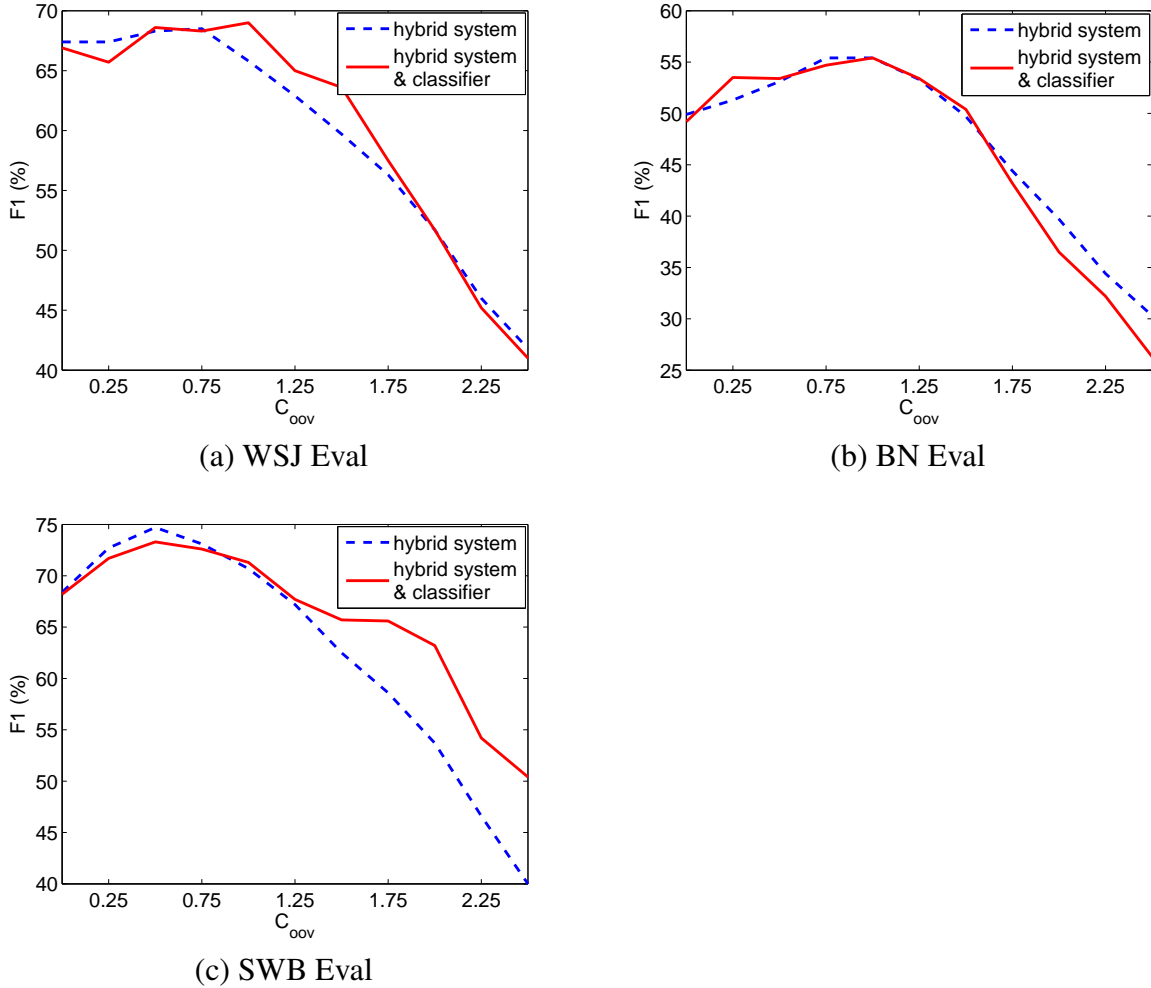


Figure 4.14: The OOV word detection results using the hybrid system and OOV word classifier.

#### 4.4.4 System combination results

In previous sections, we showed that hybrid systems using different sub-lexical units performed differently, do they also complement each other? The distribution of common detection errors among the syllable, subword and grapheme hybrid systems is given in Table 4.12. Here, we averaged the common errors from detection results when using hybrid models with  $C_{OOV} = [0, 2.5]$ . It can be seen that across all tasks, more than 50% errors are unique errors only made by one system, which in theory can be corrected by voting.

Similar to recognition errors, we can also divide detection errors into three categories: substitutions, insertions, and deletions. Substitution corresponds to an OOV word in the reference is detected as IV word in the hypothesis, and vice versa. Insertion corresponds to an extra OOV word is inserted in the hypothesis, while deletion corresponds to an OOV word in the reference is deleted in the hypothesis. From the distribution of different detection errors presented in Table 4.13, we can find that most of the detection errors are substitution errors. There is a very small

number of insertion errors and even less deletion errors.

Table 4.12: The distribution of common detection errors among the syllable, subword, and grapheme hybrid systems.

Errors appear in (%)	WSJ	BN	SWB
1 system	52.6	55.1	52.0
2 system	25.7	24.6	25.0
3 system	21.7	20.3	23.0

Table 4.13: The distribution of different detection errors.

detection errors (%)	WSJ	BN	SWB
substitution	95.0	94.7	86.5
insertion	3.6	3.5	8.1
deletion	1.4	1.8	5.4

### The results of combining multiple hybrid systems' outputs

Now we investigate to combine multiple systems' outputs using ROVER to improve the OOV word detection performance. Two voting modules, voting by word frequency, hereafter referred as "*rover-1*", and voting by word frequency and word confidence, hereafter referred as "*rover-2*", were compared. Since the phone hybrid system performed much worse than the other systems, we only worked with the outputs of the syllable, subword and grapheme hybrid systems. During ROVER, outputs of multiple systems using hybrid models with the same OOV cost were combined. The parameter  $\alpha$  and  $C(@)$  were determined using grid search on development data. And multiple outputs of individual systems were always aligned to the one with the best performance.

In the *rover-1* system, when re-scoring the transition network, only the word frequency was considered. ROVER will identify an OOV word in a region if at least two systems reported OOV words in that alignment. While in the *rover-2* system, to re-score the transition network, both the word frequency and the word confidence score were used. As a result, even two systems had the same vote, the third system could still win if it was more confident. The OOV word detection results of individual and combined systems are given in Figure 4.15. It can be seen that the *rover-1* system does not always outperform all individual systems. In fact, if two systems act similarly, the *rover-1* system usually tends to follow the performance of those two systems. For example, in the SWB task, when the performance of the syllable and subword systems are similar and better than the grapheme system, the *rover-1* system also beats the grapheme system and even slightly better than the syllable and subword systems. But when the subword and grapheme systems perform similarly and worse than the syllable system, the *rover-1* system now gets a poor performance and cannot win the best individual system. On the other hand, the *rover-2* system usually performs better than the *rover-1* system and all individual systems, indicating that the word confidence does help to find the correct hypothesis other than the majority hypothesis in

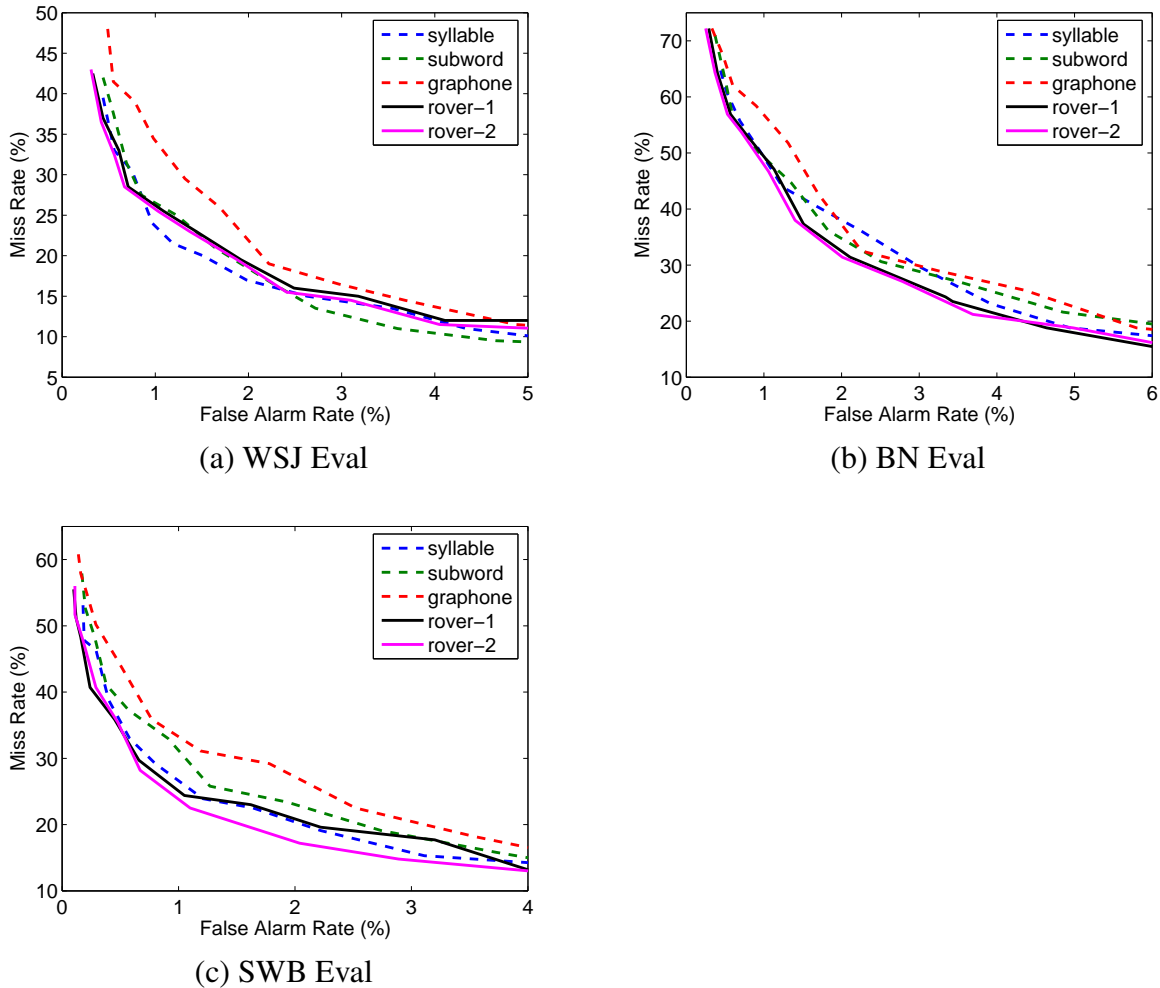


Figure 4.15: The OOV word detection results of individual and combined systems.

system combination. However, in the WSJ task, although the rover-2 system is still better than the rover-1 system, it does not outperform all individual systems. This may be because individual systems are all very confident in this case, so that the word confidence is not as effective as in the BN and SWB tasks to help differentiate the correct hypotheses from the majority ones.

We also noticed that the improvement on OOV word detection is not as large as shown in Table 4.12. One reason may be that we only applied the basic ROVER method to combine the 1-best OOV word detection hypothesis from each hybrid system. We may be able to improve the system combination performance by combining more complex outputs of individual systems, such as the n-best lists or word lattices. Another reason is in ROVER, we aligned individual hypotheses to the one with the best performance. But when finding common errors among individual systems, we used the true reference to align the hypothesis of each system. We can certainly improve the ROVER performance, if we can find better alignments among multiple systems' outputs.



## The results of combining multiple types of sub-lexical units

In our experiments, we tuned  $x$  and  $y$  for the mix-hier method on the Dev data. We changed  $x$  and  $y$  from 10 to 80 with a step size of 10. The best performance was achieved when using  $x = 60, y = 10$  in the WSJ and BN tasks and  $x = 30, y = 10$  in the SWB task. The OOV word detection results of combining multiple types of sub-lexical units are given in Figure 4.16. We can find that the mix-flat system usually performs better than the mix-hier system, which may be a result of better model of OOV words using all three types of sub-lexical units. Furthermore, the mix-flat system outperforms hybrid systems with only one type of sub-lexical units in the WSJ task and performs similarly as the best individual system in the BN and SWB tasks. The mix-hier system might be improved by more carefully tuning the parameters  $x$  and  $y$ . However, the mix-flat system is still preferable, as it does not involve any manual work and development data in training.

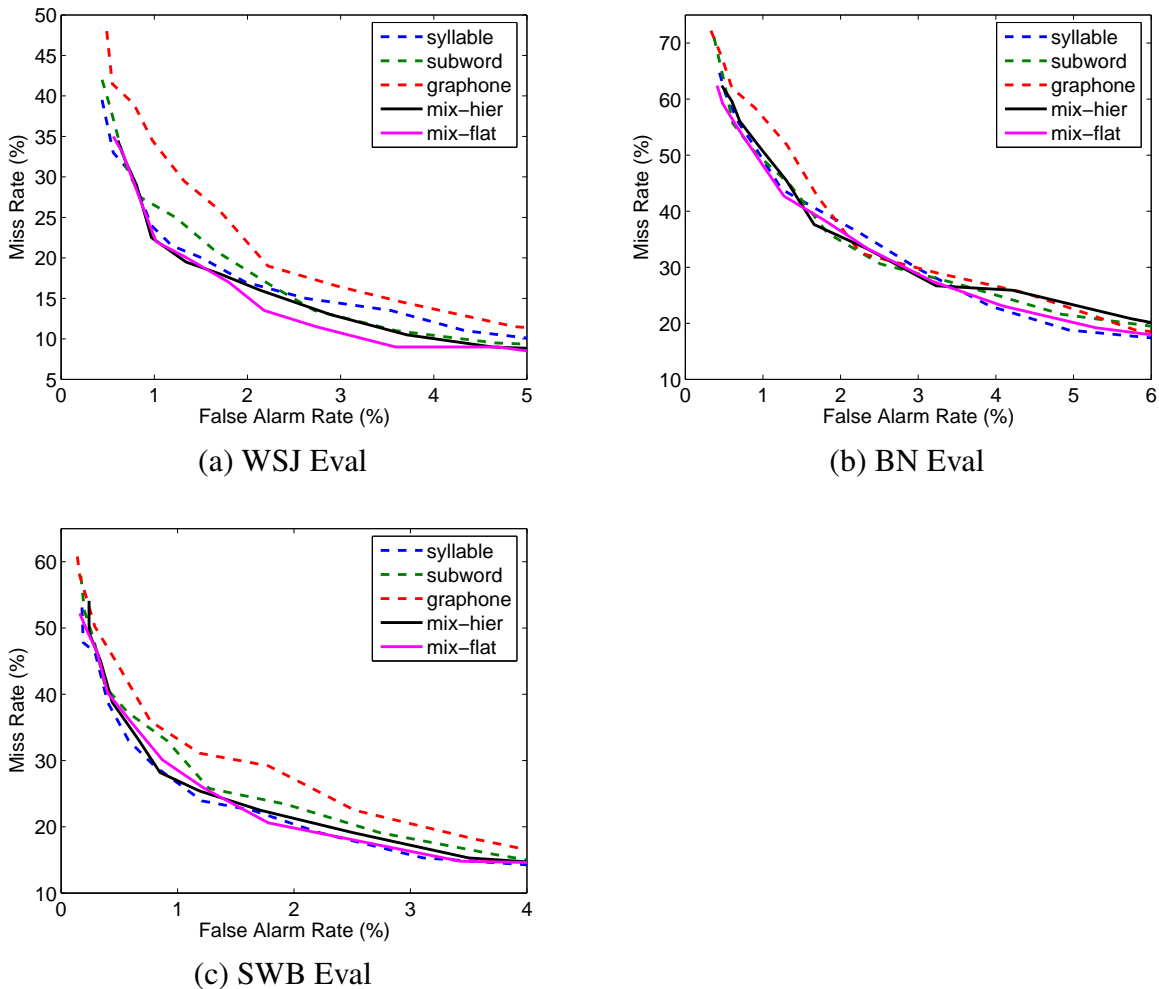


Figure 4.16: The OOV word detection results of combining multiple types of sub-lexical units in one system.

## Comparison of combining multiple systems' outputs and combining multiple types of sub-lexical units

Since we had implemented two different system combination techniques, let us compare the OOV word detection performance of those two systems, which is show in Figure 4.17. We can find that those two system combination techniques perform differently across different tasks. In the WSJ task, combining multiple types of sub-lexical units is better than combining multiple systems' outputs. Furthermore, the first combination system also outputs all individual systems, while the ROVER system does not. In the BN and SWB tasks, however, the ROVER system is better than the mixed units system. And unlike the mixed units system, the ROVER system is better than all individual systems. Therefore, by carefully selecting the proper system combination technique, we can always have a combined system which outputs individual hybrid systems.

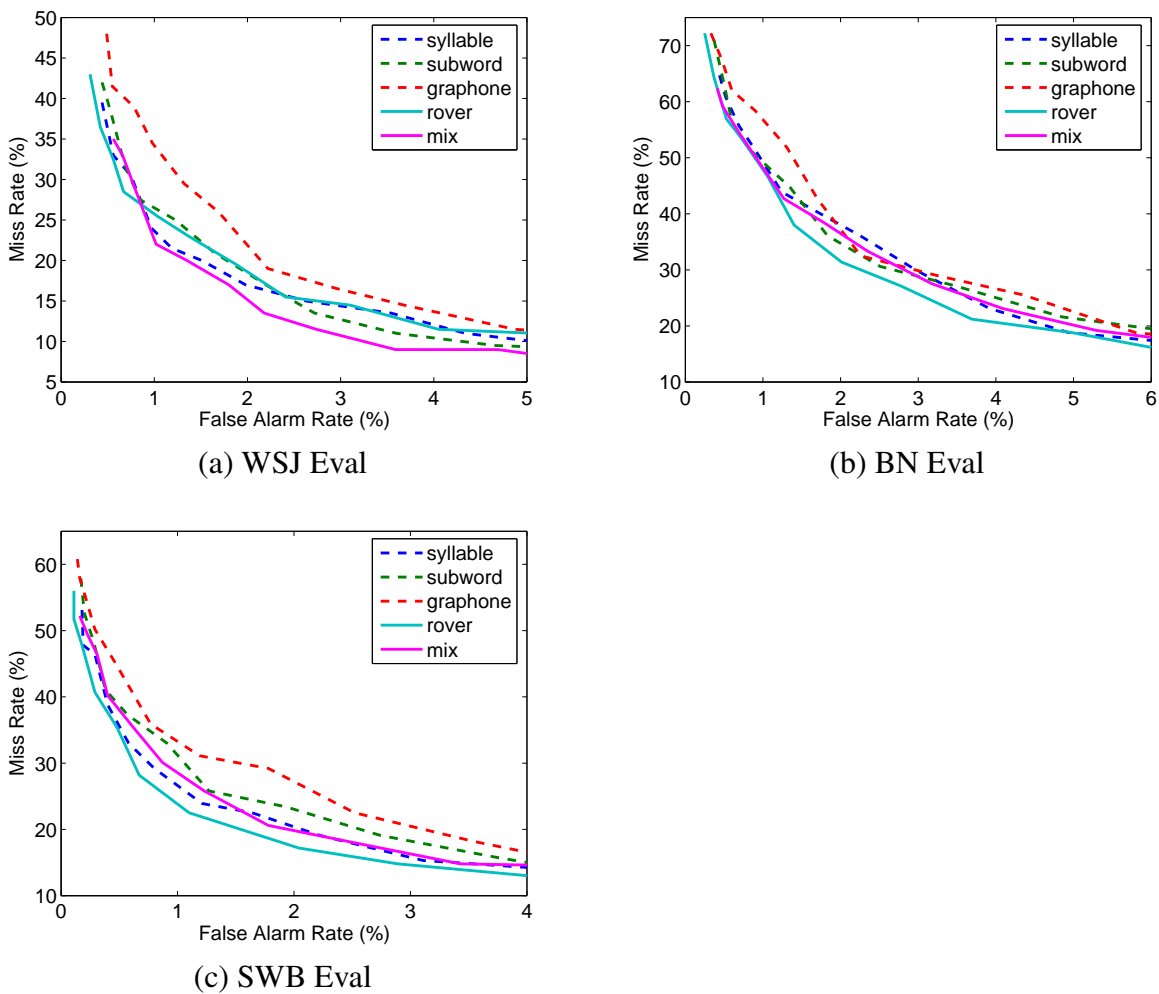


Figure 4.17: The OOV word detection results of combining multiple systems' outputs and combining multiple types of sub-lexical units.

## 4.5 Summary

In this chapter, we investigated various techniques for OOV word detection. First, different hybrid systems, the hierarchical and flat hybrid systems, using different sub-lexical units, the phone, syllable, subword and grapheme units, were built and tested. Then the OOV word classifier and system combination approaches were explored to further improve the OOV word detection performance. From the experimental results, we found that the flat hybrid system is better than the hierarchical hybrid system. We also learned that the syllable, subword and grapheme units model OOV words better than the phone units. Moreover, the effectiveness of the OOV word classifier and system combination techniques is confirmed by our experiments.



# Chapter 5

## Finding Recurrent OOV Words

OOV words can appear more than once in a conversation or over a period of time. Such multiple instances of the same OOV word provide valuable information for estimating the pronunciation, part-of-speech (POS) tag or language model scores of the word. Therefore, in this chapter, we investigated how to identify recurrent OOV words in speech recognition. Specifically, we proposed to cluster multiple instances of the same OOV word using a bottom-up approach. Phonetic, acoustic and contextual features were collected to measure the distance between OOV candidates [Qin & Rudnicky, 2013].

### 5.1 Bottom-up clustering

To find recurrent OOV words, we worked on the hybrid decoding output, from which we collected the phonetic, acoustic and contextual features for each hypothesized OOV candidates. As given in Table 5.1, the phonetic feature is simply the decoded phone sequence of an OOV candidate, the acoustic feature is posterior probability vectors extracted from the OOV region in the testing speech, while the contextual feature is obtained from the words surrounding the OOV candidate. Note that since we collected features from the hybrid system output, recognition errors might be incorporated in these features. For example, in the contextual feature of OOV candidate  $s_1$ , the word “major” is a misrecognition of “mayor”; and the correct pronunciation of OOV candidate  $s_2$  is actually “B AO R AO F”. Depending on the hybrid system performance, the collected features could be very noisy, which thus would cause a poor clustering performance.

Table 5.1: Examples of the phonetic, acoustic and contextual features of an OOV candidate.

OOV	Phonetic	Acoustic	Contextual
$s_1$	S E H L T S	[0.00 ... 0.17]	... major join crowd wall street cut ibm earning estimate ...
$s_2$	M A O R A O F	[0.01 ... 0.24]	... pakistani prime minister die gun battle govern party campaign ...
$s_3$	W A O L I Y	[0.02 ... 0.01]	... racquetball court rule play ball gym schedule ...

As we did not know the correct number of OOV words in the testing speech, and many OOV words only had one or two instances, we could not apply the centroid-based or distribution-based clustering algorithms, such as the k-means algorithm. Therefore we proposed to cluster multiple instances of the same OOV word using a bottom-up approach. Initially, each OOV candidate was considered as a single cluster. Then, in each iteration, two clusters with the smallest distance were merged. This clustering procedure ended when the distance between clusters was larger than a threshold. In this thesis, the distance between two clusters was defined as the average of pairwise distances between OOV candidates in two clusters. Formally, the distance between cluster  $C_m$  and  $C_n$  is

$$D(C_m, C_n) = \frac{1}{|C_m||C_n|} \sum_{s \in C_m} \sum_{s' \in C_n} d(s, s'), \quad (5.1)$$

where  $|C_m|$  and  $|C_n|$  are the number of OOV candidates in cluster  $C_m$  and  $C_n$ , and  $d(s, s')$  is the distance between two OOV candidates

$$d(s, s') = \omega_P d_P(s, s') + \omega_A d_A(s, s') + \omega_C d_C(s, s'). \quad (5.2)$$

Here,  $d_P(s, s')$ ,  $d_A(s, s')$  and  $d_C(s, s')$  are the phonetic, acoustic and contextual distances between OOV candidate  $s$  and  $s'$ , while  $\omega_P$ ,  $\omega_A$ ,  $\omega_C$  are their weights respectively. In addition to averaging the pairwise distances between OOV candidates, we also experimented with calculating  $D(C_m, C_n)$  as the maximum or minimum distance between OOV candidates in two clusters. However, we found that the clustering performance with different definitions of  $D(C_m, C_n)$  was essentially the same, although the average one occasionally performed better.

Actually, we could applied the semi-supervised clustering techniques, such as the similarity-adapting methods, where we can learn a better similarity metric from the development or training data rather than linearly combining different features. Given the true labels of detected OOV candidates in the development or training data, we can train classifiers using the distances between the phonetic, acoustic and contextual features of two OOV candidates as input while the label of whether those two OOV candidates are from the same OOV word as output. Then during testing, we can use those classifiers to measure the similarity between OOV candidates and classify whether they correspond to the same OOV word. But to simplify our implementation, we chose to perform unsupervised clustering, although those semi-supervised clustering techniques are probably more accurate.

## 5.2 Measuring the distance between OOV candidates

To measure the distance between OOV candidates during clustering, we calculated the distance between the phonetic, acoustic and contextual features of two OOV candidates. While the overall distance between OOV candidates is a linear combination of individual distances.

### 5.2.1 Phonetic distance

The most direct way to determine whether two OOV candidates may correspond to the same OOV word is to examine whether they have the same pronunciation. To do that, we measured the

phonetic similarity between OOV candidates by computing the distance between their decoded phone sequences. Specifically, the phonetic distance  $d_P(s, s')$  between OOV candidate  $s$  and  $s'$  was formulated as the normalized edit distance between their phone sequence  $p_s$  and  $p_{s'}$ :

$$d_P(s, s') = \frac{\text{edit}(p_s, p_{s'})}{|p_s| + |p_{s'}|} \quad (5.3)$$

where  $|p_s|$  and  $|p_{s'}|$  are the lengths of phone sequence  $p_s$  and  $p_{s'}$ . As shown previously in Table 5.1, the decoded phone sequences of OOV candidates may incorporate recognition errors. Particularly, similar phones, such as “AA” and “AO”, are more often to be mis-recognized than the other phones. Therefore, we adopted a modified edit distance that compensates for the acoustic confusability between phones [Wagner & Fischer, 1974; Audhkhasi & Verma, 2007; Pucher et al., 2007; Printz & Olsen, 2002],

$$\begin{aligned} \text{edit}(0, 0) &= 0 \\ \text{edit}(i, 0) &= i \\ \text{edit}(0, j) &= j \\ \text{edit}(i, j) &= \min \begin{cases} \text{edit}(i-1, j) + 1 \\ \text{edit}(i, j-1) + 1 \\ \text{edit}(i-1, j-1) + c(i, j). \end{cases} \end{aligned} \quad (5.4)$$

In Eq. 5.4,  $c(i, j)$  is the confusability between phone  $i$  and  $j$

$$c(i, j) = \begin{cases} 0 & \text{if } i = j \\ 1 - p(i, j) & \text{if } i \neq j, \end{cases} \quad (5.5)$$

where  $p(i, j)$  is the probability of misrecognizing phone  $i$  and phone  $j$ . We estimated  $p(i, j)$  from the recognition result of the Dev speech

$$p(i, j) = p(j, i) = \frac{N(i \rightarrow j) + N(j \rightarrow i)}{N(i) + N(j)}, \quad (5.6)$$

where  $N(i)$  and  $N(j)$  are the number of phone  $i$  and phone  $j$  in the recognition result, while  $N(i \rightarrow j)$  and  $N(j \rightarrow i)$  are the number of times phone  $i$  was recognized as phone  $j$  and vice versa. In this thesis,  $p(i, j)$  was estimated from the hybrid decoding results of the Dev data. Basically, we transformed words in the hybrid decoding result and in the reference into phone sequences, aligned those two sequences of phones, then counted how many phones were misrecognized.

## 5.2.2 Acoustic distance

Besides measuring the phonetic distance between OOV candidates, we can also compare their acoustic features extracted from the OOV region in the testing speech. Acoustic features, such as the mel-scale frequency cepstral coefficients (MFCCs), are highly sensitive to speaker and channel variations. On the other hand, posterior-based features, such as the phonetic posteriorgram,

are more robust and also widely used in speech recognition [Aradilla et al., 2006; Hazen et al., 2009; Zhang & Glass, 2009]. We therefore used the posterior feature to model OOV candidates in our system. Precisely, each frame  $f_t$  in the OOV region was represented by a probability vector

$$v_t = [P(p_1|f_t), P(p_2|f_t), \dots, P(p_K|f_t)], \quad (5.7)$$

where  $P(p_k|f_t)$  is the posterior probability of  $f_t$  belonging to phone  $p_k$  and  $K$  is the total number of phones. To estimate  $P(p_k|f_t)$ , we trained a Gaussian mixture model (GMM) with 256 Gaussian components for each phone. Then the posterior probability  $P(p_k|f_t)$  can be calculated as

$$P(p_k|f_t) = \frac{P(f_t|p_k)}{\sum_{k \in K} P(f_t|p_k)}, \quad (5.8)$$

where  $P(f_t|p_k)$  is the likelihood of observing  $f_t$  from the GMM of  $p_k$ . In our experiments, we found that the posterior probability mass was usually absorbed by only a few GMMs, most phones had a posterior probability close to zero. We ended up with a very sparse posterior probability vector  $v_t$ . To solve this problem, we performed a discounting-based smoothing on vector  $v_t$  in a way similar to [Zhang & Glass, 2009]. Specifically, each zero element in  $v_t$  was assigned a small posterior probability  $\lambda$ , and each non-zero element was discounted by  $(1 - N\lambda)$ , where  $N$  is the total number of zero elements in  $v_t$ .

After constructing the posterior features, we calculated the acoustic distance between OOV candidates using the dynamic time warping (DTW) algorithm [Vintsyuk, 1968; Sakoe & Chiba, 1978],

$$d_A(s, s') = DTW(s, s'). \quad (5.9)$$

In DTW, the distance between two posterior vectors  $v_i$  and  $v_j$  was defined as the negative log cosine similarity between  $v_i$  and  $v_j$

$$d(v_i, v_j) = -\log\left(\frac{v_i \cdot v_j}{\|v_i\| \|v_j\|}\right). \quad (5.10)$$

Moreover, similar to the phonetic distance, we also normalized the acoustic distance by the lengths of OOV regions.

### 5.2.3 Contextual distance

OOV words are usually content words such as names or locations and the same OOV word may appear in similar contexts or environments. If two OOV candidates are surrounded by the same words or used in the same topic, they may actually be the same OOV word. As presented in Eq. 5.2, besides the phonetic and acoustic distances, we also measured the contextual distance between OOV candidates during clustering. To take the position of surrounding words into account, the contextual distance has two elements:

$$d_C(s, s') = \omega^l d_C^l(s, s') + \omega^g d_C^g(s, s'). \quad (5.11)$$

Here,  $d_C^l(s, s')$  is a local contextual distance that measures the similarity between the adjacent words of OOV candidates, which works like an  $n$ -gram language model. And  $d_C^g(s, s')$  is a global contextual distance, which resembles a topic model.



Table 5.2: Examples of the local and global contextual features of OOV candidates.

OOV	$s_1$	$s_2$
Text	i am going to watch tonight because $s_1$ ryan is going to pitch	i love $s_2$ ryan i alway like to watch him pitch
Local context	tonight because $s_1$ ryan is	i love $s_2$ ryan i
Global context	watch:0.33 pitch:0.33 ryan:0.33	watch:0.25 pitch:0.25 ryan:0.25 love:0.25

To calculate the local contextual distance, we compared the left two and right two words of OOV candidates

$$d_C^l(s, s') = 1 - \frac{M}{4}, \quad (5.12)$$

where  $M$  is the number of matched words. For instance, as shown in Table 5.2, there is only one match, “*ryan*”, between the local context of OOV candidate  $s_1$  and  $s_2$ , hence the local contextual distance  $d_C^l(s_1, s_2)$  equals to 0.75.

The global contextual distance was calculated in the same manner as measuring the similarity between two documents in information retrieval. However here, we focused on words in the same sentence and we only used content words. Particularly, for an OOV candidate  $s$ , its global context was represented by a term frequency vector  $c_g$  which was built from the content words of the sentence containing  $s$ . Then the global contextual distance between OOV candidate  $s$  and  $s'$  was calculated as

$$d_C^g(s, s') = -\log\left(\frac{c_g \cdot c'_g}{\|c_g\| \|c'_g\|}\right), \quad (5.13)$$

which is the negative log cosine similarity between the global context of  $s$  and  $s'$ . Examples of the global context are also provided in Table 5.2.

## 5.3 Experiment setup

### 5.3.1 The hybrid system and dataset

As we mentioned previously, the quality of the hybrid system’s output will affect the quality of features we extract for each OOV candidate, which will then affect the OOV word clustering performance. Therefore, we tried to use the output from hybrid systems with the best OOV word detection performance for our clustering experiments. It is shown in Figure 4.13 that hybrid systems built with the large CMUDict performed better than systems built with the small in-vocabulary (IV) dictionary. We selected outputs with the highest F1 score from hybrid systems built with the large CMUDict. Specifically, the subword system with  $C_{OOV} = 0.75$  was selected in the WSJ task, the subword system with  $C_{OOV} = 1.0$  was selected in the BN task, and the syllable system with  $C_{OOV} = 1.0$  was used in the SWB task. The OOV word detection performance on the Eval data of selected hybrid systems is given in Table 5.3. We can find that the hybrid system performs very well in the WSJ and SWB tasks. But the performance in the BN task was slightly worse. This may because in the BN task, utterances are usually much longer

and multiple OOV words can appear in one utterance or even in a sequence, which make OOV word detection more difficult than in the WSJ and SWB tasks. Furthermore, although we could segment a sequence of sub-lexical units into multiple OOV candidates using word start “^” and word end “\$”, it was not guaranteed that they were always applied during the hybrid decoding. As a result, we manually checked the hybrid system’s output to make sure that sub-lexical units were properly segmented. We also manually extracted content words from the hybrid decoding result of each utterance. Finally, the number of instances an OOV word has is given in Table 5.4. It can be seen that in the hybrid system’s output, about 70% OOV words appear only once, less than 10% OOV words have more than two instances. On average, one OOV word has 1.2 instances.

Table 5.3: The OOV word detection performance on the Eval data of selected hybrid systems.

Task	WSJ	BN	SWB
Precision	63.8%	49.8%	67.2%
Recall	74.0%	62.4%	74.6%
F1	68.5%	55.4%	70.7%

Table 5.4: The number of instances an OOV word has in the hybrid system’s output.

OOV word has	WSJ	BN	SWB
1 instance	70.8%	68.8%	77.5%
2 instances	24.0%	19.5%	16.5%
≥ 3 instances	5.2%	11.7%	6.0%

### 5.3.2 Evaluation metrics

The Rand index (RI) is a common evaluation metric for clustering [Rand, 1971]. It involves counting pairs of items on which the reference and hypothesis clusterings agree or disagree. In practice however, RI does not take on a constant value for random clustering. Especially, when the number of classes is large and the number of candidates is small, a random clustering result can have a very good RI score. Contrarily, the adjusted Rand index (ARI) is another widely used clustering evaluation metric [Hubert & Arabie, 1985], which adjusts for the chance of a clustering result. The ARI score is bounded between -1 to 1. Independent clusterings has a negative ARI score, similar clusterings has a positive ARI score and an ARI score of 1 indicates a perfect match between the reference and hypothesis clusterings. As shown in Table 5.4, in our experiment, the majority of clusters only contain one candidate and the candidate to cluster ratio is as low as 1.2. If without clustering, but simply considering each candidate as one OOV word, the RI score will be almost 1, but the ARI score will be a small value close to 0. For this reason, we selected ARI to evaluate our OOV word clustering result.

The adjusted mutual information (AMI) [Vinh & Epps, 2010], which measures the mutual information between the reference and hypothesis clusterings, is another evaluation metric used

in this thesis. Similar to ARI, AMI is also normalized against chance. The AMI score is bounded between 0 to 1. An AMI score of 0 indicates purely independent clusterings, while AMI of exactly 1 means the reference and hypothesis clusterings are equal.

## 5.4 Experimental results

### 5.4.1 The intra-cluster and inter-cluster distances

Before discussing the clustering performance, we first take a closer look at the testing data. Fig. 5.1 shows the comparison of the average distance between candidates from the same OOV word (intra-cluster) with the average distance between candidates from different OOV words (inter-cluster). It can be seen that for the phonetic, acoustic and contextual features, the intra-cluster distance is always smaller than the inter-cluster distance. Moreover, the difference between the phonetic intra-cluster and inter-cluster distances is normally greater than that of the other features. Furthermore, the OOV candidates in the WSJ and SWB tasks seem to be more separable than those in the BN task, as the differences between the intra-cluster and inter-cluster distances are greater in the WSJ and SWB tasks than those in the BN task.

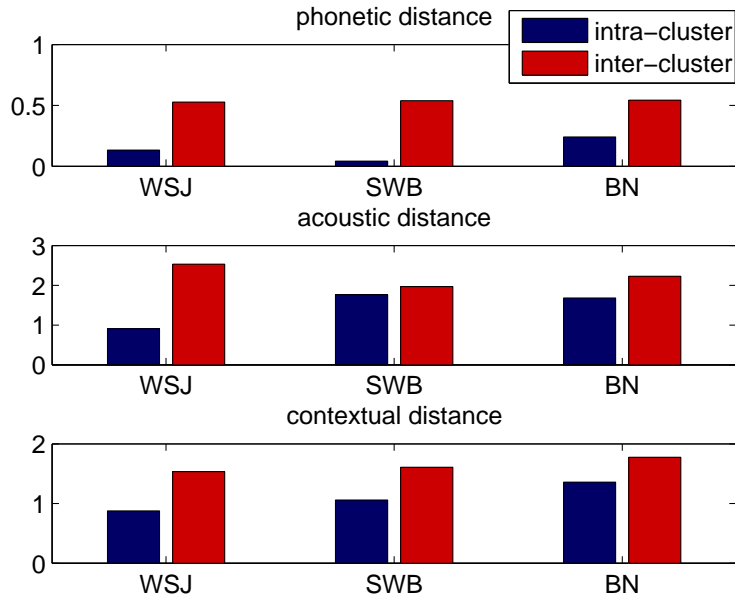


Figure 5.1: Comparison of the average distance between candidates from the same OOV word (intra-cluster) with the average distance between candidates from different OOV words (inter-cluster).

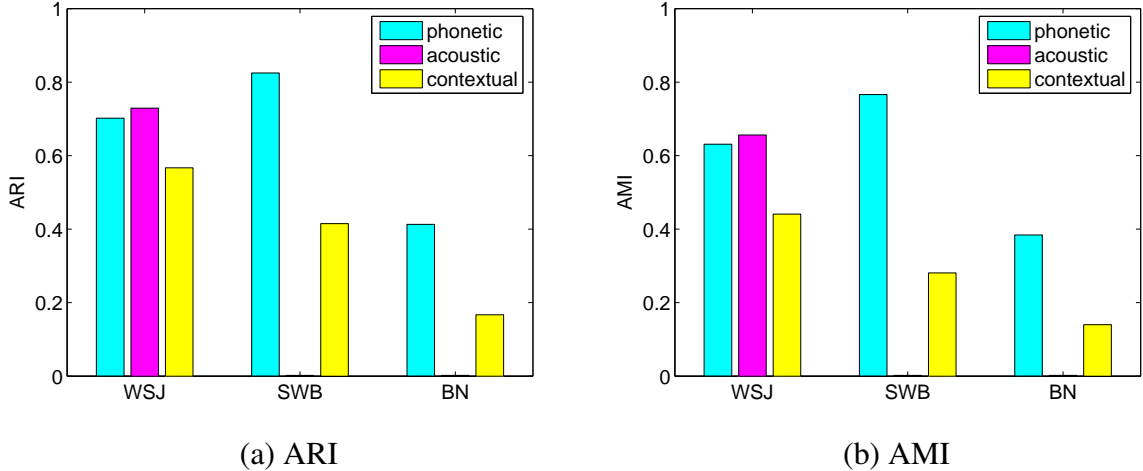


Figure 5.2: The performance of bottom-up clustering using one feature.

## 5.4.2 The bottom-up clustering results

The performance of bottom-up clustering using one feature is given in Fig. 5.2. We can find that the phonetic feature is very effective in all tasks. The acoustic feature works well in the WSJ task but shows the same score as random clustering in the SWB and BN tasks. This may be because that measuring the distance between acoustic signals in the spontaneous or noisy speech is less reliable than in clean speech. Moreover, in the WSJ task, one speaker usually recorded up to 50 utterances. As a result, multiple instances of the same OOV word may actually be spoken by the same speaker, which makes it easier to compare OOV candidates using acoustic features. Although, the contextual feature is not as good as the phonetic one, it does produce positive results across different tasks. By comparing Fig. 5.2 with Fig. 5.1, we can also learn that the clustering performance highly correlates with the difference between the intra-cluster and inter-cluster distances of one feature. Basically, the greater the difference, the better the clustering performance. For instance, the difference between the phonetic intra-cluster and inter-cluster distances is great in all tasks, and the clustering performance using the phonetic feature is always good. On the other hand, the difference between the acoustic intra-cluster and inter-cluster distances is only noticeable in the WSJ task, and clustering using the acoustic feature performs badly in the SWB and BN tasks. The best performance is obtained when using the acoustic feature in the WSJ task and using the phonetic feature in the SWB and BN tasks.

In addition to using only one feature to measure the distance between OOV candidates during clustering, we also applied the combined feature defined in Eq. 5.2. Fig. 5.3 shows the performance of bottom-up clustering using the combined feature, in which the red bar is the best clustering performance using one feature, the green bar is the performance when using both the phonetic and acoustic features, and the blue bar is the performance when combining all features. It can be seen that the clustering performance is better when using the combined feature than using individual feature to measure the distance between OOV candidates. Even for the SWB and BN tasks, where the acoustic feature does not work at all, combining the phonetic and acoustic

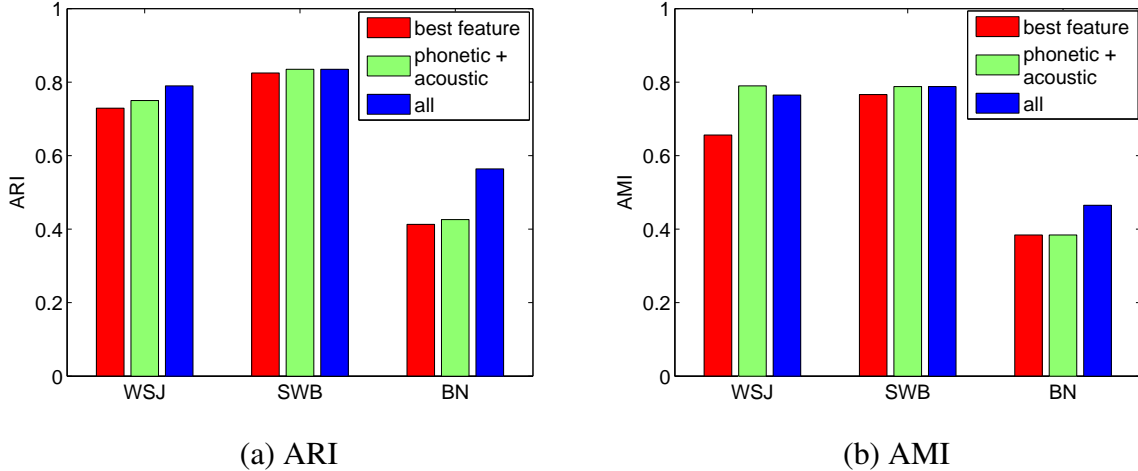


Figure 5.3: The performance of bottom-up clustering using the combined feature.

features can still yield some improvement. And the best performance is usually achieved when combining all features. Overall, the ARI and AMI score is up to 0.8 in the WSJ and SWB tasks and up to 0.6 in the BN task, which indicates that we can successfully find most of the recurrent OOV words using the proposed bottom-up clustering approach. Again, the clustering performance on the BN task is worse than that on the WSJ and SWB tasks. This is because there are many short IV words which were incorrectly recognized as OOV candidates in the BN task. And those short IV words are very hard to cluster compared to longer OOV words.

The goal of finding recurrent OOV word is to utilize its multiple instances, so that we can build a better phonetic representation or estimate better language model scores for the word. During clustering, we prefer not having different OOV words in the same cluster than trying to find all instances of one word. For example, there are four instances of the OOV word “CIBA” in the Eval data. Then we prefer finding two or three instances of “CIBA” than having all four candidates with some other OOV words in the same cluster. The details of our OOV word clustering result are given in Table 5.5. We can find that in the WSJ task, only very few clusters containing OOV candidates from different OOV words, but we did miss some candidates which should be merged with other candidates. In the SWB task, we successfully found most recurrent OOV words, but there were more clusters with irrelevant OOV candidates. In the BN task, we failed to find a large number of recurrent OOV words. Figure 5.4 provides the ARI and AMI scores calculated from all clusters and only from clusters with more than one candidate. We can find that the ARI score is much higher, up to 0.9, when calculated only from clusters with more than one candidate, indicating that most clusters only contain instances from the same OOV word. As a result, we should be able to utilize identified recurrent OOV words to improve the OOV word recovery performance, which will be covered in the following chapter.

Table 5.5: Details of the OOV word clustering result.

	WSJ	SWB	BN
Clusters with more than one candidate	13.9%	13.1%	10.3%
Clusters with candidates from different OOV words	0.5%	3.0%	1.6%
Clusters should be merged with other clusters	5.4%	2.5%	14.4%

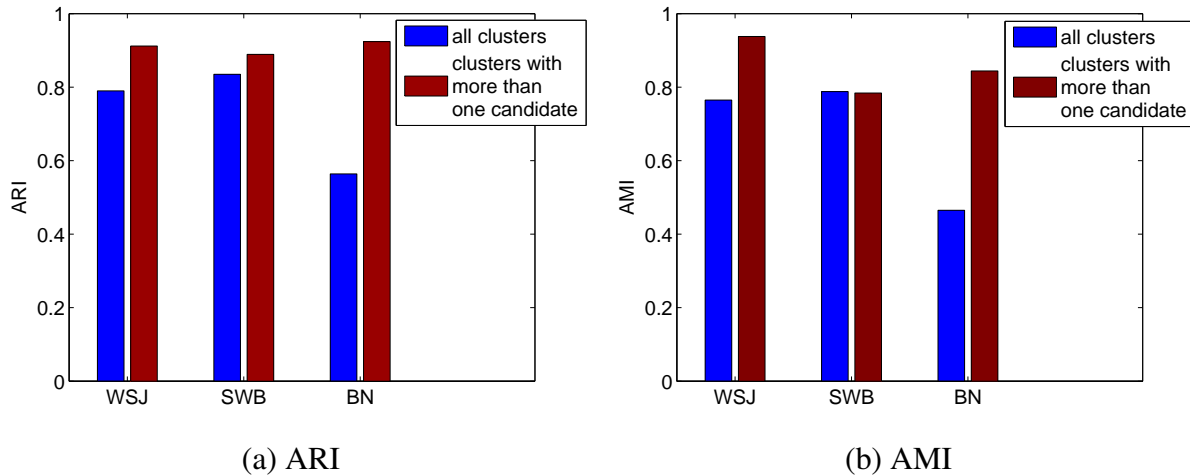


Figure 5.4: The ARI and AMI scores calculated from all clusters and only from clusters with more than one candidate.

## 5.5 Summary

In this chapter, we studied a bottom-up clustering approach to find recurrent OOV words in speech recognition. We collected the phonetic, acoustic and contextual features to measure the distance between OOV candidates. From our experimental results, we found that the phonetic feature is more effective than the acoustic and contextual features for detecting the recurrence of OOV words, but the best performance is usually achieved when combining all features. We also demonstrated the clustering result is good enough for further process. In next chapter, we will investigate how to utilize those multiple instances of the same OOV word for estimating the written form and language model scores of the word.

# Chapter 6

## OOV Word Recovery

In this chapter, we discuss the OOV word recovery problem – recovering the written form and language model scores of an OOV word so as to convert OOV words into IV words. First, we describe our work on estimating the written form of an OOV word through the phoneme-to-grapheme (P2G) conversion. We also study how to estimate language model scores for an OOV word using its syntactic property. In the second part, we investigate another recovery method when extra resources are available. After that, we discuss how to utilize the multiple instances of the same OOV word to improve the OOV word recovery performance. At last, experimental results are given.

### 6.1 The baseline OOV word recovery approach

To learn OOV words or to convert OOV words into IV words, we need to incorporate detected OOV words into the recognizer’s lexicon and language model, thus we need to estimate the written form and language model scores for an OOV word. In this section, we first describe how to estimate the written form of an OOV word through the P2G conversion. We then present how to estimate language model scores by utilizing the part-of-speech (POS) tags of the an OOV word and its surrounding IV words.

#### 6.1.1 Estimating the written form of an OOV word

As presented in Chapter 4, after detecting OOV words in the testing speech using the hybrid system, we can obtain a phonetic representation of an OOV word from the decoded sub-lexical units. We then estimated the written form of an OOV word from its phonetic representation – pronunciation using the P2G conversion.

##### **P2G conversion**

Phoneme-to-grapheme conversion (P2G) refers to the task of finding the written form of a word given its pronunciation. It is related to another task, the grapheme-to-phoneme (G2P) conversion, which performs the opposite conversion – estimating the pronunciation of a word from its written

form. One simple P2G or G2P conversion method is the rule-based conversion system, in which carefully designed matching rules are applied to convert phones to letters or vice versa. But, it is very hard and time consuming to design complete rules to cover both the general and atypical conversions of all words in a language. Therefore, data-driven approaches are more popular in recent years.

There are generally two types of data-driven P2G conversion techniques, the local classification approach and the probabilistic global optimization approach. In the local classification approach, the input string sequence is normally processed individually from left to right. For each input phoneme, one or more letters are predicted using neural networks or decision trees. As the selection of output letters is only based on the context of the current phoneme, this approach is not optimal. But it avoids the need of a search algorithm to find the global optimal solution. On the other hand, in the probabilistic approach, dynamic programming is usually applied to find the global optimal prediction. One particular algorithm in this category was used in this thesis, which was the joint-sequence model [Bisani & Ney, 2008]. The idea of the joint-sequence model is that the relation between the orthographic form and the pronunciation of a word can be generated by a common sequence of graphemes, which are grapheme-phoneme pairs of English letters and phones. For example, the word “speech” can be represented as a sequence of three graphemes

$$\text{speech} = \begin{pmatrix} s \\ S \end{pmatrix} \begin{pmatrix} pee \\ P IY \end{pmatrix} \begin{pmatrix} ch \\ CH \end{pmatrix}.$$

Then the P2G or G2P conversion task can be solved by searching for the optimal representation of a word using graphemes given its pronunciation or its written form.

Each grapheme can model a minimum and maximum number of letters and phones. Meanwhile, a joint-sequence model can be built on different lengths of grapheme sequences. Therefore, to achieve the best P2G conversion performance, we trained a 6-gram joint-sequence model with short grapheme units as suggested in [Chen, 2003]. The word start and end symbols, “^” and “\$”, were used to segment the decoded sub-lexical sequences into multiple OOV words. While for the hybrid system built from graphemes, we did not need to perform the P2G conversion to restore the OOV word spellings, instead we could simply concatenate the letters from the decoded grapheme sequence.

### **Training a better joint-sequence model**

In our experiment, we found only a small portion of the detected OOV words had the correct pronunciation and even fewer OOV words’ written forms were successfully estimated. Since we trained the joint-sequence model from IV words whose pronunciations are always correct, the P2G conversion is not reliable when the input phonemes are mixed with recognition errors. Figure 6.1 provides an examples of the P2G conversion from the decoded OOV word’s pronunciation to the OOV word’s written form. We can find that because the OOV word was incorrectly recognized as “K AE D IY” instead of the correct pronunciation “K AE D R IY”, the predicted spelling “CADY” was also wrong. Therefore, we proposed to train a better joint-sequence model, so that we were able to correctly estimate the written form of an OOV word, even if its phonetic representation incorporated recognition errors. For instance, the P2G conversion using a better



joint-sequence model would be able to correctly predict the written form “CADRE” from the incorrect pronunciation “K AE D IY”.

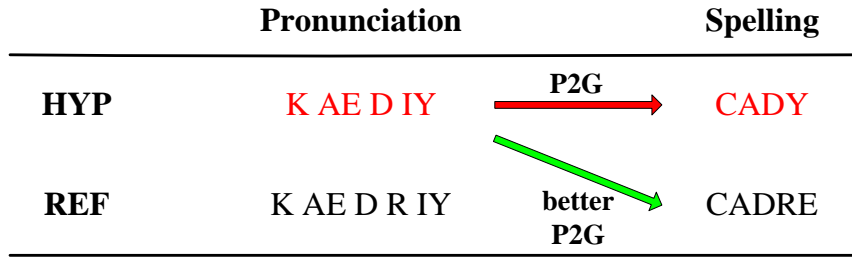


Figure 6.1: An example of the P2G conversion from an incorrect OOV word pronunciation.

We began with collecting detected OOV words from the hybrid decoding results of the training speech. In the hybrid decoding result, there were detected OOV words with correct pronunciations, from which we could extract alignments between the correct letter sequences of reference OOV words and the correct phone sequences of detected OOV words. Furthermore, because the decoder could make the same errors on the training data as on the Eval data, we could also find alignments between the correct spellings of reference OOV words and incorrect pronunciations of detected OOV words. Then we trained a joint-sequence model using the detected OOV words together with the IV words. This joint-sequence model measured both the regular matchings between the correct phone and letter sequences. It also learned the alignments between the correct spellings and incorrect pronunciations. As a result, it was capable of recovering more OOV words from the noisy hybrid decoding output. During our experiment, we found sometimes a long IV word could be recognized as a sequence of short IV words or a number of short OOV words. Then when we aligned the reference with the decoding hypothesis, many reported OOV words could be aligned to the wrong reference words. It was sub-optimal to train the joint-sequence model from those data. Instead, we only used positive OOV detections in the training speech as the training data for the joint-sequence model, which guaranteed us to obtain the correct alignments between the reference OOV words and their decoded pronunciations.

### 6.1.2 Estimating the language model scores of an OOV word

It is very difficult to estimate language model scores of an OOV word, as there is no training data available. Normally, we do not have any text data but the sentence where the OOV word is detected. Therefore, we cannot directly estimate language model scores of an OOV word using conventional techniques, but to find some alternative ways. In this thesis, we investigated to estimate the language model scores of an OOV word using the POS tags of the word and its surrounding IV words.

From the hybrid decoding output, we could estimate the POS label of an OOV word using the Stanford MaxEnt POS tagger [Toutanova & Manning, 2000; Toutanova et al., 2003]. Figure 6.2 provides an example of the estimated POS labels of the OOV word “AIRCOA” and its surrounding IV words. With the POS label, the language model scores of an OOV word could be estimated from IV words in the same syntactic category - words with the same POS label. Specifically, we first trained a POS class-based language model from the training text data. Then, for


<b>Detection HYP</b>	Partner	Of	<i>AIRCOA</i>	Hotel	Partners
					
<b>POS</b>	NN	IN	NNP	NN	NNS

Figure 6.2: An example of the POS labels of decoded OOV and IV words.

a detected OOV word, language model scores were estimated using the POS labels of the word and its surrounding IV words. For example, the unigram score of OOV word “AIRCOA” could be estimated as

$$P(W_{AIRCOA}) = P(W_{AIRCOA}|C_{NNP})P(C_{NNP}), \quad (6.1)$$

where  $P(C_{NNP})$  is the unigram probability of the POS class “NNP” and  $P(W_{AIRCOA}|C_{NNP})$  is the probability of observing “AIRCOA” in class “NNP”. Here, we used a simple uniform distribution to estimate  $P(W|C)$ ,

$$P(W|C) = \frac{1}{N}, \quad (6.2)$$

where  $N$  is the number of words in class  $C$ . Similarly, we could estimate the bigram and trigram language model scores for “AIRCOA” as

$$P(W_{AIRCOA}|W_{OF}) = P(W_{AIRCOA}|C_{NNP})P(C_{NNP}|C_{IN}), \quad (6.3)$$

and

$$P(W_{AIRCOA}|W_{PARTNER}, W_{OF}) = P(W_{AIRCOA}|C_{NNP})P(C_{NNP}|C_{NN}, C_{IN}). \quad (6.4)$$

For one detected OOV word, we could add one unigram and up to two bigram and three trigram scores of the word. However, as our ASR system applied a back-off language model during decoding, it required that trigrams only contained bigrams that already existed in the language model. For instance, if the bigram “HOTEL PARTNERS” was not in the language model, we could not add trigram “AIRCOA HOTEL PARTNERS” into the language model.

In our experiment, we found  $P(W|C)$  could be very small, especially for the “NNP” class, which could contain more than 10k words. As a result, the estimated bigram and trigram scores of an OOV word were usually much smaller than those of existing IV words. Therefore, we set a threshold on  $P(W|C)$ , so as to obtain proper bigram and trigram scores for an OOV word. In this thesis, we used all labels from the Penn Treebank POS tag set [Marcus et al., 1993]. Furthermore, in our ASR system, words like “I’VE” or “TEAM’S” were processed as a single unit. However, in the POS tagger, those words were predicted with separate labels. For example, the POS tagger output of “TEAM’S” was “TEAM\_NN” and “’S\_POS”. To solve this problem, we combined the separate labels of a word to form a compound label, such as “TEAM’S\_NN+POS”. Therefore, besides the 35 base labels, there were also many compound POS labels in our system.

An OOV word may come up in a totally different context in the future, therefore we also need to estimate possible context an OOV word may appear and their corresponding language model scores. To do that, we still utilized the surrounding IV words of an OOV word. For each IV word in the left and right two words of an OOV word, we substituted it with other semantic similar IV words, and then augmented the language model with those new  $n$ -grams. For instance, we found the IV word “HOTEL” is very similar to another IV word “INN”. Therefore, we augmented the language model with new bigram “AIRCOA INN”, new trigram “OF AIRCOA INN”, “AIRCOA INN PARTNERS”, etc. The semantic features of each IV word were obtained from the WordNet corpus [Miller, 1995; Fellbaum, 1998]. And we measured the similarity between words based on their connectivity [Pedersen et al., 2004]. As the WordNet only collects semantic features of nouns and verbs, only IV nouns and verbs surrounding an OOV word were processed. For an IV noun or verb, we ranked the other IV nouns or verbs based on their semantic similarity. We only selected the top three most similar IV words to prevent from adding too many  $n$ -grams. We also skipped words which are morphological changes of the IV word, such as “HOTELS” to “HOTEL”. Moreover, if the similarity between IV words is too small, we did not add them into the language model. For example, we did not want to substitute a proper noun with other IV words, and in WordNet, the similarity between proper nouns is usually very small.

## 6.2 OOV word recovery through filtering

As we mentioned in Chapter 2, in certain applications, ASR systems cannot apply a very large vocabulary, although there are normally more data available for building a larger lexicon or a larger language model. For example, in a dialog system, there is generally not enough computation resource to handle the large vocabulary continuous speech recognition (LVCSR) or the LVCSR is too slow for real time interactions. In other situations, it is not always the best to use a very large vocabulary in an ASR system. A larger vocabulary tends to increase the perplexity of the language model to the testing data. Furthermore, increasing the vocabulary size may also increase the acoustic confusability between words, which also makes the recognition task harder. Figure 6.3 presents the OOV rate, the recognition performance and the run time when increasing the vocabulary size of an ASR system. We can find that the OOV rate drops dramatically when we start to increase the vocabulary size until the vocabulary reaches 60k words. The WER curve is similar to the OOV rate one. The WER drops when increasing the vocabulary size. The lowest WER is achieved when using a 60k-word vocabulary. After that, the WER starts to increase when adding more words into the vocabulary. At last, the recognition run time increases monotonically when the vocabulary becomes larger. On the other hand, even if we use a very large vocabulary in an ASR system, such as a commercial voice search system, there will still be OOV words, because new words always emerge in a language. As large commercial systems cannot afford to update their lexicon and language model very often, such new words will be OOV words for a while. But we can probably find numerous examples of those new words on Internet very shortly after they appear, and use those data to estimate language model scores of new OOV words. For instance, very shortly after the FBI released the name of the Boston Bombing suspects, you could find tons of articles online writing about those names. However, many commercial systems might not be able to recognize those names for certain time, as they

were infrequent foreign names which you probably would not find in the lexicon. Even if those words were in the recognizer’s lexicon, people might not know how to pronounce it correctly. Therefore, the recognizer still could not recognize those names. If the ASR system can automatically detect those new words or new pronunciations, then utilize extra resources to recover those words, the ASR system will be able to recognize emerging OOV words in a short time without human intervention. Therefore, an interesting question is how an ASR system can utilize those extra resources for OOV word recovery after it detects OOV words in the testing speech.

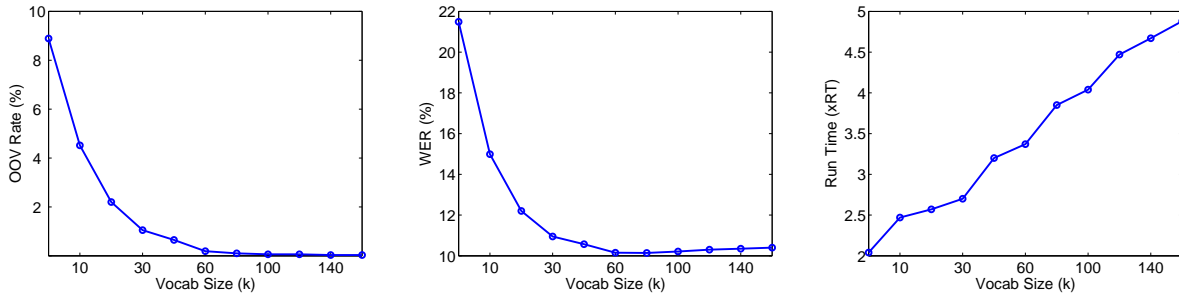


Figure 6.3: The OOV rate, WER and run time of the WSJ Eval set when increasing the vocabulary size of the recognizer.

In this thesis, we investigated a new OOV word recovery approach when a larger lexicon and a larger text corpus are available. Formally, we first collected detected OOV words from the hybrid decoding output. For each detected OOV word, we searched for words in the large lexicon. If the phonetic distance between the OOV word and the word in the large lexicon was small enough, or say if those two words sounded very similar, the word in the large lexicon may actually be the OOV word we detected in the testing speech. We then added that word into our recognizer’s vocabulary. Here, we calculated the phonetic distance in the same way as Eq. 5.3 when measuring the similarity between OOV samples in Chapter 5. After processing all words in the larger lexicon, we ended up with a new vocabulary which contains all existing IV words and new detected OOV words. At last, We used the larger text corpus to train a new language model, which incorporated  $n$ -gram scores for both IV and OOV words.

### 6.3 Recovering recurrent OOV words

In Chapter 5, we discussed how to identify recurrent OOV words through a bottom-up clustering process. As shown in Table 6.1, after OOV word clustering we found three OOV candidates all correspond to the same OOV word “PASHOVSKI”. We now have three different phonetic representations and different context for one OOV word. Such multiple instances of the same OOV word incorporate valuable information. And in this section, we will study how to utilize those knowledge to improve the OOV word recovery performance.

According to our best grapheme-to-phoneme (G2P) conversion output, the correct pronunciation for “PASHOVSKI” should be “ P AH SH AA V S K IY”. We can notice that none of the three pronunciations is correct. Therefore, if we directly perform the P2G conversion on those pronunciations, the estimated written forms may all be wrong. Instead, we could use those

Table 6.1: Examples of the OOV word clustering results.

OOV	Pronunciation	Context
PASHOVSKI	K R AH SH N AA V S K IY	... replace mr PASHOVSKI as ambassador ...
PASHOVSKI	P AH S EH V S K IY	... Bulgarian ambassador PASHOVSKI said he ...
PASHOVSKI	P AE SH AA F S K IY	... dismissal ambassador PASHOVSKI join us ...

different pronunciations simultaneously to build a better phonetic representation for OOV word “PASHOVSKI”. Specifically, we applied a similar approach as ROVER to combine different phone sequences, so as to produce a better pronunciation for recurrent OOV words. For instance, after combining the three decoded pronunciations of OOV word “PASHOVSKI”, we obtained the correct phone sequence “P AH SH AA V S K IY”. As a result, we also recovered the correct written form for this OOV word.

Since we estimated the language model scores of an OOV word based on its POS label, we could also learn a better POS label for an OOV word from its multiple instances, which might then improve the language model scores we learned for the word. Again, we estimated the POS label of recurrent OOV words by voting among the POS labels of its multiple instances.

Another advantage of identifying recurrent OOV words is that we can easily discover different context for the OOV word. As we mentioned previously, an OOV word may appear in a different context when encountered by the recognizer in the future. Therefore, we have to estimate possible context by finding other IV words similar to the surrounding IV words of an OOV word. By doing that, we usually add a large number of new  $n$ -grams into the language model, as we are not sure which context the OOV word may really appear. With the multiple instances of a recurrent OOV word, we not only have more knowledge about the OOV word but also obtain multiple  $n$ -grams of the word. Furthermore, we could also use those contextual information to predict new  $n$ -grams for recurrent OOV words without adding too many irrelevant entries into the language model.

## 6.4 Experiment setup

We used the same datasets as Chapter 4 in our OOV word recovery experiments. The optimal configurations of hybrid systems were adopted to perform OOV word detection. Then OOV word recovery experiments were conducted on the OOV word detection output. To evaluate OOV word recovery, we introduced the following metrics:

- *Pronunciation Accuracy (PA)*, which measures how many detected OOV words’ pronunciation are correct. *PA* can be calculated as

$$PA = \frac{\#\text{OOVs detected with correct pronunciation}}{\#\text{OOVs detected}} \times 100\%. \quad (6.5)$$

- *Recovery Rate (RR)*, which measures how many detected OOV words’ written forms are successfully recovered. *RR* can be calculated as

$$RR = \frac{\#OOVs \text{ recovered}}{\#OOVs \text{ detected}} \times 100\%. \quad (6.6)$$

- *Word Error Rate (WER)*, which measures how many errors are made by the recognizer. *WER* can be calculated as

$$WER = \frac{\#Substitution \text{ errors} + \#Deletion \text{ errors} + \#Insertion \text{ errors}}{\#Words \text{ in reference}} \times 100\%. \quad (6.7)$$

In this thesis, we calculated *Pronunciation Accuracy (PA)* and *Recovery Rate (RR)* as the average of the results from hybrid systems built with the OOV cost  $C_{OOV}$  ranging from 0 to 2.5 with a step size of 0.25.

To evaluate the overall OOV word learning performance, we need to test how many recovered OOV words can be recognized by the ASR system in the future, such as whether the system can recognize the same OOV words appear in a different context spoken by a different speaker. Initially, we considered to learn OOV words from the Dev set and test the learning result on the Eval set. However, as shown in Table 4.1, there are very few common OOV words between the Dev and Eval sets of each task. Therefore, we collected another Test set, which contains OOV words learned from the Eval data. We tried to make the new Test set to have similar properties as the old Dev and Eval sets, such as a similar OOV rate and recognition accuracy. The OOV rate and baseline word recognition result of new Test data are given in Table 6.2. We can find that the BN Test data has a higher OOV rate than the Eval data, while the Test data of the WSJ and SWB tasks has a similar OOV rate as the Eval data. The baseline WER of three tasks is in the same range as the WER on the Eval data. In Table 6.2, we can also find how many of the Test OOV words also appear in the Eval data. It can be seen that the Test data of the WSJ and SWB tasks shares more than 50% OOV words with the Eval data. But the BN Test data only has less than 25% OOV words that also appear in the BN Eval data.

Table 6.2: The OOV rate and WER of the new Test data.

	WSJ	BN	SWB
OOV Rate	2.13%	2.79%	1.80%
OOV in Eval	52.3%	23.9%	66.1%
WER	10.13%	30.39%	33.26%

In our experiments, we found some incorrectly detected and recovered OOV words were very short words, such as words with only one or two letters. But in our Dev, Eval or Test data, there was no such short OOV word. The shortest OOV word had at least three letters. In general, we will probably not observe too many OOV words with only three letters or any OOV word with just one or two letters in a practical ASR system. Therefore, in this thesis, we skipped short recovered OOV words which contained only one or two letters.

## 6.5 Experimental results

### 6.5.1 The baseline OOV word recovery results

#### The results of estimating the OOV word written form through the P2G conversion

We collected the detected OOV words from the hybrid decoding result, then estimated the written form of an OOV word from its decoded pronunciation through the P2G conversion. Here, we first measured how many detected OOV words had the correct pronunciation and how many OOV words had the correct written form after the P2G conversion. The *Pronunciation Accuracy* (PA) of different hybrid systems is given in Table 6.3. We can find that hybrid systems in the WSJ and SWB tasks have higher PAs than systems in the BN task. This is consistent with what we found in Chapter 4 and Chapter 5. As the utterance is usually very long and many OOV words appear in the same utterance, the BN task is much harder than the WSJ and SWB tasks. Among different systems, the syllable hybrid system has higher PAs than the subword and grapheme hybrid systems. This is because syllable units are more constraint than subword and grapheme units, thus the P2G conversion from syllable units is much easier.

Table 6.3: The pronunciation accuracy of different hybrid systems.

PA (%)	Syllable	Subword	Grapheme
WSJ	39.8	39.0	38.1
BN	20.7	14.9	18.5
SWB	47.8	43.0	39.2

Table 6.4: The recovery rate of different hybrid systems.

RR (%)	Syllable	Subword	Grapheme
WSJ	18.4	15.2	38.2
BN	13.0	6.1	22.6
SWB	36.2	33.3	45.6

Furthermore, by comparing PAs with *Recovery Rates* (RRs) given in Table 6.4, we can find that RR is lower than PA in the syllable and subword hybrid systems, indicating that not all correct OOV pronunciations produce correct spellings after OOV word recovery. But in the grapheme hybrid system, RR is even higher than PA, since some OOV words are in fact successfully recovered from incorrect pronunciations. In the syllable and subword hybrid systems, the spellings are recovered through the P2G conversion, which may involve additional errors. While in the grapheme hybrid system, letters and phones are aligned and simultaneously modeled as a pair. Then during decoding, the pronunciation and the written form of an OOV word are evaluated as a single unit. Thus correct spellings may still be recovered from graphemes with pronunciation errors.

The OOV word recovery results are given in Figure 6.4, where the solid lines are the WERs of detection results in which hypothesized OOV words are represented as sub-lexical units, the

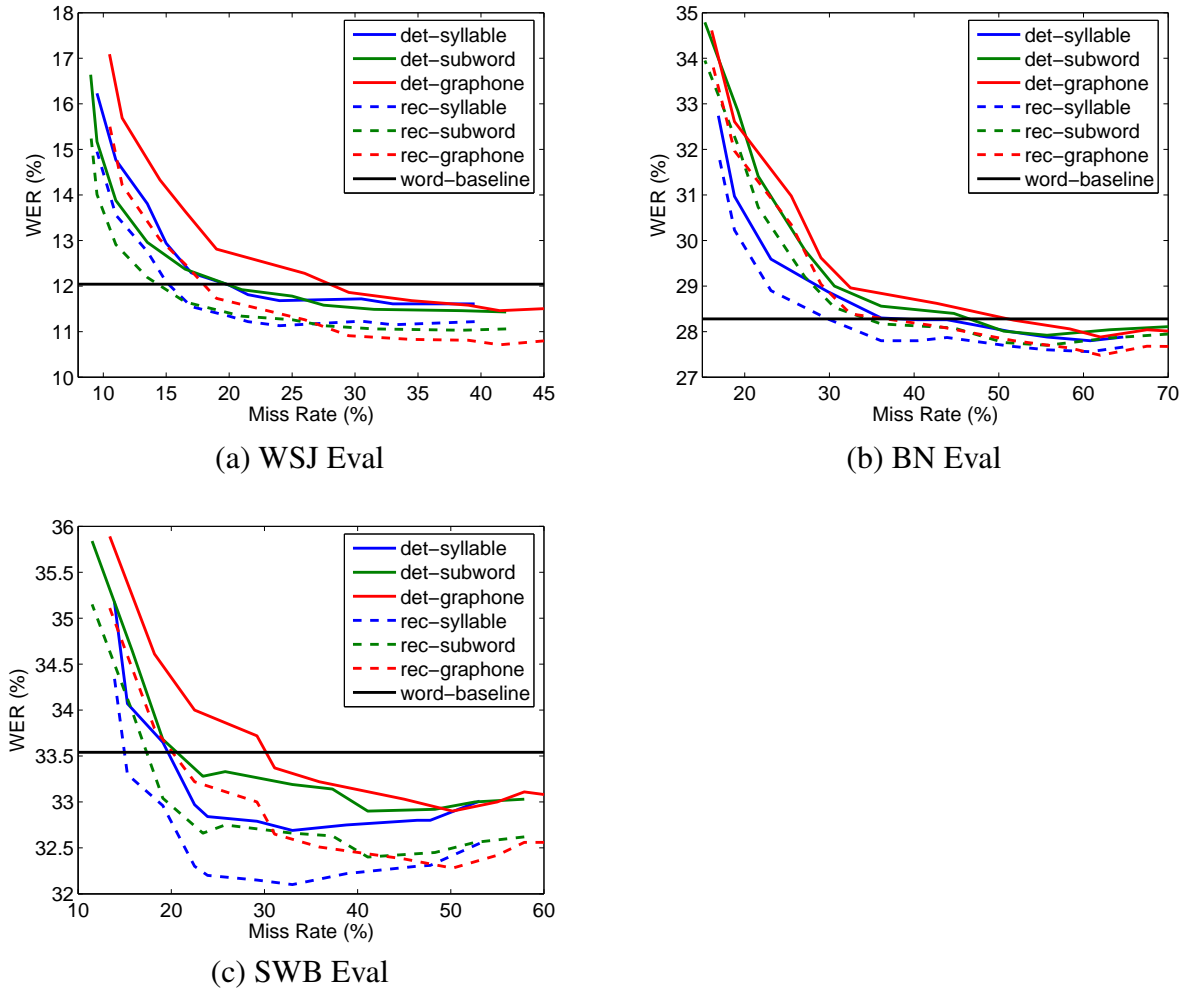


Figure 6.4: The OOV word recovery results using trigram flat hybrid models with different sub-lexical units.

dotted lines are recovery results in which the written forms of hypothesized OOV words are estimated through the P2G conversion, and the black horizontal solid line corresponds to the baseline word recognition performance. We can notice that even without recovering OOV words' written forms, sometimes the detection results are still better than the word baseline. This is because by successfully detecting the presence of OOV words, the recognizer can avoid making errors on the surrounding IV words of an OOV word. But when the hybrid system starts to report more OOV words, the *Miss Rate* becomes lower and the *False Alarm Rate* becomes higher. The hybrid system now makes more detection errors, which then causes more recognition errors and worse WERs than the word baseline.

Now, we focus on the OOV word recovery results in Figure 6.4. It can be seen that compared to the detection results, around 1% absolute improvement on WER is achieved after recovering the OOV word's written form. The differences between the syllable, subword and graphone hybrid systems are small, but the syllable hybrid system is slightly better than the other two



systems. For the best performance, there is about 11% relative improvement over the word baseline in the WSJ task, 3% in the BN task and 4% in the SWB task. Such improvement on WER comes from three sources: 1) by detecting OOV words, the hybrid system recognizes IV words that are adjacent to OOV words better; 2) some OOV words are successfully recognized after recovering the OOV word’s written form; 3) a few IV words are mis-detected as OOV words but later recovered.

### Utilizing a larger dictionary

Similar to OOV word detection, where a larger dictionary could be very helpful, we can also utilize the large dictionary to train a better P2G model for estimating the written form of an OOV. Therefore, we trained a 6-gram joint-sequence model from the large CMUdict and applied this model in the P2G conversion. And here, we worked on the decoding results of hybrid systems built with the large CMUdict. The PA and RR when using the large dictionary are shown in Table 6.5 and Table 6.6. First, we can see that the WSJ and SWB tasks still have better performance than the BN task. Meanwhile, the syllable systems are still the best. Then by comparing those tables with PA and RR when using a smaller dictionary in Table 6.3 and Table 6.4, we can find that more correct pronunciations and spellings are produced for OOV words. PA is improved, as now we have a better model of OOV words when training sub-lexical units using the large CMUdict. And because of the better P2G model trained from the large CMUdict, we also have better RRs. The difference between PA and RR is also smaller. But in the graphone system, RR is smaller than PA now, which again may be caused by the use of less optimal joint-sequence model built from the small dictionary when training graphone units.

Table 6.5: The pronunciation accuracy of hybrid systems built with the large dictionary.

PA (%)	Syllable	Subword	Graphone
WSJ	55.3	50.9	47.2
BN	32.7	26.8	27.1
SWB	64.2	56.8	53.1

Table 6.6: The recovery rate of hybrid systems built with the large dictionary.

RR (%)	Syllable	Subword	Graphone
WSJ	49.7	44.2	42.4
BN	30.8	25.3	26.2
SWB	59.4	53.4	52.6

From the OOV recovery results given in Figure 6.5, we can find that about 1% absolute improvement on WER is obtained when using hybrid systems and P2G models built with the large CMUdict. Compared to the baseline word recognition results, because of using a better hybrid model and a better P2G model trained from the large dictionary, we can now achieve 14% relative improvement on WER in the WSJ task, 4% in the BN task and 6% in the SWB task. Yet

again, the graphone hybrid system can perform better if we use a better joint sequence model to segment OOV words into graphone units.

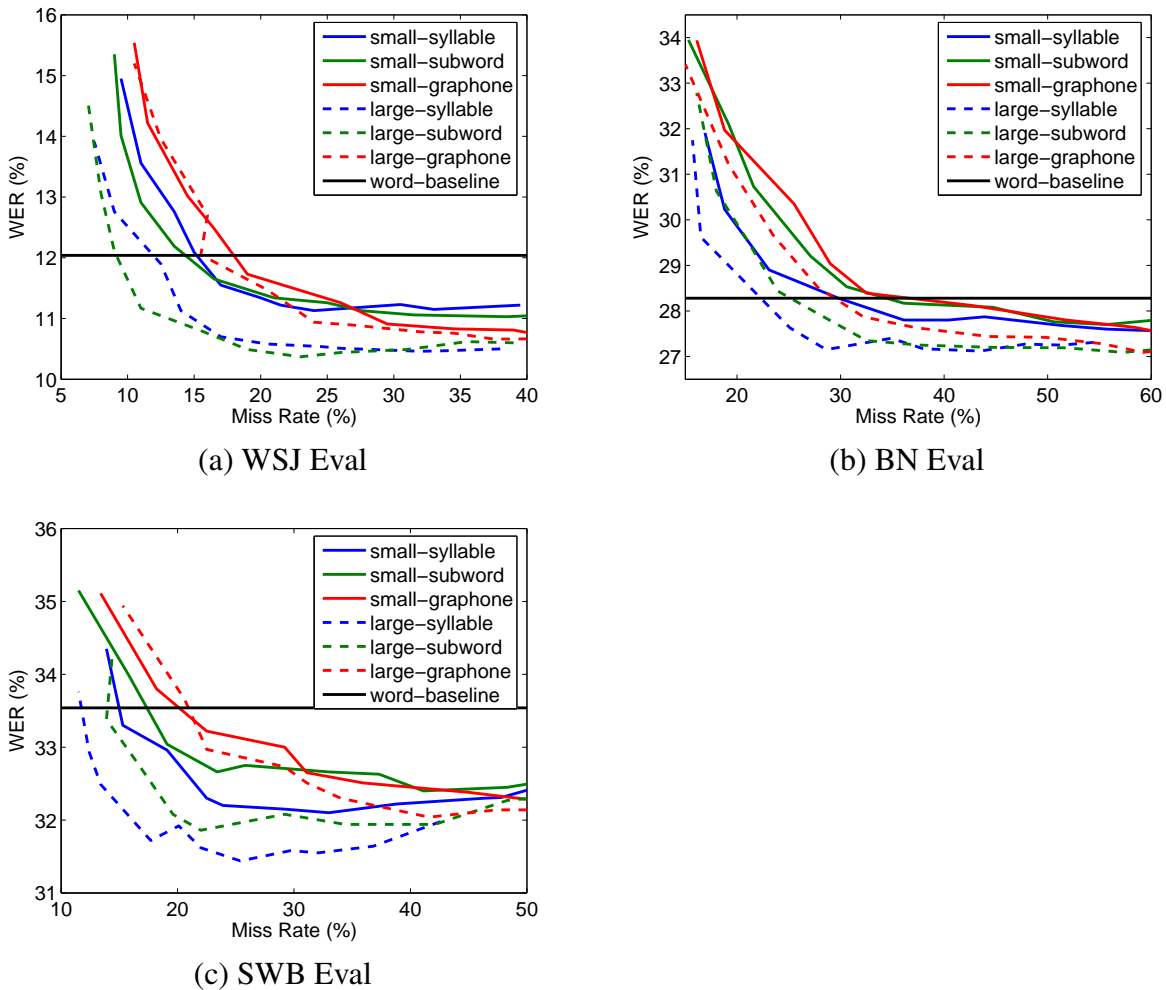


Figure 6.5: The OOV word recovery results using hybrid systems and P2G models built with small and large dictionaries.

### Training a better P2G model from recognition errors

To train a better joint-sequence model for the P2G conversion, we performed hybrid decoding on the training speech. Table 6.7 presents the statistics on OOV words in the training speech of each task. We can find that the OOV rate in the training speech is similar to that in the Dev, Eval and Test data. There are more OOV tokens and OOV words in the BN and SWB task than in the WSJ task. Moreover, in the WSJ task, there are 14k OOV tokens but only 3k unique OOV words. As a result, there are fewer OOV samples we could learn from in the WSJ task than the other tasks, which may limit the improvement by learning the P2G model from those data.

Table 6.7: The statistics on OOV words in the training speech.

	WSJ	BN	SWB
OOV Rate	2.2%	2.9%	2.0%
No. OOV Tokens	14k	44k	31k
No. OOV Words	3k	14k	14k

We selected the best hybrid system to perform hybrid decoding in each task, which is the subword hybrid system in the WSJ and BN tasks and the syllable hybrid system in the SWB task. The OOV word detection performance on the training speech is given in Table 6.8. We can find that the OOV word detection performance on the training speech in the WSJ and BN tasks is similar to that on the Dev and Eval data. But in the SWB task, the OOV word detection performance is much worse than that on the Dev and Eval data. This may be because that there are more word fragments in the training speech than in the Dev and Eval data. The hybrid system could incorrectly recognize those word fragments as OOV words, which will result in more false alarm errors. We also experimented with using the subword hybrid system output for training the P2G model used in the subword system and using the syllable hybrid system output for training the P2G model used in the syllable system. The performance is essentially the same as just using the best system in each task to perform hybrid decoding on the training speech.

Table 6.8: The OOV word detection performance on the training speech.

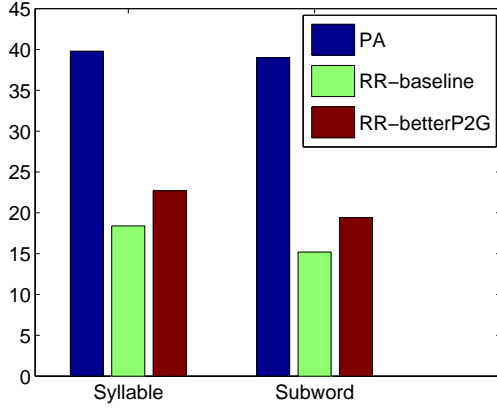
(%)	WSJ	BN	SWB
Precision	65.0	55.9	49.7
Recall	66.8	39.4	56.7
F1	65.8	46.2	53.0

As we collected the aligned data from the hybrid decoding result, let us count how much data we can obtain for training the joint-sequence model. The number of alignments and unique OOV words collected from the hybrid decoding result are provided in Table 6.9. We can see that there are normally more alignments than unique OOV words. This is because the same OOV word may be recognized into different pronunciations in different context. Furthermore, there are much more alignments in the BN and SWB tasks than in the WSJ task. Hence, we had more training samples in the BN and SWB tasks.

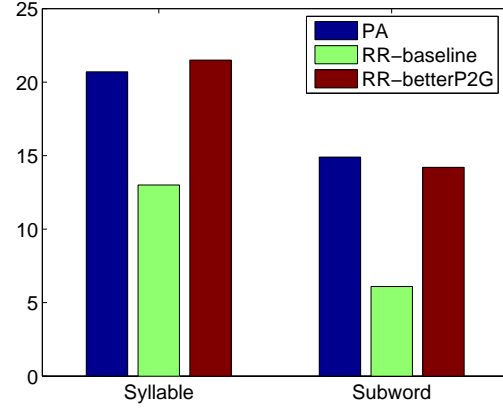
Table 6.9: The number of alignments and unique OOV words in the hybrid decoding output of the training speech.

	WSJ	BN	SWB
No. Alignments	4k	11k	11k
No. OOVs	2k	9k	9k

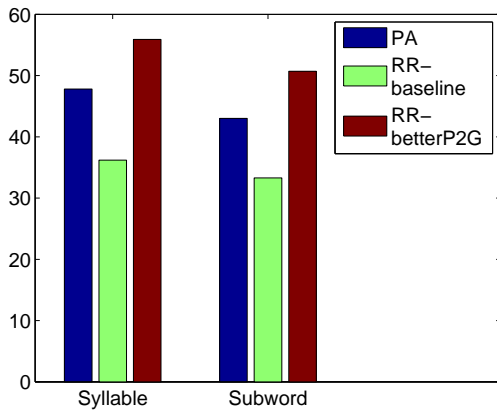
The PA and RR of hybrid systems with the baseline and better P2G conversion is shown in Figure 6.6. It can be seen that RR is significantly increased when using the better joint-sequence



(a) WSJ Eval



(b) BN Eval



(c) SWB Eval

Figure 6.6: The pronunciation accuracy and recovery rate of hybrid systems with the baseline and better P2G conversion.

model for P2G conversion. Among three tasks, the improvement on RR in the BN and SWB tasks is more than that in the WSJ task. Again, this may be because there are more alignments for training the better P2G model in the BN and SWB tasks than in the WSJ task. Furthermore, in the BN and SWB tasks, RR is now even larger than PA, which clearly indicates that many OOV words were now recovered from incorrect pronunciations.

Finally, let us take a look at the OOV word recovery results when using the better joint-sequence model for P2G conversion. As given in Figure 6.7, the solid line is WER when using the baseline P2G conversion for OOV word recovery, while the dotted line is WER when using the better joint-sequence model for P2G conversion. We can find that the WER is always lower when using the better P2G conversion than the WER when using the baseline P2G conversion. By improving the P2G conversion performance, we can now achieve 9% relative improvement over the word baseline in the WSJ task, 3% in the BN task and 5% in the SWB task through OOV word recovery.

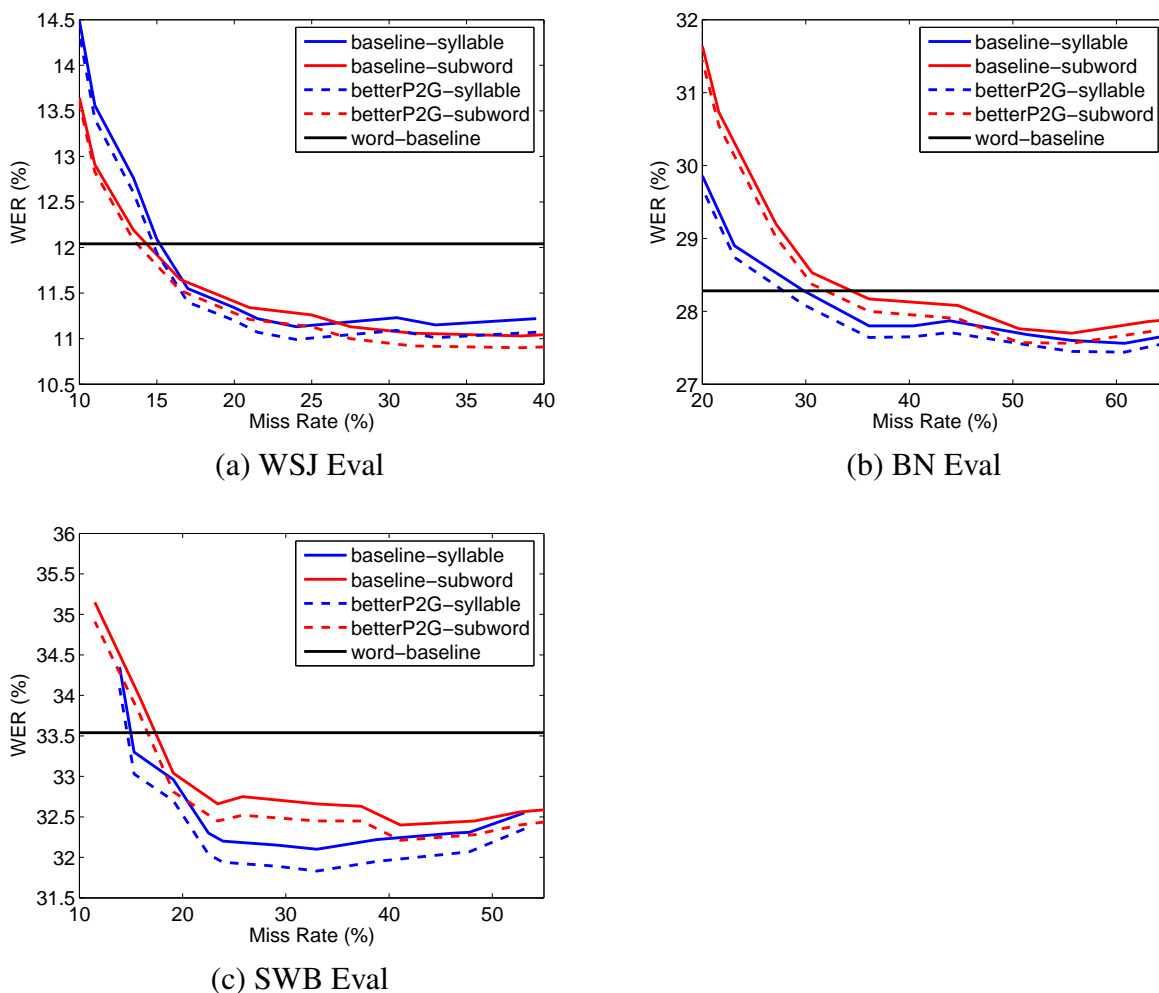


Figure 6.7: The OOV word recovery results with the baseline and better P2G conversion.

### The results of estimating language model scores of an OOV word

After estimating the written form of an OOV word and integrating the word into the recognizer’s lexicon, we need to estimate language model scores for the OOV word. As we estimated the language model scores of an OOV word based on the POS labels of the word and its surrounding IV words, let us first see how good the POS tagger we used. We manually labeled all OOV words in the Eval data. Then we counted how many OOV words were assigned the correct label when running POS tagging on the transcription of the Eval data. There are two models in the POS tagger. One model was trained from the WSJ text data, hereafter referred as the “WSJ” model. The other model was trained from WSJ and some extra text data, hereafter referred as the “English” model. The POS label accuracy on OOV words in the Eval data is provided in Table 6.10. It can be seen that the WSJ model usually performs better than the English model. Therefore, in the following experiments, we applied the WSJ model when performing POS tagging on the hybrid decoding output. Overall, the POS label accuracy in the WSJ and BN tasks is better than that

in the SWB task. This is because compared to the read speech, the spontaneous conversational speech which contains many word repetitions and fragments, is more difficult for POS tagging.

Table 6.10: The POS label accuracy on OOV words in the Eval data.

(%)	WSJ	BN	SWB
WSJ model	84.5	84.3	77.5
English model	85.5	82.0	74.2

Now, we take a closer look at the distribution of POS labels of OOV words in the Eval data. From Table 6.11, we can find that most OOV words are nouns, especially proper nouns. There are also many verbs and adjectives as well as very few adverbs. We can also find some compound labels in all tasks.

Table 6.11: The distribution of POS labels of OOV words in the Eval data.

Label	WSJ	BN	SWB
JJ	12	15	25
NN	48	50	81
NNP	87	144	44
NNPS	0	0	6
NNS	21	21	20
RB	1	1	0
VB	5	7	17
VBD	1	2	3
VBG	9	3	2
VBN	11	4	7
VBP	0	0	2
VBZ	3	2	0
NN+POS	0	1	0
NNP+POS	2	5	2
Total	200	255	209

Our language model score learning experiments were performed on the hybrid decoding output of the Eval data using hybrid systems with the most recovered OOV words. Specifically, we selected the grapheme system with  $C_{OOV} = 1.5$  in the WSJ task, grapheme system with  $C_{OOV} = 1.25$  in the BN task, and syllable system with  $C_{OOV} = 0.75$  in the SWB task. The number of recovered OOV words whose written forms are correct and the POS label accuracy of those OOV words are presented in Table 6.12. It can be seen we can recover about 40% OOV words in the WSJ and BN tasks and a little more than 20% OOV words in the BN task. The POS label accuracy on recovered OOV words is similar to the numbers in Table 6.10. In fact, even on all detected OOV words including words whose written form is not correct, the POS label accuracy is still fairly good. Because the POS tagger usually assigns the “NNP” label to words it never observes, such as words with an incorrect written form. And we have a large number of

OOV words with the “NNP” label. Therefore, it is highly possible that those unrecovered OOV words can still obtain correct POS labels.

Table 6.12: The number of recovered OOV words and their POS label accuracy.

	WSJ	BN	SWB
OOV recovered	79	60	93
POS accuracy	87%	83%	70%

After detecting OOV words and recovering the written form of OOV words from the Eval data, we incorporated all hypothesized OOV words which had more than 2 letters into our recognizer’s lexicon. There were a few OOV words even with multiple pronunciations, as they were recognized differently in different utterances. Figure 6.8 presents the change of the vocabulary size and OOV rate on the Eval and Test data when integrating OOV words into the recognizer’s lexicon. It can be seen that by augmenting the vocabulary with a small number of OOV words learned from the Eval speech, there is a big drop of OOV rate on both the Eval and Test data. The drop of OOV rate in the Eval data is more than that in the Test data, because there are only up to 50% OOV words in the Test data that also appear in the Eval data.

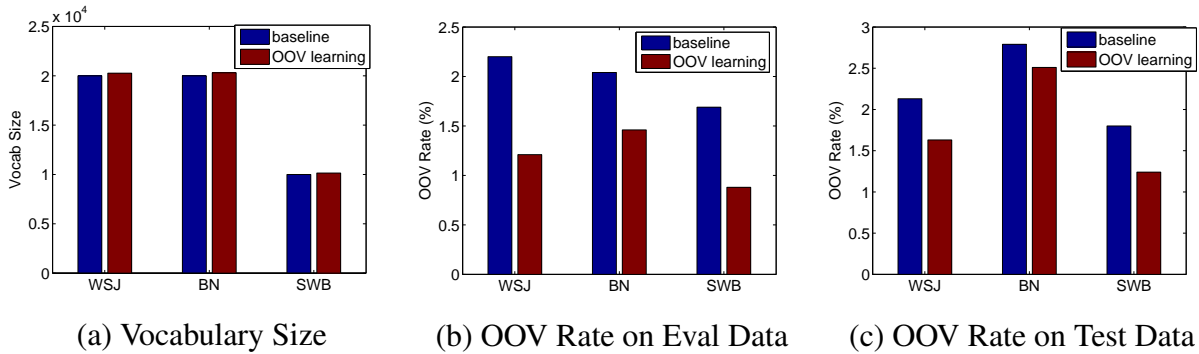


Figure 6.8: The vocabulary size and OOV rate on the Eval and Test data of the baseline and expanded vocabulary.

Table 6.13: The number of hypothesized OOV words and OOV words eliminated in the Eval and Test data.

	WSJ	BN	SWB
Hypothesized OOVs	267	305	143
Eliminated OOVs in Eval	90	73	101
Eliminated OOVs in Test	61	39	119

As shown in Figure 6.8, after OOV word recovery, many OOV words were eliminated from the OOV word list and converted into IV words. The number of hypothesized OOV words and OOV words eliminated in the Eval and Test data are shown in Table 6.13. Up to 50% OOV words

were eliminated in the Eval data of the WSJ and SWB tasks and up to 30% OOV words were eliminated in the Eval data of the BN task. We can find that the number is better than Table 6.12. This is because some OOV words appeared multiple times in the Eval data. But our system did not recovered all instances of those OOV words. However, when we added detected OOV words into lexicon, all instances of those OOV words become IV words. Therefore, more OOV words were eliminated. Again, the number of OOV words eliminated in the Test data is smaller than that in the Eval data, as there are fewer Eval OOV words in the Test data. On average, more than 40% OOV words in the Test data were converted into IV words.

Table 6.14: The number of  $n$ -gram scores added into the language model.

	WSJ	BN	SWB
Unigrams	267	305	143
Bigrams	536	621	312
Trigrams	683	860	442

After estimating the POS labels of an OOV word and its surrounding IV words in the hybrid decoding output of the Eval data, we estimated language model scores for the word, then integrated those new  $n$ -grams into the recognizer’s language model. Table 6.14 shows the number of  $n$ -gram scores added into the language model. We can find a proper number of  $n$ -grams were incorporated into our language model.

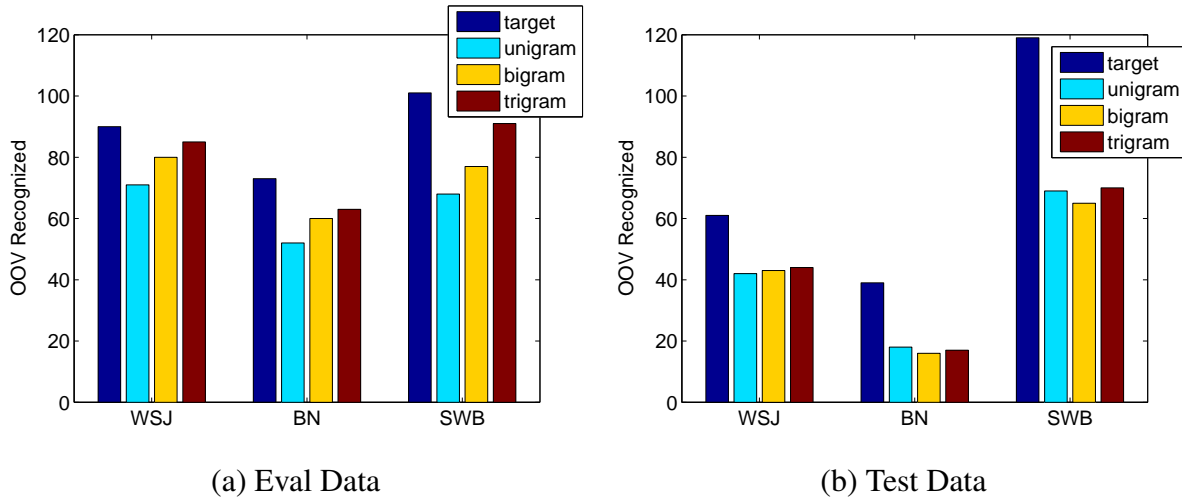


Figure 6.9: The number of recognized OOV words when integrating OOV  $n$ -grams into the language model.

The goal of learning OOV words is to correctly recognize those words when encountered by the recognizer in the future. To evaluate our OOV word learning framework, we measure how many of the eliminated OOV words in Table 6.14 can be recognized in the second pass decoding of the Eval data and in the first pass decoding of the Test data. Figure 6.9 shows the number of recognized OOV words in the Eval and Test data when integrating OOV unigram,



bigram and trigram scores into the language model, where the dark blue bar is the target number of OOV words we tried to recognize, while the light blue, yellow and red bars correspond to the number of OOV words recognized when incorporating OOV language model scores. It can be seen that when only adding unigram scores of OOV words into the language model, we can already recognized more than half of OOV words in both the Eval and Test data. When adding longer  $n$ -grams, such as bigrams and trigrams, more OOV words are recognized in the Eval data, but not in the Test data. As we mentioned previously, OOV words in the Test data usually appear in different context from OOV words in the Eval data. Therefore, adding bigrams and trigrams learned from the Eval context does not help to recognize more OOV words in the Test data. That is why we also need to estimate new context an OOV word may appear. The recognition performance when integrating OOV  $n$ -grams into the language model is given in Figure 6.10, where the dark blue bar is the baseline word recognition WER, while the light blue, yellow and red bars are the WERs when decoding with estimated OOV language model scores. Different from the number of recognized OOV words, the WER was slightly higher when integrating OOV bigram and trigram scores into the language model. Although we are able to recognized more OOV words, the recognizer probably also misrecognizes many IV words as OOV words, because of higher order  $n$ -grams added into the language model. As a result, we cannot observe improvement on WER when integrating OOV bigrams and trigrams into the language model.

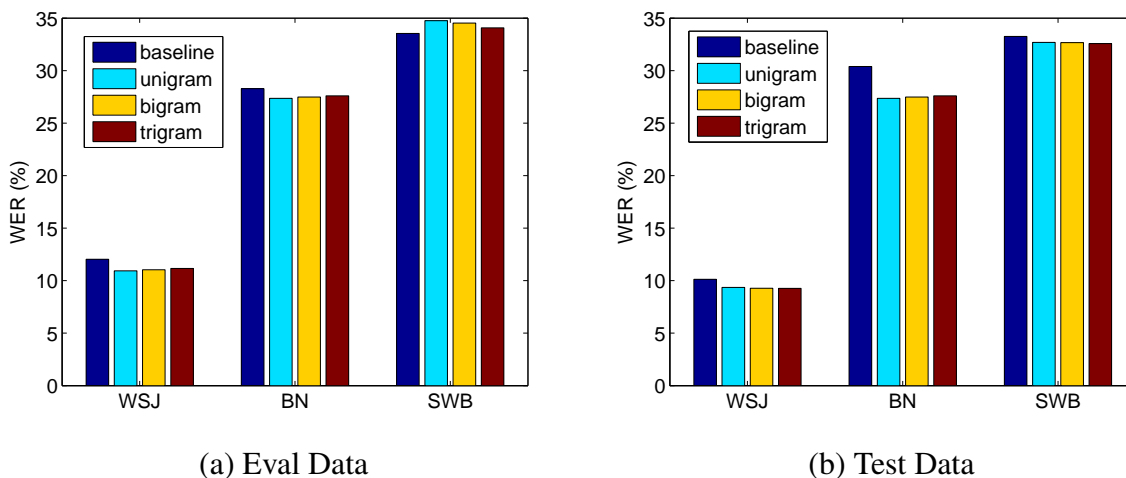


Figure 6.10: The recognition performance when integrating OOV  $n$ -grams into the language model.

After integrating the OOV  $n$ -grams into the language model, we estimated new context an OOV word may occur. For nouns and verbs in the left and right two words of an OOV word, we found other semantic similar IV nouns and verbs, then formed new OOV  $n$ -grams using those words. The number of new OOV bigram and trigram scores added into the language model is given in Table 6.15. It can be seen that a large number of new bigram and trigram scores were incorporated into the language model. In fact, since we only substituted nouns and verbs with high semantic similarities, we already skipped many other possible context. Otherwise, there will be even more new OOV bigrams and trigrams. For comparison, we also experimented with

only adding oracle new OOV  $n$ -grams extracted from the transcription of the Test data. The scores of those oracle  $n$ -grams were still estimated based on the POS labels of OOV words and their surrounding IV words. The number of oracle OOV  $n$ -grams added into the language model is presented in Table 6.16. We can find that there are much fewer oracle OOV  $n$ -gram compared to that of the estimated new  $n$ -grams.

Table 6.15: The number of new OOV  $n$ -grams added into the language model.

	WSJ	BN	SWB
Bigrams	590	689	346
Trigrams	887	947	492

Table 6.16: The number of oracle OOV  $n$ -grams added into the language model.

	WSJ	BN	SWB
Bigrams	46	49	131
Trigrams	84	82	102

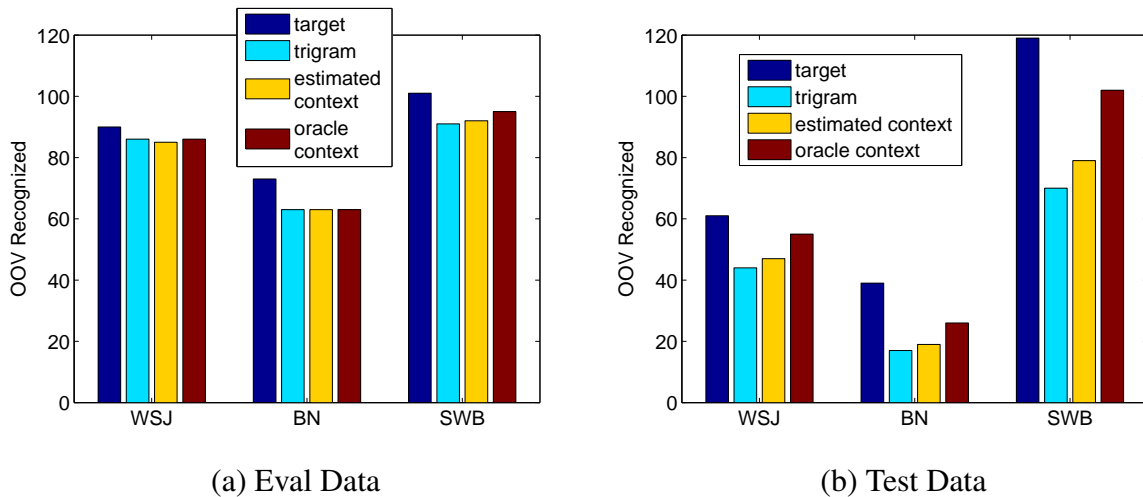


Figure 6.11: The number of recognized OOV words when integrating unseen OOV  $n$ -grams into the language model.

Figure 6.11 presents the number of recognized OOV words when integrating new OOV  $n$ -grams into the language model. The dark blue bar is the target number of OOV words we try to recognize, the light blue bar is the result when only adding seen  $n$ -grams into the language model, the yellow bar is the result when adding estimated new  $n$ -grams, while the red bar correspond to the result when adding oracle new  $n$ -grams. We can find that a few more Test OOV words were recognized when adding estimated unseen  $n$ -grams into the language model. Furthermore, new  $n$ -grams from the Test data did not increase the number of recognized OOV words in the

Eval data. We can also see that our system recognized much more OOV words with oracle new  $n$ -grams, which indicates that the estimated new  $n$ -grams were not good enough. The recognition performance when decoding with new OOV  $n$ -grams is given in Figure 6.12, where the dark blue bar is the WER of the baseline word recognition, the light blue bar is the WER when only adding seen OOV  $n$ -grams, the yellow bar is the WER when adding estimated unseen OOV  $n$ -grams and the red bar is the WER when adding oracle unseen OOV  $n$ -grams from the transcription of the Test data. It can be seen that the WER did not change much in both the Eval and Test data when integrating unseen OOV  $n$ -grams into the language model, although we recognized more OOV words in the Test data. This is probably because we added too many irrelevant OOV  $n$ -grams into the language model, which produced many recognition errors. Overall, we could recognize up to 90% learned OOV words in the Eval data and about 60% to 80% learned OOV words in the Test data. And there is about 8% relative improvement on WER in the WSJ task, about 2% relative improvement in the BN task and 1% in the SWB task.

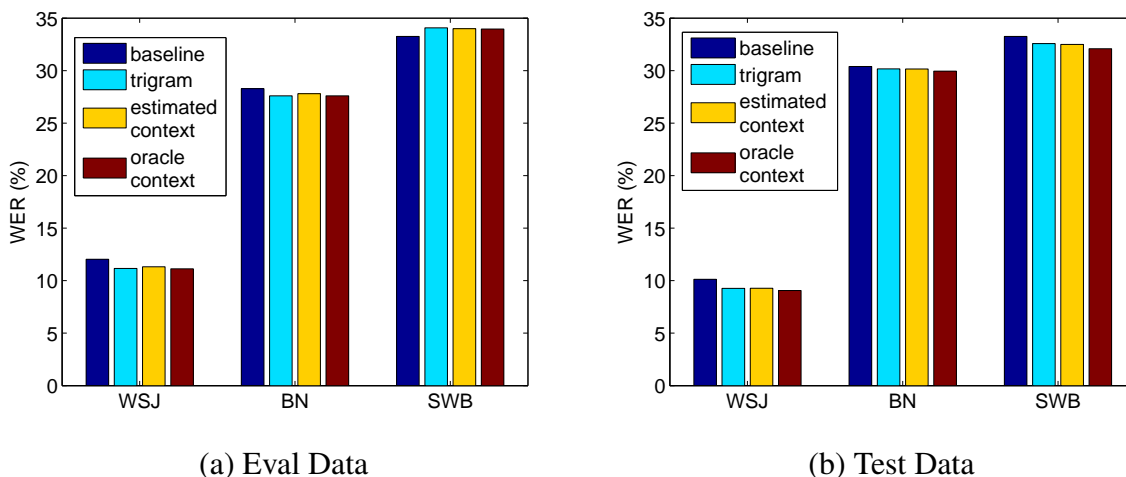


Figure 6.12: The recognition performance when integrating unseen OOV  $n$ -grams into the language model.

## 6.5.2 The results of OOV word recovery through filtering

In this section, we tried to build a vocabulary with a proper size but meanwhile keep the OOV rate as low as possible. We selected the best system's output for our OOV word filtering experiments. Specifically, we selected the subword hybrid system with  $C_{OOV} = 0.75$  in the WSJ and BN tasks and the syllable hybrid system with  $C_{OOV} = 1.0$  in the SWB task. For each task, the vocabulary size of the baseline system and system using all words from the training text data and their corresponding OOV rate on the Eval data is provided in Table 6.17. We can find that when building the system from all words in the training data, the vocabulary size is extremely large. As demonstrated in Figure 6.3, the recognition performance could be hurt when using a very large vocabulary in an ASR system. We can also notice that even using the very large vocabulary, there are still some OOV words in the Eval data.

Table 6.17: The vocabulary size and OOV rate on the Eval data of different systems.

	WSJ		BN		SWB	
	baseline	all	baseline	all	baseline	all
Vocab Size	20,000	165,503	20,000	258,834	10,000	60,752
OOV Rate	2.20%	0.01%	2.02%	0.18%	1.69%	0%

Some statistics about the hybrid decoding output in each task is given in Table 6.18. It can be seen that half of the detected OOV words have the correct pronunciation in the WSJ and SWB tasks, while only about 20% detected OOV words have correct pronunciations in the BN task. Therefore, we can recover those OOV words by searching for words with the same pronunciation in the large lexicon. But we also found that more than 70% of the detected OOV words have identical pronunciations with words in the large lexicon. As a result, we will also add some irrelevant words from the large lexicon into our vocabulary. We could try to utilize some other features when measuring the similarity between detected OOV words and words in the large lexicon, such as the contextual feature used in Chapter 5, which may help to prevent from adding too many irrelevant words.

Table 6.18: The statistics of the hybrid decoding output in each task.

	WSJ	BN	SWB
No. reported OOVs	221	235	210
Pronunciation accuracy	46.2%	19.1%	55.0%
Identical pronunciations	80.1%	67.7%	76.2%

When adding words from the large lexicon to the recognition vocabulary, we could control how many words will be added by setting a proper threshold on the word similarity. Figure 6.14 presents the OOV rate on the Eval data of different vocabularies. We compared two different ways of adding words into the recognition vocabulary. In one method, we added words into the recognition vocabulary based on the word frequency in the training data, which is indicated by the solid blue line. In another method, we added words by adjusting the threshold for the word similarity, which corresponds to the dotted red line. It can be seen that the OOV rate of vocabularies produced by filtering the large lexicon is much lower than that of expanding vocabulary based on word frequency. The gap between the solid line and the dotted line becomes smaller when the vocabulary becomes larger. This is because to add more words into the vocabulary, we had to apply a loose threshold when filtering words in the large lexicon. As a result, we added many irrelevant words rather than true OOV words into our recognition vocabulary.

In practice, we will not add too many words into the recognition vocabulary, since there are generally not that many OOV words in the testing speech. Assuming we will only increase the vocabulary size by 5%, which is 1000 more words in the WSJ and BN tasks and 500 more words in the SWB task. Figure 6.14 provides the OOV rate of ASR systems built with different vocabularies, where the blue bar is the baseline vocabulary, the green bar is the expanded vocabulary based on word frequency and the red bar is the expanded vocabulary through filtering. We can find that both expanded systems have lower OOV rate than the baseline ASR system, as

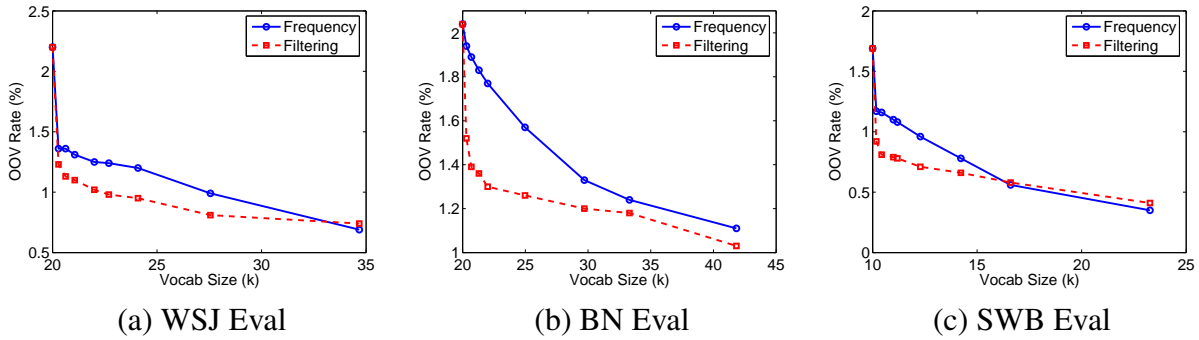


Figure 6.13: The OOV rate of different size recognition vocabularies produced by choosing upon word frequency and filtering a large lexicon.

the expanded vocabularies consist of more words than the baseline vocabulary. The expanded vocabulary from OOV filtering usually has much lower OOV rate than the expanded vocabulary based on word frequency, except in the BN task on the Test data. As shown in Table 6.2, there are less than 25% OOV words in the BN Test data which also occur in the BN Eval data. As a result, learning OOV words from the Eval data cannot reduce the OOV rate on the Test data very much. Overall, by filtering OOV words in the large lexicon, we can reduce the OOV rate by up to 50% in the Eval data and up to 30% in the Test data.

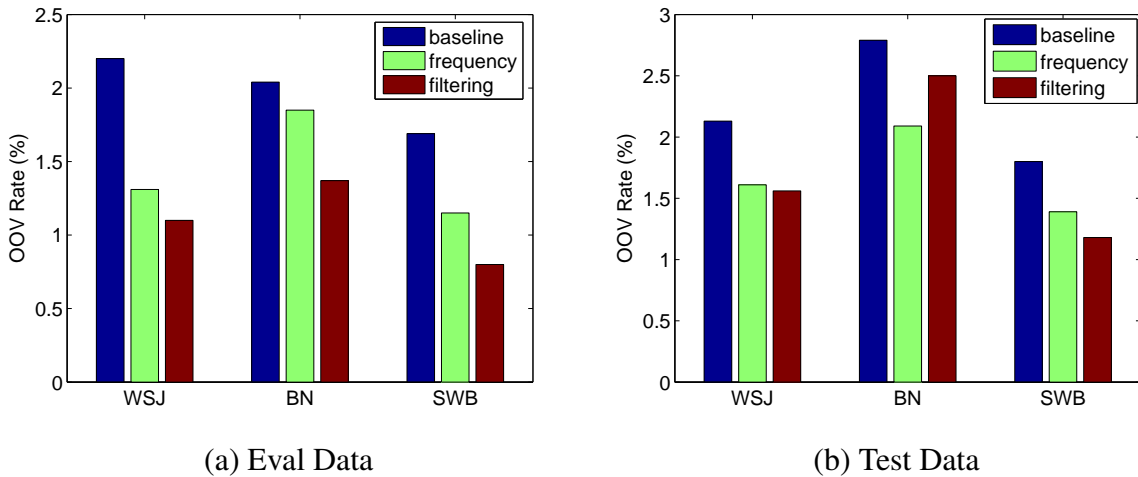


Figure 6.14: The OOV rate of recognition systems built from different vocabularies.

Let us now compare the recognition performance using vocabularies produced by adding words based on their frequency and filtering words in the large lexicon. Figure 6.15 provides the recognition performance on the Eval and Test data when using different vocabularies. We can find that the WER is smaller when using the expanded vocabularies than using the baseline vocabulary. It is simply because the OOV rate of the larger vocabularies is much lower than that of the baseline vocabulary. Between the two expanded vocabularies, the one produced by

filtering the large lexicon is preferred than the one based on word frequency. Again, there are fewer OOVs when using the vocabulary produced by filtering the large lexicon. Overall, when using the expanded vocabulary through filtering, we can achieve up to 10% relative improvement on WER on both the Eval and Test data in the WSJ and SWB tasks, while up to 4% relative improvement in the BN task.

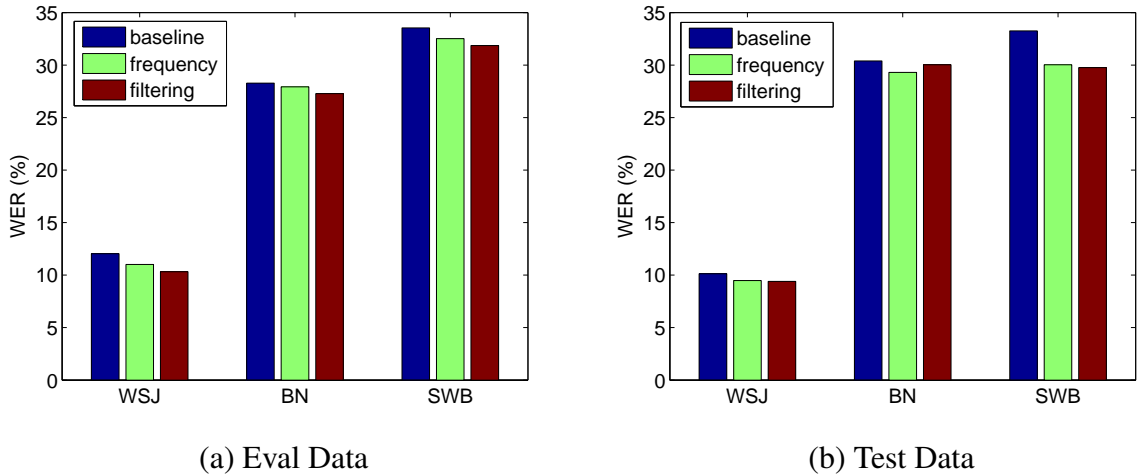


Figure 6.15: The WER of recognition systems built from different vocabularies.

### Comparison of the two OOV word recovery approaches

Since two OOV word recovery approaches were investigated in this paper, here, let us compare the OOV word learning performance of these two approaches. To do that, we produced two expanded vocabularies with the same size. In one approach, hereafter referred as “recovered”, we expanded the recognition vocabulary with detected and recovered OOV words. Then, we estimated language model scores for OOV words based on the POS labels of those words and their surrounding IV words. In another approach, hereafter referred as “filtered”, we searched for phonetic similar words as detected OOV words in the large lexicon and trained language model scores from the extra text data.

The size of the recovered and filtered expanded vocabularies is shown in Table 6.19. We can find the recovered and filtered expanded vocabularies have similar size. In fact, when building the filtered vocabulary, we only added words with the same pronunciations as OOV words from the large lexicon. Otherwise, there will be more words added into the filtered vocabulary.

Table 6.19: The size of the recovered and filtered expanded vocabularies.

	WSJ	BN	SWB
recovered vocab	20267	20205	10143
filtered vocab	20307	20298	10171

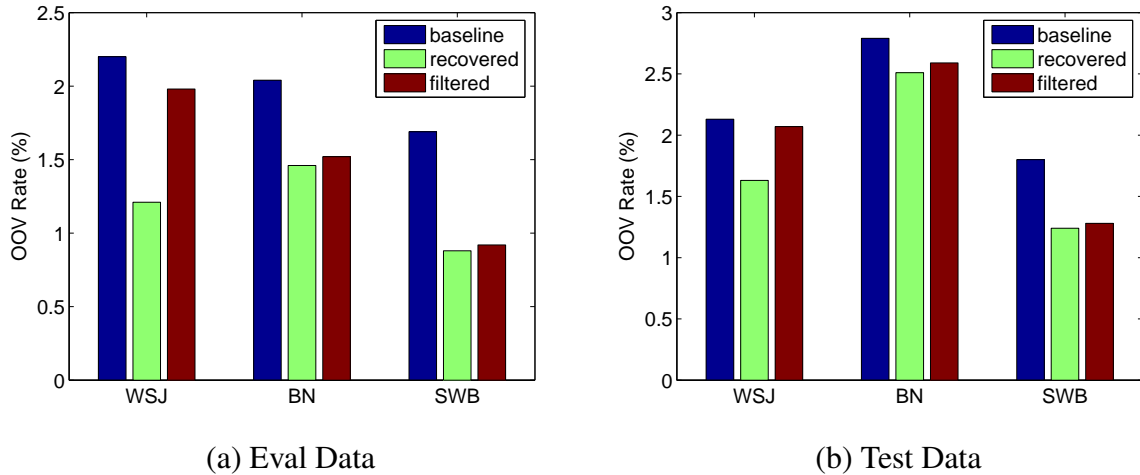


Figure 6.16: The OOV rate of the recovered and filtered expanded vocabularies in the Eval and Test data.

The OOV rate of the recovered and filtered expanded vocabularies in the Eval and Test data is given in Figure 6.16. It can be seen that as more words were incorporated, both expanded vocabularies have lower OOV rate than the baseline vocabulary. Between the two expanded vocabularies, the recovered expanded vocabulary has slightly lower OOV rate than the filtered expanded vocabulary in the BN and SWB tasks. But in the WSJ task, the recovered expanded vocabulary has much lower OOV rate than the filtered expanded vocabulary.

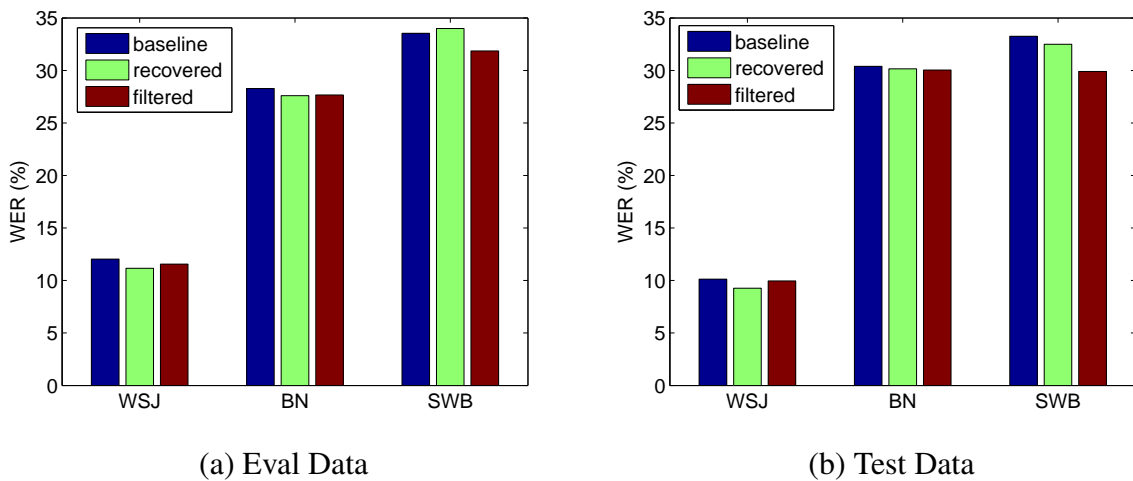


Figure 6.17: The recognition performance when using the recovered and filtered expanded vocabularies on the Eval and Test data.

Figure 6.17 provides the recognition performance when using the recovered and filtered expanded vocabularies during decoding. We can find that ASR systems using the expanded vocabularies usually have lower WER than systems using the baseline vocabulary. The two expanded

vocabularies performed similarly in the WSJ and BN tasks, while systems using the filtered expanded vocabulary is much better than systems using the recovered expanded vocabulary in the SWB task. Although the recovered expanded vocabulary had lower OOV rate than the filter expanded vocabulary, ASR systems built with the recovered vocabulary did not have a lower WER. This may be because the language scores learned for recovered OOV words were still not as good as language model scores directly learned from a text corpus.

From Figure 6.18, we can find how many learned OOV words were recognized when using the recovered and filtered expanded vocabularies. Here, we counted when expanding vocabulary with learned OOV words, how many of those OOV words were recognized in the second pass decoding of the Eval data and in the first pass decoding of the Test data. It can be seen that systems using the recovered expanded vocabulary usually have a higher recognition rate than systems using the filtered expanded vocabulary on the Eval data. However, on the Test data, systems using the filtered expanded vocabulary perform better. Again, this is probably because the language model scores learned for OOV words in the recovered system are not optimal. But we should also notice that we did not use any extra resources when building ASR systems with the recovered expanded vocabulary.

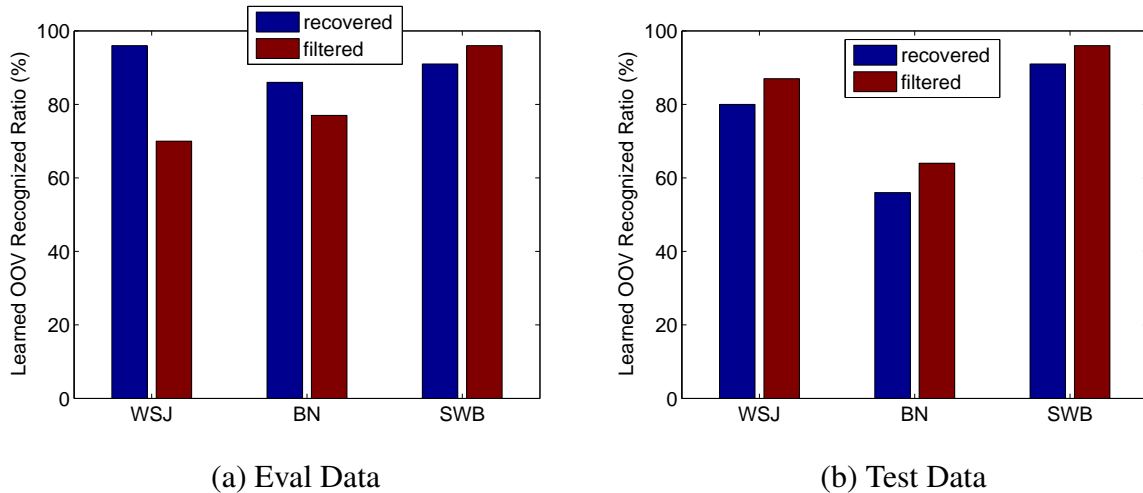


Figure 6.18: The OOV words recognized ratio when using the recovered and filtered expanded vocabularies on the Eval and Test data.

### 6.5.3 The results of recovering recurrent OOV words

When evaluating the performance of recovering recurrent OOV words, we only worked on identified OOV words with more than one instances, since we needed those multiple instances to estimate a better written form and better language model scores for the word. The accuracy of identified OOV clusters will affect our OOV word recovery performance. For example, if many instances of different OOV words were unfortunately identified as the same OOV word, then because of the noise introduced by clustering errors, the recovery performance might even be hurt. As discussed in Chapter 5, our OOV word clustering results were very accurate, there were only



a few OOV candidates which belong to different OOV words were clustered into the same group. Therefore, the clustering results were good enough for our OOV word recovery experiments.

To estimate the written form of recurrent OOV words, we implemented a ROVER-like combination procedure, which combines multiple pronunciations from the multiple instances of the same OOV word to build a better phonetic representation for the word. Then, the better joint-sequence model trained with recognition results of the training speech was applied to perform the P2G conversion. To compare those results, we also directly performed the P2G conversion on the decoded pronunciation of each OOV candidate, which is considered as a baseline. The pronunciation accuracy of recurrent OOV words before and after combination is given in Table 6.20. It can be seen that PA is significantly increased after combining the multiple phonetic representations of an OOV word in the WSJ and BN tasks. In the SWB task, multiple instances of the same OOV words were usually decoded with the same pronunciation. As a result, we did not observe much improvement from combination.

Table 6.20: The pronunciation accuracy of recurrent OOV words before and after combination.

	WSJ	BN	SWB
Before combination	56.1%	34.9%	72.8%
After combination	62.9%	42.2%	72.8%

We have demonstrated that after combining multiple pronunciations of the same OOV word, we could have a better phonetic representation for the word. Now, we evaluate how many OOV words have the correct written form after the P2G conversion. The recovery rate of recovered recurrent OOV words with and without combination is given in Table 6.21. We can see that RR also greatly increases when perform the P2G conversion on the combined pronunciation of a recurrent OOV word in the WSJ and BN tasks. As there is no improvement on PA in the SWB task, RR also remains unchanged.

Table 6.21: The recovery rate of recovered recurrent OOV words with and without combination.

	WSJ	BN	SWB
Without combination	25.6%	17.5%	69.3%
With combination	35.8%	30.6%	69.3%

Table 6.22: The POS label accuracy of recurrent OOV words before and after combination.

	WSJ	BN	SWB
Baseline	86.4%	66.7%	62.1%
Combined	91.5%	85.5%	70.7%

Besides combining the pronunciations of multiple instances of an OOV word to obtain a better phonetic representation, we could also combine the POS labels from multiple instances of the word to estimate a more accurate POS label. Table 6.22 presents the POS label accuracy

of recurrent OOV words before and after combination. It can be seen that the POS accuracy is improved after combining the POS labels from multiple instances of an OOV word.

Finally, let us compare the recognition performance of ASR systems built from the baseline recovered expanded vocabulary and the recurrent recovered expanded vocabulary. From Figure 6.19, we can find that the difference between those two systems is very small. The improvement of better phonetic representation, better written form and more accurate POS label does not produce a lower WER. One possible explanation is there are only very few recurrent OOV words in the Eval and Test data. Even we achieved good results on recurrent OOV words, they are too few to make contributions to the WER of the whole Eval and Test data. The performance may be improved if we tested on a datasets with more recurrent OOV words.

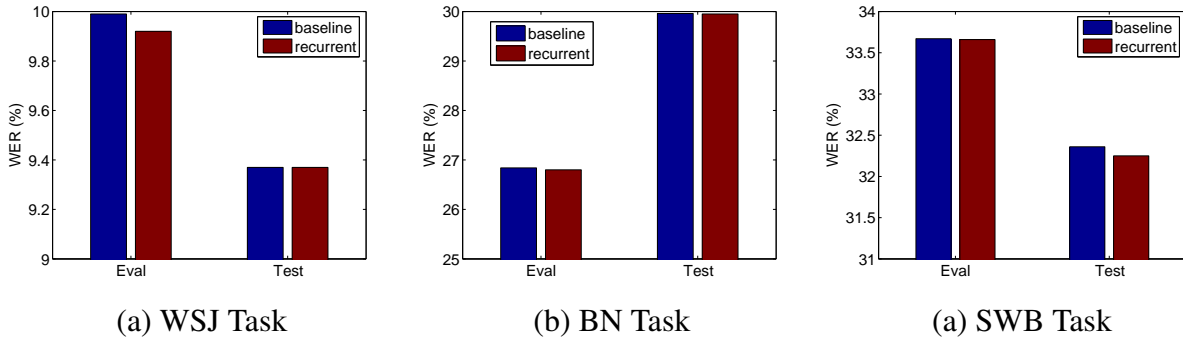


Figure 6.19: The recognition performance of ASR systems built from baseline recovered expanded vocabulary and recurrent recovered expanded vocabulary.

## 6.6 Summary

In this chapter, we investigated various techniques for OOV word recovery. First, the conventional P2G conversion was applied to estimate the written form of the detected OOV words. As we found many detected OOV words' pronunciations were incorrect, we proposed to train a better P2G model by learning from the recognition errors of the training speech. Then, we explored to estimate the language model scores of an OOV word from IV words in the same syntactic category. We also studied an alternative OOV word recovery approach when extra language resources are available. At last, we showed that the multiple instances of recurrent OOV words can be used to further improve the OOV word recovery performance. To summary, we recovered more than 40% OOV words after integrating detected OOV words into the recognizer's lexicon. Then, our system recognized more than 90% recovered OOV words in the Eval data and up to 70% recovered OOV words in the Test data. Overall, our OOV word learning framework successfully learned up to 40% OOV words from the Eval data.

# Chapter 7

## Conclusion and Future Work

In this thesis, we investigated a fundamental problem in speech recognition - how to automatically learn OOV words in an ASR system. To learn an OOV word or to convert an OOV word into an IV word, a recognizer has to first detect the presence of the OOV word in the testing speech, and then incorporate the OOV word into its lexicon and language model. Therefore, we built an OOV word learning framework, which can automatically detect, cluster and recover OOV words in an ASR system. From the experimental results, we found that about 40% OOV words were successfully converted into IV words through the proposed OOV word learning framework.

### 7.1 Summary of results and contributions

Our proposed OOV word learning framework consists of three main components: the OOV word detection component, the OOV word clustering component and the OOV word recovery component. In the following paragraph, the results and contributions in each component are summarized. Furthermore, the results on different datasets are also summarized and analyzed.

#### OOV word detection

In the OOV word detection component, we tried to detect the presence of OOV words in the testing speech. We adopted the hybrid system, which applies a hybrid lexicon and language model during decoding, to explicitly represent OOV words using sub-lexical units. Since two different training schemes had been individually studied for building the hybrid system, we compared the OOV word detection performance using the hierarchical and flat hybrid systems. We found that those two hybrid systems had similar OOV word detection performance if they were built on simple sub-lexical units, such as phones. However, the flat hybrid system was significantly better than the hierarchical hybrid system when more complicated sub-lexical units were used, such as the syllable, subword or grapheme units. Furthermore, we also compared the effectiveness of modeling OOV words using different types of sub-lexical units. Our experimental results showed that the syllable, subword and grapheme units performed much better than the simple phone units, because longer context histories were embedded in more complex sub-lexical units.

In this part of thesis, we also studied to remove incorrect detections from the hybrid decoding output using the OOV word classifier. We collected the acoustic, lexical, lattice and contextual

features from the OOV word detection result of the development data, and built a LogitBoost classifier with feature selection. Then, the LogitBoost classifier was applied to the OOV word detection hypotheses of the evaluation data. The classification result indicated that the OOV word classifier effectively reduced false alarm errors when the hybrid system reported many incorrect detections. As a result, the OOV word detection performance was improved. Furthermore, better classifiers could be trained if we collected more OOV word detection examples from the hybrid decoding result of the training speech.

At last, we introduced two system combination techniques to further improve the OOV word detection performance. In one approach, multiple systems' outputs were aligned, combined and re-scored to produce a better hybrid decoding result. The word frequency and confidence score of individual hypothesis were applied for re-scoring. As we found different types of sub-lexical units had their own advantages and problems, in another combination approach, multiple types of sub-lexical units including the syllable, subword and grapheme units were simultaneously used in one hybrid system, so that different types of sub-lexical units can complement each other. From the experimental results, we found both system combination techniques performed better than individual systems which were built using one type of sub-lexical units. Between these two combination techniques, the one combining multiples systems' outputs performed better in the BN and SWB tasks, while the other one performed better in the WSJ task. Overall, we can detect more than 70% OOV words with more than 60% accuracy in the OOV word detection component

## **OOV word clustering**

An OOV word can appear more than once in a conversation or over a period of time. Such multiple instances of the same OOV word provide valuable information for estimating the pronunciation, part-of-speech (POS) tag or language model scores of the word. In the OOV word clustering component, we therefore studied how to identify recurrent OOV words through a bottom-up clustering process. Initially, each OOV candidate in the OOV word detection output was considered as a single cluster. Then, in each iteration, two clusters with the smallest distance were merged. This clustering procedure ended when the distance between any two clusters was larger than a threshold. The distance between two clusters was calculated as the average of the pairwise distances between OOV candidates in each cluster. And to measure the distance between OOV candidates, the phonetic, acoustic and contextual features of each OOV candidate were collected. Specifically, the phonetic distance measured whether two OOV candidates had the same pronunciation or not, the acoustic distance compared their acoustic representations, while the contextual distance measured whether two OOV candidates appeared in the same context and whether they were about the same topic. Our experimental results presented that the phonetic feature was more effective than the acoustic and contextual features for detecting the recurrence of OOV words, but the best performance was achieved when combining all features. As the goal of finding recurrent OOV words is to utilize the multiple instances of the same OOV word to learn a better pronunciation or POS tag of the word, in this chapter, we also showed that the OOV word clustering result was good enough for further process.

## OOV word recovery

After detecting and clustering OOV words in the testing speech, we worked on recovering OOV words - converting OOV words into IV words of an ASR system in the OOV word recovery component. To finally recover an OOV word, we had to integrate the OOV word into the recognizer's lexicon and language model, which means we would need to estimate the written form and language model scores of the word. We first applied the conventional phoneme-to-grapheme (P2G) conversion to estimate the written form of an OOV word from its decoded phonetic representation. However, the experimental results showed that we could only recover a small portion of the detected OOV words. By carefully examined the detected OOV words and the P2G conversion result, we found that the decoded pronunciations of many OOV words incorporated recognition errors. But the conventional P2G model could not handle incorrect phone sequences, as it was trained only from the alignments between correct phone and letter sequences. Therefore, we investigated to build a better P2G model by also learning from the hybrid decoding result of the training speech. The evaluation results indicated that significantly more correct written forms were estimated using the better P2G model. As a result, more improvement on the WER was observed. To estimate the language model scores of an OOV word, we utilize the POS tags of the OOV word and its surrounding IV words. Precisely, we first trained a POS tag class-based language model using the text training data. Then given the POS tags of the OOV word and its surrounding words, the  $n$ -gram scores of an OOV word could be estimated from corresponding  $n$ -grams in the class-based language model. As an OOV word may appear in a different context when encountered by the recognizer in the future, we also proposed possible context for an OOV word by substituting its surrounding IV words with other semantic similar IV words. Our recognition results indicated that we were able to successfully estimate language model scores for detected OOV words. Overall, we converted more than 40% OOV words into IV words after integrating recovered OOV words into the recognizer's lexicon. And we could recognizer more than 90% learned new words in the Eval data and up to 70% learned new words in the Test data.

In some situation, ASR systems cannot use a very large vocabulary, although there are normally more data available for building a larger lexicon or a larger language model. Therefore, in the OOV word recovery component, we also studied an alternative OOV word recovery approach, when a larger lexicon and a larger text corpus are available. Here, we still worked on the OOV word detection output. For each detected OOV word, we searched for words in the large lexicon. If the phonetic distance between the OOV word and the word in the large lexicon was small enough, or say if those two words sounded very similar, the word in the large lexicon may actually be the OOV word we detected in the testing speech. We then added that word into our recognition vocabulary. After processed all detected OOV words, we trained a new language model with the new expanded vocabulary using the extra text corpus. The experimental results showed when only increased the vocabulary size by 5%, we could learn about 40% of OOV words in the testing speech. And the WER on the evaluation and test data was substantially improved.

As we mentioned previously, the multiple instances of recurrent OOV words incorporated valuable information for estimating the written form and language model scores of an OOV word, in this part, we also investigated how to improve the OOV word recovery performance on recurrent OOV words. We first introduced a ROVER-like combination technique to build a

better phonetic representation for the recurrent OOV word from the decoded pronunciations of its multiple instances. We then combined the POS tags of the multiple instances of an OOV word to estimate a more accurate POS tag for the word. We also utilized the different context a recurrent OOV word appeared to construct exact  $n$ -grams for recurrent OOV words. We could also use those context to estimate new  $n$ -gram scores for recurrent OOV words without adding too many irrelevant entries into the language model. From the experimental results, we found about 7% more OOV words obtained the correct phonetic representation by combining the pronunciations of their multiple instances. And because of that, about 10% more OOV words ended up with the correct written form. But overall, only a few more OOV words were recognized in the evaluation and test data.

### 7.1.1 Comparing the results on different datasets

We tested our OOV word learning framework on three different datasets. It is interesting to see the different performance of each component on different data, from which we can understand the advantage and weakness of our system better.

First, as the recording condition and speaking style of three datasets is quite different, the WER of the baseline word recognition system is also in different ranges. From Table 4.3, we can find that we have the best recognition accuracy on the WSJ task, which has the optimal condition – prepared read speech in a very quiet environment. The BN speech is extracted from broadcast news, which has fairly good audio quality. But there is occasionally background speech or music, which can degrade the recognition performance. Furthermore, in BN, the speech is more free style than WSJ, that’s another factor to make it harder to recognize. At last, it is not surprise to find that the WER on the SWB data is the highest. The SWB speech is 8k Hz telephone speech, which has much worse audio quality compared to the WSJ and BN data. Moreover, the data is spontaneous human to human conversation containing a large number of natural speech phenomena, such as deletions, repetitions, etc., which can cause a big problem to the ASR system. The findings here are basically the same as those reported by other researchers on the same datasets indicating that we built a reasonable recognizer for our OOV word learning framework.

The OOV word detection performance does not always align with the recognition performance of different tasks. Again, we have pretty good OOV word detection performance in the WSJ task, where we can detect more than 70% OOV words with more than 60% precision. However, different from the recognition performance, we have better OOV word detection performance in the SWB task than in the BN task. One observation is that many utterances are very long in the BN task. As a result, more than one OOV word can appear in the same utterance or even appear in a row, which may confuse our OOV word detection component. Another reason comes from the training data. We found all OOV tokens are in the training data of the SWB task, but that is not true in the BN task. What we can learn from these experiments are: 1) the OOV word detection component works much better if the OOV tokens have been seen in the training, from which we can know how those OOV words sound like and where they usually appear in an utterance. But in this case, those OOV tokens are not “true” OOV words, they are just low frequency words which are excluded from the recognizer’s vocabulary. 2) Our system does not work well when multiple OOV words appear consecutively in speech. It normally can detect the

presence of OOV words, but cannot tell how many OOV words are there. Although we already incorporated the word start and word end symbols into the sub-lexical units, they only give us information about which sub-lexical units appear more often at the word boundaries. To solve this problem, we can investigate to use more acoustic evidence to cut decoded sub-lexical sequences into multiple OOV words. For instance, we can try to find silence within the OOV region in the testing speech and use that to identify word boundaries. Another place our OOV word detector may fail is when trying to identify very short OOV words, such as words with only two or three phones. Those OOV words are hard to detect and recognize because of lacking enough acoustic evidence. They are frequently to misrecognize with short in-vocabulary words, since they may sound very similar. But we probably do not need to worry about this too much, because there are normally not many such short OOV words in a language.

The OOV word clustering performance on different datasets has similar picture as in the OOV word detection experiments, where we have better performance in the WSJ and SWB tasks than in the BN task. Particularly, when only using the phonetic feature to measure the similarity between OOV candidates, the OOV word clustering performance in the BN task is much worse than those in the WSJ and SWB tasks. First, as the OOV word detection result is less accurate in the BN task, more IV words were incorrectly detected as OOV words, which can affect the clustering accuracy. Second, we found that the decoded pronunciation of OOV words is also more noisy in the BN task. Contrarily, the decoded pronunciations of multiple instances of the same OOV word are more or less the same in the WSJ and SWB tasks, no matter whether those multiple instances appear in the same context or not. We can also learn this from Figure 6.6. It can be seen that the pronunciation accuracy (PA) is much higher in the WSJ and SWB tasks, especially, in the SWB task, where PA is up to 50%. But in the BN task, PA is only about 20%, most decoded pronunciations of OOV words are wrong. And because of that, the phonetic feature in the BN task is not as reliable as that in the WSJ and SWB tasks. From our experiments we also notice that the acoustic feature only works well in the WSJ task. Certainly, as we have better recognition result in WSJ, the OOV segment timing information is more accurate. But that should not give such a big impact on the clustering performance. One possible reason is, in WSJ, we usually have many utterances (more than 20) from the same speaker, which may contain the same OOV word in multiple places. Then, when comparing the acoustic distance between OOV candidates, the distance will be smaller if the two OOV candidates are from the speech of the same speaker. Moreover, the WSJ speech is quite clean, without pollution from the background noise. On the other hand, in the BN and SWB tasks, even if two OOV candidates correspond to the same OOV word, their acoustic distance may still be large, since they usually from different speakers and more probably influenced by noise. Therefore, we need something better than the simple posterior feature representation for the acoustic signal.

The phoneme-to-grapheme (P2G) conversion performance in different tasks is similar to what we learned from the pronunciation accuracy, except that we only obtained very small improvement by learning from the recognition results of training speech when building the P2G model in the WSJ task. This may because the recognizer makes much less recognition errors on the training speech in the WSJ task than in the BN and SWB tasks, as the recognizer accuracy is more than 90% in the WSJ task. As a result, the benefit of learning from errors is small in WSJ. In terms of the part-of-speech (POS) label accuracy, we have better results in the WSJ and BN tasks than in the SWB task. This is simply because the SWB sentences are from human to hu-

man conversation, which is usually not grammatical. Therefore, POS tagging on the SWB data is much harder than on the WSJ and BN data. For OOV word learning, again, we have better results in the WSJ and SWB tasks than in the BN task, which is probably a result of accumulating the better OOV word detection, clustering and P2G conversion results. One interesting finding is from the OOV word learning performance on the testing data as shown in Figure 6.11. It can be seen that we recognized much more newly learned OOV words in the WSJ task than in the BN and SWB tasks. As we mentioned in previous sections, the estimated language model scores for detected OOV words are not very accurate, especially the new contexts for OOV words are mostly irrelevant. Then when recognizing unseen OOV words in the testing speech, most of time, the unigram scores of OOV words will be applied. In the case, the recognition depends more on the acoustic model. And a more accurate acoustic will definitely produce better recognition result. Therefore, we have the best recognition performance on unseen OOV words in the WSJ task, which has a very good acoustic model. From this, we can also learn that our OOV word learning framework will be more appreciated when having a strong acoustic model in the recognizer.

## 7.2 Future work

A few interesting topics were not addressed in this thesis but worth for future investigations.

### **Contextual evidence for OOV word detection**

In daily communications, most of time, we can successfully identify OOV words in human speech. When we hear an OOV word, besides the uncertainty about individual OOV words, we often rely on broad contextual evidence especially syntactic and semantic evidence to verify whether what we hear is reasonable in a certain context. Therefore, we may be able to improve the OOV word detection performance by utilizing high level syntactic and semantic evidences. Although we applied certain contextual features when building the OOV word classifier in the OOV word detection component, longer range dependencies between words across sentences and even paragraphs are still not explored. Besides the OOV word classifier, we can also try to find other ways of applying high level syntactic and semantic evidences. For example, we can re-score the lattices by using a factored language model (FLM) built from various features to find a better recognition hypothesis after the first pass hybrid decoding.

### **Batch learning or hill climbing learning?**

In this thesis, we studied how to learn OOV words in an ASR system in the batch mode. We did not investigate how often should we integrate newly learned OOV words into the recognizer's lexicon and language model. If we expand the recognition vocabulary very often, we always have a low OOV rate on the testing speech. We may then have a better recognition accuracy and detect more OOV words. On the other hand, we will not be able to obtain multiple instances of an OOV word, if we update the recognition vocabulary too often. In this case, although we can detect those OOV words, we may not be able to correctly recover their written form and language



model scores. This problem is more important for a commercial system, where certain events may emerge and suddenly generate high demand OOV words.

### **OOV word learning curve**

When we learn new words in our daily life, there is usually a learning curve to successfully learn the word. We will probably not learn a new word after the first time we encounter the word. Our brain needs more examples of the word and usually more time to understand and memorize the word. Similarly, to an ASR system, it may not be able to successfully learn an OOV word when it detects the OOV word at the first time. Maybe the pronunciation is not correct, or the written form can be wrong or the estimated language model scores are not good enough. When the ASR system encounters more and more instances of the same OOV word, how does it update its existing information about the word? Should it keep its past understandings of the word or trust new examples more? Furthermore, maybe both the old and new information of the word is correct. For instance, the word may have multiple pronunciations. It is interesting to see how the ASR system learns multiple phonetic representations of one word.

### **OOV word learning in a dialog system**

Another interesting problem is how to integrate our OOV word learning framework into a dialog system. A dialog system usually operates on a small closed vocabulary, which makes it more often to encounter OOV words than large vocabulary commercial ASR systems. Therefore, a dialog system with the ability of automatically learning new words will be extremely valuable. It may be easier and more accurate to learn OOV words in a dialog system, as we can rely on the interactions between the user and system. If the system suspects there may be an OOV word in the input speech, it can directly ask confirmation from user. The system can even ask user how to spell the word, what is the meaning of the word and if there is a synonym of the word, etc. From those answers, we can even learn syntactic and semantic properties of the word. There are also challenges for OOV word learning in a dialog system. For example, once the system detect there is an OOV word, how to present the OOV word to user? Should the system synthesize the word or directly replay a segment of the input speech? Or maybe there is a even easier way by simply notifying user there is an OOV word. Some researcher found user can usually guess which word in an utterance a dialog system may not know.



# Bibliography

- CMU SPHINX: the open source toolkit for speech recognition. URL <http://cmusphinx.sourceforge.net/>.
- G. Aradilla, J. Vepa, and H. Bourlard. Using posterior-based features in template matching for speech recognition. In *Proc. Interspeech-2006*, 2006.
- K. Audhkhasi and A. Verma. Keyword search using modified minimum edit distance measure. In *Proc. ICASSP-2007*, volume 4, pages 929–932, 2007.
- L. R. Bahl, S. Das, P. V. deSouza, M. Epstein, R. L. Mercer, B. Merialdo, D. Nahamoo, M. A. Picheny, and J. Powell. Automatic phonetic baseform determination. In *Proc. ICASSP-1991*, volume 1, pages 173–176, 1991.
- L. E. Baum and J. A. Egon. An inequality with applications to statistical estimation for probabilistic functions of a Markov process and to a model for ecology. *Bull. Amer. Meteorol. Soc.*, 73:360–363, 1967.
- L. E. Baum and T. Petrie. Statistical inference for probabilistic functions of finite state Markov chains. *Annals of Mathematical Statistics*, 37:1554 – 1563, 1966.
- L. E. Baum, T. Petrie, G. Soules, and N. Weiss. A maximization technique occurring in the statistical analysis of probabilistic functions of Markov chains. *Ann. Math. Stat.*, 41(1):164–171, 1970.
- I. Bazzi and J. Glass. Modeling out-of-vocabulary words for robust speech recognition. In *Proc. ICSLP-2000*, volume 1, pages 401–404, 2000.
- M. Bisani and H. Ney. Open vocabulary speech recognition with flat hybrid models. In *Proc. Interspeech-2005*, pages 725–728, 2005.
- M. Bisani and H. Ney. Joint-sequence models for grapheme-to-phoneme conversion. *Speech Communication*, 50(5):434–451, 2008.
- A. W. Black, P. Taylor, and R. Caley. *The Festival Speech Synthesis System*. University of Edinburgh, 1997.
- L. Burget, P. Schwarz, P. Matejka, H. Hermansky, and J. Cernocky. Combining of strongly and weakly constrained recognizers for reliable detection of OOVs. In *Proc. ICASSP-2008*, pages 4081–4084, 2008.
- S. F. Chen. Conditional and joint models of grapheme-to-phoneme conversion. In *Proc. Eurospeech-2003*, pages 2033–2036, 2003.
- S. F. Chen and J. Goodman. An empirical study of smoothing techniques for language modeling.

- Technical report tr-10-98, Center for Research in Computing Technology (Harvard University), August 1998.
- S. B. Davis and P. Mermelstein. Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences. *IEEE Transactions on Acoustics, Speech and Signal Processing*, 28(4):357 – 366, 1980.
- A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *J. Roy. Stat. Soc.*, 39(1):1–38, 1977.
- C. Fellbaum. *WordNet: An Electronic Lexical Database*. MIT Press, 1998.
- J. Fiscus, J. Garofolo, M. Przybocky, W. Fisher, and D. Pallett. *1997 English Broadcast News Speech (HUB4)*. Linguistic Data Consortium, 1998.
- J. G. Fiscus. A post-processing system to yield reduced word error rates: Recognizer output voting error reduction (ROVER). In *Proc. ASRU-1997*, pages 347–354, 1997.
- M. J. F. Gales. Maximum likelihood linear transformations for hmm-based speech recognition. *Computer Speech and Language*, 12:75–98, 1998.
- L. Galescu. Recognition of out-of-vocabulary words with sub-lexical language models. In *Proc. Eurospeech-2003*, pages 249–252, 2003.
- J. Garofalo, D. Graff, D. Paul, and D. Pallett. *CSR-I (WSJ0) Complete*. Linguistic Data Consortium, 1993.
- J. Garofalo, D. Graff, D. Paul, and D. Pallett. *CSR-II (WSJ1) Complete*. Linguistic Data Consortium, 1994.
- J. J. Godfrey and E. Holliman. *Switchboard-1 Release 2*. Linguistic Data Consortium, 1997.
- I. Good. The population frequencies of species and the estimation of population parameters. *Biometrika*, 40:237–264, 1953.
- D. Graff, R. Rosenfeld, and D. Paul. *CSR-III Text*. Linguistic Data Consortium, 1995.
- D. Graff, J. Garofolo, J. Fiscus, W. Fisher, and D. Pallett. *1996 English Broadcast News Speech (HUB4)*. Linguistic Data Consortium, 1997.
- R. Haeb-Umbach and H. Ney. Linear discriminant analysis for improved large vocabulary continuous speech recognition. volume 1, pages 13–16, 1992.
- M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten. The weka data mining software: An update. 11(1).
- M. Hannemann, S. Kombrink, M. Karafiat, and L. Burget. Similarity scoring for recognizing repeated out-of-vocabulary words. In *Proc. Interspeech-2010*, pages 897–900, 2010.
- S. Hayamizu, K. Itou, and K. Tanaka. Detection of unknown words in large vocabulary speech recognition. In *Proc. Eurospeech*, pages 2113–2116, 1993.
- T. Hazen, W. Shen, and C. White. Query-by-example spoken term detection using phonetic posteriorgram templates. In *Proc. ASRU-2009*, pages 412–426, 2009.
- X. Huang, A. Acero, A. Acero, and H. W. Hon. *Spoken Language Processing: A Guide to Theory, Algorithm and System Development*. Prentice Hall, 2001.

- X. D. Huang. *Semi-Continuous Hidden Markov Models for Speech Recognition*. Phd thesis, University of Edinburgh, 1989.
- L. Hubert and P. Arabie. Comparing partitions. *Journal of Classification*, 2(1):193–218, 1985.
- D. Huggins-Daines. *An Architecture for Scalable, Universal Speech Recognition*. Phd thesis, Carnegie Mellon University, 2011.
- M. Y. Hwang. *Subphonetic Acoustic Modeling for Speaker-Independent Continuous Speech Recognition*. Phd thesis, Carnegie Mellon University, 1993.
- S. M. Katz. Estimation of probabilities from sparse data for the language model component of a speech recognizer. *IEEE Transaction on Acoustic, Speech and Signal Processing*, 35(3):400–401, 1978.
- D. Klakow, G. Rose, and X. Aubert. OOV-detection in large vocabulary system using automatically defined word-fragments as fillers. In *Proc. Eurospeech-1999*, pages 49–52, 1999.
- R. Kneser and H. Ney. Improved backing-off for M-gram language modeling. In *Proc. ICASSP-1995*, volume 1, pages 181–184, 1995.
- S. Kombrink, L. Burget, P. Matejka, M. Karafiat, and H. Hermansky. Posterior-based out-of-vocabulary word detection in telephone speech. In *Proc. Interspeech-2009*, pages 80–83, 2009.
- B. Lecouteux, G. Linares, and B. Favre. Combined low level and high level features for out-of-vocabulary word detection. In *Proc. Interspeech-2009*, pages 1187–1190, 2009.
- K. F. Lee, H. W. Hon, and R. Reddy. An overview of the SPHINX speech recognition system. *IEEE Transactions on Acoustics, Speech and Signal Processing*, 38(1):35–45, 1990.
- H. Lin, J. Bilmes, D. Vergyri, and K. Kirchhoff. OOV detection by joint word/phone lattice alignment. In *Proc. ASRU-2007*, pages 478–483, 2007.
- B. Maison. Automatic baseform generation from acoustic data. In *Proc. Eurospeech-2003*, pages 2545–2548, 2003.
- M. P. Marcus, M. Marcinkiewicz, and B. Santorini. Building a large annotated corpus of English: the penn treebank. *Computational Linguistics*, 19(2):313–330, 1993.
- G. A. Miller. Wordnet: A lexical database for english. *Communications of the ACM*, 38(11):39–41, 1995.
- S. Ortman and H. Ney. A word graph algorithms for large vocabulary continuous speech recognition. *Computer Speech and Language*, 11:43–72, 1997.
- C. Parada, M. Dredze, D. Filimonov, and F. Jelinek. Contextual information improves OOV detection in speech. In *Proc. HLT-NAACL-2010*, pages 216–224, 2010a.
- C. Parada, A. Sethy, M. Dredze, and F. Jelinek. A spoken term detection framework for recovering out-of-vocabulary words using the web. In *Proc. Interspeech-2010*, pages 1269–1272, 2010b.
- T. Pedersen, S. Patwardhan, and J. Michelizzi. Wordnet::similarity: measuring the relatedness of concepts. In *Proc. HLT-NAACL-2004*, pages 38–41, 2004.
- H. Printz and P. Olsen. Theory and practice of acoustic confusability. *Computer Speech and*

- Language*, 16:131–164, 2002.
- M. Pucher, A. Turk, J. Ajmera, and N. Fecher. Phonetic distance measures for speech recognition vocabulary and grammar optimization. In *Proc. the 3rd Congress of the Alps Adria Acoustics Association*, 2007.
- L. Qin and A. Rudnicky. The effect of lattice pruning on MMIE training. In *Proc. ICASSP-2010*, pages 4898–4901, 2010a.
- L. Qin and A. Rudnicky. Implementing and improving MMIE training in SphinxTrain. In *CMU SPHINX Users and Developers Workshop*, 2010b.
- L. Qin and A. Rudnicky. OOV word detection using hybrid models with mixed types of fragments. In *Proc. Interspeech-2012*, 2012.
- L. Qin and A. Rudnicky. Finding recurrent out-of-vocabulary words. In *Proc. Interspeech-2013*, 2013.
- L. Qin, M. Sun, and A. Rudnicky. OOV detection and recovery using hybrid models with different fragments. In *Proc. Interspeech-2011*, pages 1913–1916, 2011.
- L. Qin, M. Sun, and A. Rudnicky. System combination for out-of-vocabulary word detection. In *Proc. ICASSP-2012*, 2012.
- L. Rabiner and B. H. Juang. *Fundamentals of Speech Recognition*. Prentice Hall, 1993.
- W. M. Rand. Objective criteria for the evaluation of clustering methods. *Journal of the American Statistical Association*, 66(336):846–850, 1971.
- A. Rastrow, A. Sethy, and B. Ramabhadran. A new method for OOV detection using hybrid word/fragment system. pages 3953–3956, 2009a.
- A. Rastrow, A. Sethy, B. Ramabhadran, and F. Jelinek. Towards using hybrid, word, and fragment units for vocabulary independent LVCSR systems. In *Proc. Interspeech-2009*, pages 1931–1934, 2009b.
- R. Rosenfeld. Optimizing lexical and N-gram coverage via judicious use of linguistic data. In *Proc. Eurospeech-1995*, pages 1763–1766, 1995.
- A. Rudnicky. The CMU pronunciation dictionary, release 0.7a, 2007. URL <http://www.speech.cs.cmu.edu/cgi-bin/cmudict>.
- H. Sakoe and S. Chiba. Dynamic programming algorithm optimization for spoken word recognition. *IEEE Trans. ASSP*, 26(1):43–49, 1978.
- T. Schaaf. Detection of OOV words using generalized word models and a semantic class language model. In *Proc. Eurospeech-2001*, pages 2581–2584, 2001.
- M. A. B. Shaik, A. E.-D. Mousa, R. Schluter, and H. Ney. Hybrid language models using mixed types of sub-lexical units for open vocabulary German LVCSR. In *Proc. Interspeech-2011*, pages 1441–1444, 2011.
- F. Stouten, D. Fohr, and I. Illina. Detection of OOV words by combining acoustic confidence measures with linguistic features. In *Proc. ASRU-2009*, pages 371–375, 2009.
- H. Sun, G. Zhang, F. Zheng, and M. Xu. Using word confidence measure for OOV words detection in a spontaneous spoken dialog system. In *Proc. Eurospeech-2003*, pages 2713–

2716, 2003.

- K. Toutanova and C. D. Manning. Enriching the knowledge sources used in a maximum entropy part-of-speech tagger. In *Proc. EMNLP/VLC-2000*, pages 63–70, 2000.
- K. Toutanova, D. Klein, C. Manning, and Y. Singer. Feature-rich part-of-speech tagging with a cyclic dependency network. In *HLT-NAACL-2003*, pages 252–259, 2003.
- K. Vertanen. Combining open vocabulary recognition and word confusion networks. In *Proc. ICASSP-2008*, pages 4325–4328, 2008.
- N. X. Vinh and J. Epps. Information theoretic measures for clusterings comparison: Variants, properties, normalization and correction for chance. *Journal of Machine Learning Research*, 11:2837–2854, 2010.
- T. Vintsyuk. Speech discrimination by dynamic programming. *Kibernetika*, 4:81–88, 1968.
- R. Wagner and M. Fischer. The string-to-string correction problem. *J. ACM*, 21(1):168–173, 1974.
- F. Wessel, R. Schluter, K. Macherey, and H. Ney. Confidence measures for large vocabulary continuous speech recognition. *IEEE Transactions on Speech and Audio Processing*, 9(3): 288–298, 2001.
- C. White, G. zweig, L. Burget, P. Schwarz, and H. Hermansky. Confidence estimation, OOV detection and language ID using phone-to-word transduction and phone-level alignments. In *Proc. ICASSP-2008*, pages 4085–4088, 2008.
- A. Yazgan and M. Saraclar. Hybrid language models for out of vocabulary word detection in large vocabulary conversational speech recognition. In *Proc. ICASSP-2004*, volume 1, pages 745–748, 2004.
- Y. Zhang and J. Glass. Unsupervised spoken keyword spotting via segmental dtw on gaussian posteriorgrams. pages 398–403, 2009.