LEARNING SUB-WORD UNITS AND EXPLOITING CONTEXTUAL INFORMATION FOR OPEN VOCABULARY SPEECH RECOGNITION

by

Maria Carolina Parada

A dissertation submitted to The Johns Hopkins University in conformity with the requirements for the degree of Doctor of Philosophy.

Baltimore, Maryland

June, 2011

Abstract

Large vocabulary continuous speech recognition (LVCSR) systems fail to recognized words beyond their vocabulary, many of which are information rich terms such as named entities, technical terms, or foreign words. Mis-recognizing these Out-of-Vocabulary (OOV) words can have a disproportionate impact in transcript coherence, and cause recognition failures which propagate through pipeline systems, impacting the performance of downstream applications. Ideally, a speech recognition system would be able to recognize arbitrary, even previously unseen, words.

This dissertation presents an approach to recover from failures caused by OOVs by automatically identifying when OOVs are spoken and transcribing them using sub-lexical units. This results in a hybrid word/sub-word system which predicts full-words for in-vocabulary terms and sub-lexical units for OOVs. We first present an approach to model OOVs using sub-lexical units automatically learned from data. The learned units are variable-length phone sequences, which are included in the recognizer's vocabulary and language model. Previous work heuristically creates the sub-word lexicon from phonetic representations of text using simple statistics to select common phone sequences. Instead, we propose a novel unsupervised approach to learn the sub-word lexicon optimized for a given task. This approach employs a log-linear model with overlapping features to learn multi-phone units obtained by segmenting the phonetic representation of a corpus.

OOV Detection is the task of identifying regions in the recognizer's output where out-of-vocabulary words were uttered. The detection of OOV regions is helpful to avoid error propagation to downstream applications such as machine translation, named entity recognition, and spoken document retrieval. We combine the proposed hybrid system with confidence based metrics to improve OOV detection performance. Previous work address OOV detection as a binary classification task, where each region is independently classified using local information. This dissertation treats this problem as a sequence labeling problem, and shows that 1) jointly predicting out-of-vocabulary regions, 2) including contextual information from each region, and 3) learning sub-lexical units optimized for this task, leads to substantial improvements with respect to state-of-the-art systems.

The resulting sub-word representation and OOV detector is helpful to recover the correct spelling of new words, resulting in an open-vocabulary system; and improves performance in downstream applications strongly affected by out-of-vocabulary terms, such as: spoken term detection and named entity recognition in speech.

Readers: Hynek Hermansky (co-advisor) and Mark Dredze (co-advisor)

Committee: Hynek Hermansky, Mark Dredze, Andreas Andreou, Gerald Meyer, and Bhuvana Ramabhadran

Acknowledgments

There are many people I would like to thank for their help and support during my time at CLSP. First and foremost, I was privileged and honored to have Fred Jelinek as my advisor and I am very thankful for knowing him and for all I learned through my interactions with him over four years. He inspired me to tackle important problems, regardless of how difficult they were; to focus on the root of the problem rather than making small additive improvements; and to always keep in mind the big picture. Most importantly, he taught me to always be curious, humble, and to have a unquenchable thirst for learning. I am a better researcher and person because I knew him. He left us only seven months before my defense, but nothing would have been possible without his guidance, support, and inspiration.

Thanks to my collaborator and advisor, Mark Dredze, who took me under his wing and helped me continue with my thesis work. Mark's enthusiasm for machine learning and research is contagious. Without his guidance and support I wouldn't have continued smoothly with my research and finished my thesis as planned.

I am also very grateful to my advisor Hynek Hermansky, a brilliant researcher whose support and humility taught me a lot with his example. I feel incredibly lucky to have worked with these three amazing researchers as my advisors.

Thanks to my committee members for their helpful comments and support: Prof. Andreas Andreou, Prof. Gerald Meyer, and Dr. Bhuvana Ramabhadran.

Thanks to my colleagues and mentors from IBM: Bhuvana Ramabhadran and Abhinav Sethy. I am grateful for the opportunity to work with them during my internship at IBM and our continued collaboration through the rest of my PhD. I learned so much from them and I really enjoyed discussing research problems and collaborating with them.

Thank you to the brilliant and thriving students at CLSP: Erin Fitzgerald, Jason Smith, Nick Andrews, Scott Novotney, Balakrishnan Varadarajan, Sam Thomas, Sivaram Garimella, Sriram Ganapathy, Sridhar Nemala, Keith Kintzley, Mike Carlin, who always took the time to discuss research problems on the whiteboard, and made the CLSP more fun, especially at coffee time. Special thanks to Ariya Rastrow and Markus Dreyer, who always went out of their way to explain things to me and share tools.

Thank you to Denis Filimonov from University of Maryland for collaborating with me and sharing his language modeling and POS tools for some of the experiments presented in Chapter 4.

I am also thankful for our wonderful staff Desiree Cleves and Monique Folk, who always did more than they needed to help the students and kept a very efficient and professional atmosphere at CLSP.

I thank the Center for Excellence for their financial and technical support. It was a privilege to be part of the Center and interact with many of their brilliant researchers. I am also thankful for the Google PhD Fellowship program, which also funded part of my PhD and provided amazing computing resources for my work.

I want to thank my parents and siblings who always supported me, always told me I can do anything I set my mind to, and inspired me to aim higher.

To my husband Jorge, I am certain I would not have been able to do this without your love and support. Thank you for your patience and sacrifices during these five years of working many many long hours and late nights. To our beautiful daughters Diana and Samantha, your eyes and smiles teach me every day what is most important in life, reminding me how lucky I am to have you in my life, and renewing my strength and focus.

Dedication

This thesis is dedicated to my husband Jorge and our beautiful daughters Diana and Samantha

Contents

List of Tables

List of Figures

Chapter 1

Introduction

The goal of automatic speech recognition (ASR) is to transcribe a spoken utterance into the corresponding string of words. State-of-the-art automatic speech recognition systems operate with a large but limited vocabulary, finding the most likely words in this vocabulary for the given acoustic signal. These are known as Large Vocabulary Continuous Speech Recognition (LVCSR) systems, which means that the set of words that can be transcribed belongs to a large but finite set typically ranging between 40,000 to 100,000 words. It is *continuous* because it transcribes a sequence of words that run together naturally, i.e. full utterances, as supposed to *isolated* word recognition. This framework works extremely well since it combines different knowledge sources (acoustic models, language models, pronunciation models) to obtain the most likely spoken word string available in the search space: the set of words which are known to be in the language. In practice the search space is limited to the set of words seen during training.

Recognition Output

“in serbia today president **SLOW BUT I AM A LOST OF ITS** has pledged to the united states that within one week bosnian serb leader **RATHER THAN CARRIAGES** will be effectively removed from power"

Reference

“In Serbia today President **SLOBODAN MILOŠEVIĆ** has pledged to the United States that within one week Bosnian Serb Leader **RADOVAN KARADŽIĆ** will be effectively removed from power.”

Figure 1.1: Recognition output and reference for utterance 19960523_ABC_WNT in English Broadcast News Corpus, 14:57-15:06. Decoded using state-of-the-art IBM’s Attila recognizer with a 83K word vocabulary.

1.1 The Problem

While LVCSR systems produce high quality transcripts, they cannot recognize out of vocabulary (OOV) words. The root of the problem is having the word as the basic lexical unit in the LVCSR system: if a word is not in the system’s vocabulary, there is no way of hypothesizing it, and hence it is guaranteed to be an error. This is known as **the Out-of-Vocabulary problem** in speech recognition.

Figure 1.1 illustrates this problem with an example from the Broadcast News corpus. The utterance was transcribed with an 83,000 word vocabulary system[1] , which is close to state-of-the-art for this domain (Rastrow, Sethy, Ramabhadran, and Jelinek, 2009b). However, the names in this example “Slobodan Milošević” and “Radovan Karadžić” were not in the system’s vocabulary. These words are un-common, foreign, so were not encountered

[1] We use IBM’s Attila LVCSR system (Soltau, Saon, and Kingsbury, 2010). Section 2.2.1 describes the details of the system.

in the English training text for the system, and thus not included in the vocabulary of the recognizer. When out-of-vocabulary words are spoken at recognition time, LVCSR systems simply predict the most likely *in vocabulary* word sequence for those acoustics. In this case, Slobodan Milošević was transcribed as: “slow but i am a lost of its” and Radovan Karadžić as: “rather than carriages”.

For any reasonably sized domain, it is essentially impossible to predefine a vocabulary that covers all words that will be encountered at recognition. OOVs are always a problem for LVCSR systems, regardless of the size of the vocabulary since topics and words are constantly changing. The OOV rate for a given system depends on two properties: the size of the vocabulary, and the mismatch between training and testing. For very large vocabularies (>100,000), well matched train/test conditions, and static domains, the OOV rate can be as low as a fraction of a percent. However the OOV rate is significantly larger for open domains (Hetherington, 1995). Out-of-Vocabulary words are an important source of errors in speech recognition for a number of reasons:

- OOVs are often information rich nouns, such as named entities, technical terms, and foreign words, and mis-recognizing them can have a disproportionate impact on transcript coherence.

- OOVs also cause errors in their neighboring words, since they affect context, and typically there are two errors per OOV occurrence (Sadaoki Furui and Iwano, 2005).

- Recognition errors caused by OOVs propagate through downstream applications greatly affecting the performance of information extractions tasks, such as: spoken term detection (Can, Cooper, Sethy, White, Ramabhadran, and Saraclar, 2009), spoken document retrieval (Mamou, Ramabhadran, and Siohan, 2007), speech-to-speech translation (Stymne, Holmqvist, and Ahrenberg, 2010), and named-entity recognition (Huang, 2005).

In LVCSR systems the designation *word* denotes *word form* defined by its spelling. For example, two inflections or derivations of the same word (eat vs eats) are considered different words. Since each variation of a word must be included in the dictionary, morphological variations of a word are seen as OOVs. Hence, this problem is more pronounced in morphologically rich languages, such as: Turkish, Finish, Czech, etc.

The OOV problem remains an important one for English, specially in open domains with constantly changing vocabularies, such as the news domain, or applications with constant domain shifts such as using speech to search the web (*VoiceSearch* [2]) or any social-media application (*YouTube* transcription[3]). In this dissertation the experimental evaluations are carried out in English. We focus on two domains: Broadcast News, where OOV words are mostly named-entities, and MIT Lectures with OOVs representing mostly technical terms. However, the methods presented are mostly language independent.

[2] http://www.google.com/mobile/voice-search/

[3] http://googlesystem.blogspot.com/2009/11/youtube-audio-transcription.html

Can we solve this problem by adding more words to the lexicon of the LVCSR system?

A common approach to solve the OOV problem is to increase or adapt the vocabulary of the LVCSR system to the domain of interest. While this is a reasonable solution for static, well-known domains with abundant resources, it is not a practical one for domains with constantly changing topics and vocabulary. Figure 1.2 illustrates the vocabulary growth as a function of corpus size for nine different corpora varying in languages (English, French, and Italian) as well as domain (human-computer interactions, conversational speech, news domain) presented by Hetherington (1995). We can see in this figure that the set with slowest growth rate corresponds to: ATIS, F-ATIS, VOYAGER, and I-VOYAGER, which are corpora with spontaneous utterances from human-computer interactions; and the group with highest rate (Wall Street Journal, New York Times, and BREF) corresponds to orthographic transcriptions of newspaper articles. The higher vocabulary growth rate in news articles is due to constantly evolving topics which introduce new words. More importantly, we can see that the growth rates in the news domain do not plateau, indicating that new words will always appear and increasing the lexicon will not solve the out-of-vocabulary problem.

The new word effect is also an important problem for web-based applications. Figure 1.3 depicts the percent of new n-grams found on crawled websites by the Bing[4] search engine over a period of six months in 2008 and 2009. Each bar shows the ratio of new

[4] www.bing.com

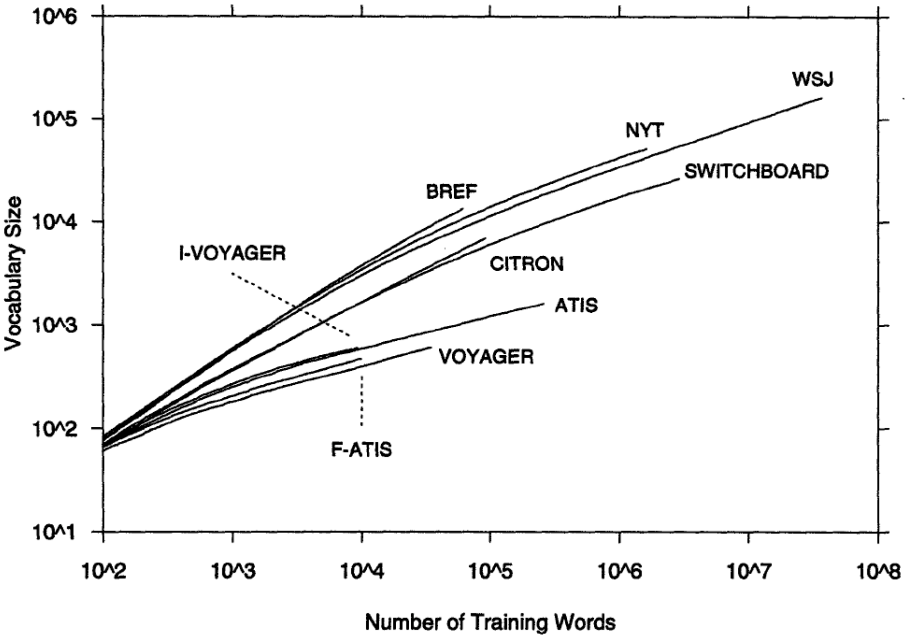


Figure 1.2: Vocabulary growth as a function of training data size for nine corpora spanning English, French, and Italian languages as well as different domains (human-computer interactions, conversational speech, news). (Hetherington, 1995)

.

n-grams with respect to the vocabulary seen up to the previous month[5]. We can see that regardless of efforts to update the vocabulary each month, new N-grams continue to appear at a similar rate. This suggests that solving the OOV problem is critical to the success of web-based speech applications, such as Voice Search or YouTube transcriptions.

Finally, increasing the vocabulary of the LVCSR system has two conflicting effects. On one hand it improves performance by lowering the OOV rate, but at the same time it includes more acoustic confusability in the recognizer (Rosenfeld, 1995). This leads to a tradeoff between recognition accuracy of frequent and rare words. An alternative solution

[5] This figure is courtesy of Kuansan Wang, Principal Researcher at Microsoft Research. Figure from seminar presentation at CLSP, Johns Hopkins University, Feb 02 2011. http://www.clsp.jhu.edu/news-events/abstract.php?sid=20110202 (Wang, 2011)

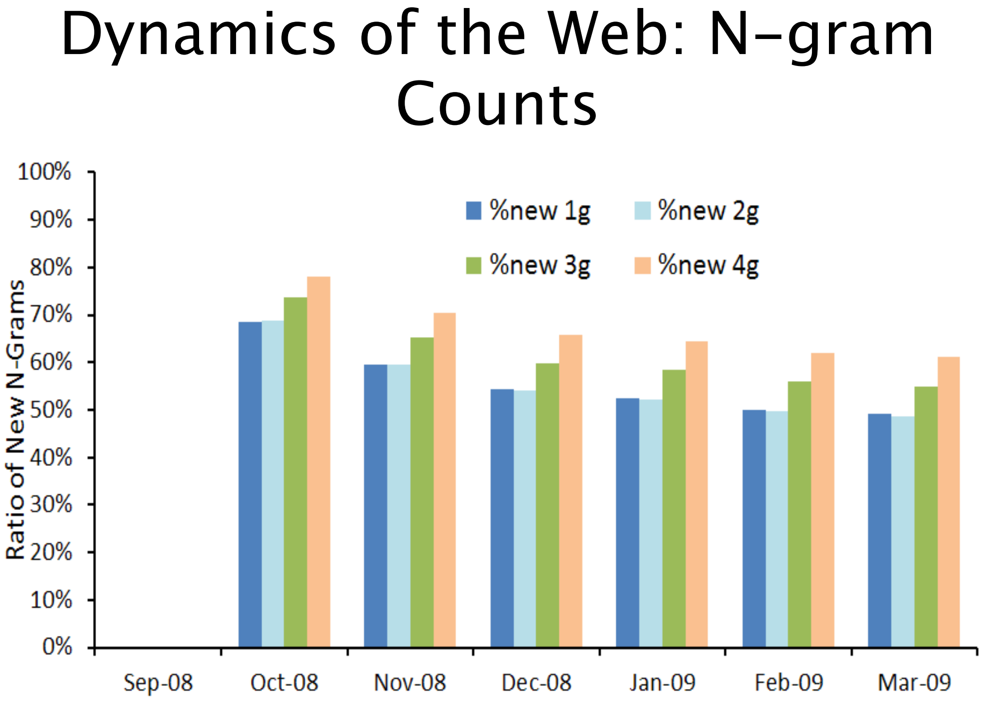


Figure 1.3: Dynamics of the Web: N-gram counts. Each bar shows the ratio of new N-grams with respect to the vocabulary seen up to that month. The list of known words accumulates over time, hence in Nov 08, 60% of words on the web have never been seen in all previous months combined. (Wang, 2011)

is to build a recognizer at the sub-lexical level: phones or syllables are finite and they can be used to represent any word. However, sub-lexical recognizers have been shown to achieve significantly lower performance in number of errors at the phonetic and word level (Bazzi and Glass, 2000). Using a word system reduces ambiguity in the number of possible output transcriptions achieving superior performance to sub-lexical approaches.

Hybrid word/sub-word systems provide a compromise between word-only and sub-word only based approaches. Adding sub-word units to a large vocabulary word lexicon allows the recognizer to output sub-word sequences when OOVs are spoken since sub-words can closely match the pronunciation of new words. Complete words are transcribed for in-vocabulary (IV) terms. Consider the word “Slobodan” from our earlier example (Figure 1.1). While a LVCSR system outputs the closest known words (e.x. “slow but i am”), a hybrid system could output a sequence of multi-phoneme units: `s_l_ow`, `b_ax`, `d_ae_n`. The latter is more useful for automatically recovering the word’s correct spelling, identifying that an OOV was spoken, or improving performance of downstream applications such as: spoken term detection system with OOV queries or information extraction applications on speech.

In fact, hybrid systems have achieved better phone error rates, especially in OOV regions (Rastrow et al., 2009b), obtained state-of-the-art performance for OOV detection (Rastrow, Sethy, and Ramabhadran, 2009a; Wang, 2009; Choueiter, 2009), and achieved improvements in spoken term detection and spoken document retrieval (Akbacak, Vergyri, and Stolcke, 2008; Ng, 1990) in English. Despite these advantages, hybrid systems are not commonly used in state-of-the-art English recognizers. Previous studies typically report results on artificially low vocabulary sizes and it is not clear if these advantages hold for large vocabulary systems[6] . Furthermore, the design of these systems is often decoupled from the final task: reducing word errors when OOVs are spoken for transcription.

[6] A notorious exception is the work by Creutz, Hirsimaki, Kurimo, Puurula, Pylkkonen, Siivola, Varjokallio, Arisoy, Saraclar, and Stolcke (2007); Arisoy, Can, Parlak, Sak, and Saraclar (2009), which presents a hybrid system for morphologically rich languages with very large vocabulary sizes, concluding that hybrid approaches are capable of modeling a larger set of words without, however, compromising the performance of limited vocabulary covered by the word models in a statistical significant way. They also show improved spoken term detection performance in Turkish.

1.2 The Proposed Solution

Towards the goal of open vocabulary recognition, we recover from LVCSR system failures due to out-of-vocabulary words by designing systems that can identify and recover arbitrary previously unseen words in the output of a hybrid recognizer and pass this knowledge to downstream applications.

This dissertation presents a system that identifies when new words are spoken and transcribes them using optimized sub-word units. This results in a hybrid word/sub-word recognizer which predicts complete words for in-vocabulary utterances, and sub-word units for OOVs. The output of this system is used to 1) recover the orthography of new words for transcription, resulting in an open-vocabulary system; and 2) improve performance of downstream information extraction tasks such as: spoken term detection and named-entity recognition in speech.

We introduce a data-driven approach to learn sub-word units. Unlike previous approaches, we learn sub-word units that can be optimized for a given task, and we select the task of open-vocabulary recognition. The goal is to learn sub-word units such that the recognizer output them only for OOV regions and prefers to output complete words for in-vocabulary regions. We also propose a novel method to identify OOV regions in the output of the hybrid system combining the new sub-word units with confidence based methods including contextual information. These two contributions achieve significant improvements in OOV detection compared to state-of-the-art large vocabulary systems.

Furthermore, we show that the learned sub-words and improved detection are critical to recover the spelling of new words for transcription, resulting in an open-vocabulary system; and for improving robustness in downstream applications. While previous work focuses on evaluating new sub-word units for OOV detection, we show significant improvements in detection, recognition accuracy, spoken term detection, and information extraction applications in a Broadcast News and MIT Lectures task.

Figure 1.4 depicts an example utterance processed through the open vocabulary system proposed in this dissertation. In this example, the word “Fujimori" was not in the 83,000 word vocabulary, however the proposed open-vocabulary system is able to: (a) transcribe this word using sub-lexical units[7], (b) identify when this new word was spoken, (c) recover its correct spelling, (d) correctly label it as a PERSON’s name, and (e) find all instances of this word in the audio when searching for “Fujimori".

As can be seen in Figure 1.4, the system consists of several components, each of which includes a novel contribution. We start by proposing a *hybrid* recognizer which transcribes new words using the proposed sub-word units optimized for this task (a). The resulting hybrid output is input to an OOV detector which helps identify when new words are spoken (b). Given the identified OOV regions, we exploit their context and the vast and constantly updated vocabulary of the Web to recover the spelling of new words for transcription (c). We also show that the hybrid output with identified OOV regions can be used to improve

[7] The hybrid output is shown in the form of confusion networks, which are compact representations of the recognizer’s hypotheses. Each set of words between two nodes represent competing hypothesis for a particular time interval.

performance for downstream information extraction tasks, such as: spoken term detection (d) and information extraction systems (e).

These components will be described one by one in the following chapters. In what follows, we present the contributions of this dissertation in the context of relevant previous work.

Learning Sub-Word Units for Open Vocabulary Recognition

Hybrid word/sub-word systems include sub-word units in large vocabulary word based systems. How do we select the set of words and sub-words to be included in such a hybrid system? How relevant is the lexicon selection to performance?

An important shortcoming of hybrid systems is that sub-word units can be predicted when in-vocabulary words are spoken, degrading performance for typically correct words. Ideally these units would be produced only when OOVs are spoken and the system should prefer words otherwise. Previous work heuristically created the sub-word lexicon from phonetic representations of text using simple statistics to select common phone sequences (Bazzi and Glass, 2001; Bisani and Ney, 2005; Rastrow et al., 2009a). However, it isn't clear why these units would produce the best hybrid output.

Our Contribution:

We propose a probabilistic model to *learn* the sub-word lexicon optimized for a given task. We consider the task of open vocabulary recognition encouraging sub-words to be predicted if, and only if, new words are spoken. Our approach employs a log-linear model with overlapping features to learn variable-length multi-phone units obtained by segmenting the phonetic representation of a corpus. This probabilistic model is learned un-supervised, requiring as input only text, a dictionary, and a letter-to-sound model. The learned sub-words are combined with a word lexicon to generate a hybrid system. The proposed system improves out-of-vocabulary detection and achieves lower phone error rates with respect to state-of-the-art approaches.

Out-of-Vocabulary Detection

Identifying when new words are spoken and mis-recognized is a key step to avoid error propagation to downstream applications, as well as to recover the orthography of new words for transcription. This task is known as Out-of-Vocabulary detection.

OOV detection is typically addressed as a binary classification task, where each region is independently classified using local information (Rastrow et al., 2009a; Hazen and Bazzi, 2001; Lin, Bilmes, Vergyri, and Kirchhoff, 2007). However, OOVs tend to be recognized as multiple in-vocabulary words and have specific distributional similarities and syntactic roles. In this dissertation we investigate whether contextual information can improve OOV detection.

Our Contribution:

We show that jointly predicting OOV regions, and including contextual information from each region, leads to substantial improvement in OOV detection. Specifically, we treat OOV detection as a sequence labeling problem and use a conditional random field

(CRF) model to label all regions in an utterance as OOV or IV (in-vocabulary). We also exploit lexical information from the context, such as word N-grams (i.e. what are the previous two words for this region?), and language model scores as features in our CRF model. Compared to state-of-the-art results, these methods reduce the missed OOV rate significantly, the detector is used to recover the orthography of new words, and improve robustness of downstream applications.

Recovering Out-of-Vocabulary words

For dictation applications we need to transcribe the correct spelling for any novel word. We refer to this task as *OOV Recovery*. Previous work acquires Web data to increase the amount of language model training text (Ng, Ostendorf, Hwang, Siu, Bulyko, and Lei, 2005; Creutz, Virpioja, and Kovaleva, 2009; Oger, Popescu, and Linares, 2009). The main drawback of these methods is that they require re-decoding the test audio after updating the system. In some applications it is impractical to re-decode, e.g. in a media monitoring/surveillance/browsing system.

Our Contribution:

This dissertation presents a novel approach to *recover* the spelling of OOV terms using the Web as a corpus. Similar to previous work, we query the web for words relevant to our test utterance. To correct the errors we explore two approaches: use a spoken term detection (STD) framework to correct each error region on demand (without the need to re-decode) or by re-decoding with an augmented lexicon.

Downstream applications affected by OOVs

OOVs pose an important problem for information extraction and search applications since queries typically relate to information rich nouns, such as named-entities and foreign words, which have poor coverage in the vocabulary. In this dissertation, we demonstrate the usefulness of the proposed hybrid recognizer and OOV detector and introduce further algorithmic improvements to enhance robustness to OOVs in two information extraction applications: spoken term detection (STD), and named-entity recognition (NER) in speech.

Spoken Term Detection

The goal in spoken term detection (STD) is to do an open-vocabulary search over a large speech database. This task is typically addressed by first processing the audio through a LVCSR system, and subsequently building an index from the output lattices or confusion networks from the recognizer (see J. Mamou and Hoory (2006) and references therein). Critically, the search of queries containing OOV terms in the LVCSR processed output will not return any results.[8] A typical solution to OOV queries, is to build a phonetic index. We can then search for the phonetic representation of the OOV term (i.e. its pronunciation) in this phonetic index (Mamou et al., 2007; Can et al., 2009). However, this approach yields low recall and high false alarm rates for OOV queries.

[8] This is due to the fact that LVCSR systems only transcribe words from a closed word lexicon. By definition, OOVs are NOT in the lexicon.

Our Contribution:

We demonstrate the benefits of the proposed hybrid and OOV detection systems to improve performance for out-of-vocabulary queries in a spoken term detection system. We also explore incorporating phonetic confusability and additional features that boost the probability of a hit in accordance with the number of neighboring hits for the same query and query-length normalization to improve the overall performance of the spoken-term detection system.

Named-Entity Recognition

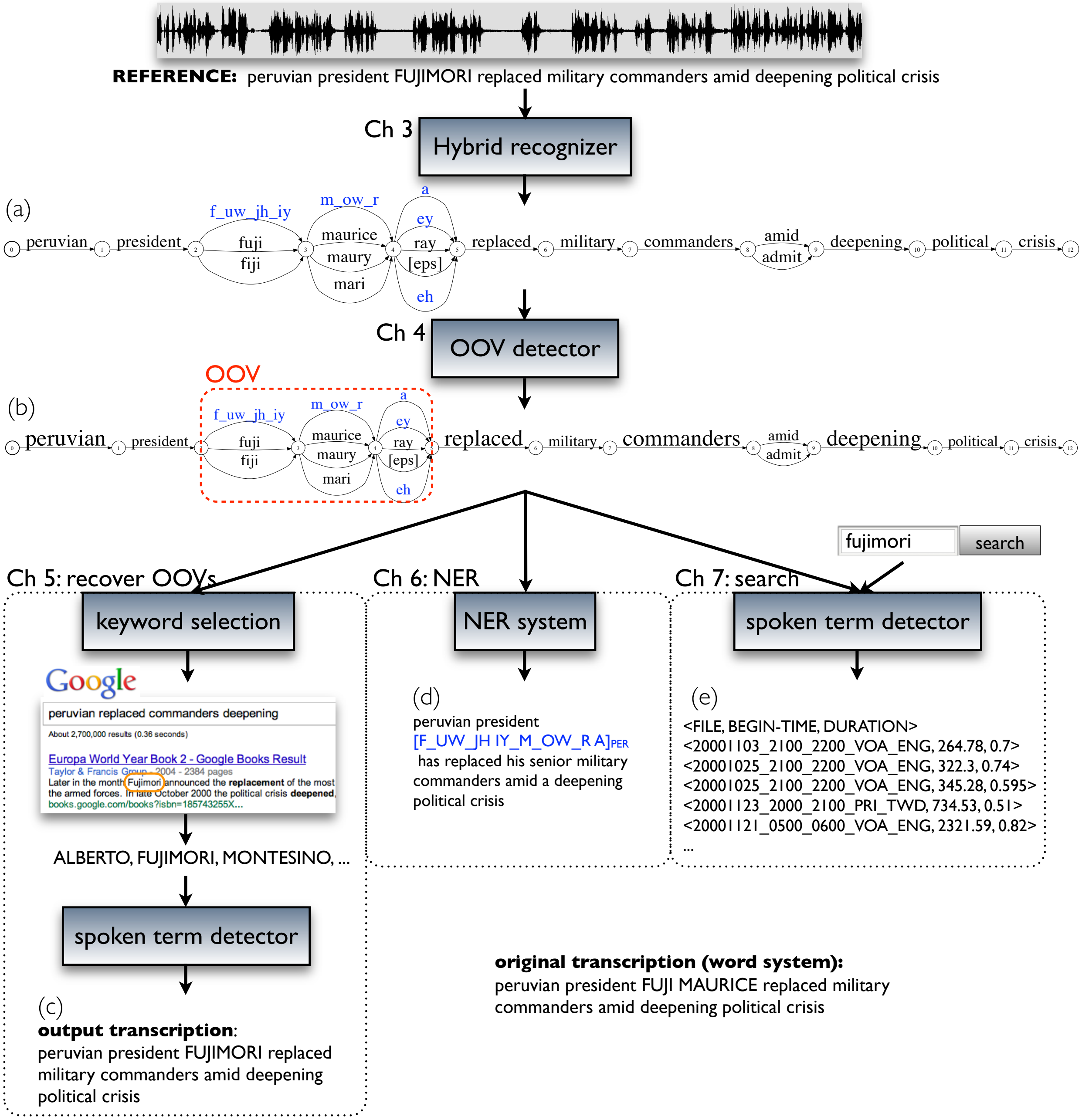
Named Entity Recognition (NER), an information extraction task, is typically applied to spoken documents by cascading an automatic speech recognizer (ASR) and a named entity tagger. However, ASR errors are especially problematic for NER, since many OOVs are proper names (66% of the OOVs in our corpus are named entities). However, the problem with named-entities containing out-of-vocabulary terms has been largely un-addressed.

Our Contribution:

We improve speech NER by including features indicative of OOVs obtained from the proposed OOV detection system. This allows us to identify regions of speech containing named entities, even if they are incorrectly transcribed.

1.3 Roadmap

This dissertation is organized as follows. In Chapter 2 we present the relevant background material, describing in detail the speech recognition pipeline used, as well as previous approaches to OOV modeling. Each of the following five chapters present the contributions of this dissertation and are based on previous publications by the author (Parada, Dredze, Sethy, and Rastrow, 2011; Parada, Dredze, Filimonov, and Jelinek, 2010a; Parada, Sethy, and Ramabhadran, 2010c; Parada, Sethy, Dredze, and Jelinek, 2010b), respectively. These publications have been extended here by adding more expositions and connections to the bigger picture and by including more experiments and error analysis. Figure 1.4 illustrates how these contributions are integrated to build an open-vocabulary system. Chapter 3 describes our approach to learn optimal sub-word units for open-vocabulary speech recognition. This hybrid system is assumed as input to all remaining modules of the system. Chapter 4 describes the Out-of-Vocabulary detector, which is a key component to identify new words and transcribe them (Chapter 5), and to improved performance of downstream applications, such as: Spoken Term Detection (Chapter 6) and Named-Entity Recognition (Chapter 7). Finally we summarize and conclude in Chapter 8.



REFERENCE: peruvian president FUJIMORI replaced military commanders amid deepening political crisis

peruvian replaced commanders deepening

About 2,700,000 results (0.36 seconds)

Europa World Year Book 2 - Google Books Result
Taylor & Francis Group - 2004 - 2384 pages
Later in the month Fujimori announced the **replacement** of the most
the armed forces. In late October 2000 the political crisis **deepened**,
books.google.com/books?isbn=185743255X...

ALBERTO, FUJIMORI, MONTESINO, ...

output transcription:
peruvian president FUJIMORI replaced military commanders amid deepening political crisis

peruvian president [F_UW_JH IY_M_OW_R A]PER has replaced his senior military commanders amid a deepening political crisis

<FILE, BEGIN-TIME, DURATION>
<20001103_2100_2200_VOA_ENG, 264.78, 0.7>
<20001025_2100_2200_VOA_ENG, 322.3, 0.74>
<20001025_2100_2200_VOA_ENG, 345.28, 0.595>
<20001123_2000_2100_PRI_TWD, 734.53, 0.51>
<20001121_0500_0600_VOA_ENG, 2321.59, 0.82>
...

original transcription (word system):
peruvian president FUJI MAURICE replaced military commanders amid deepening political crisis

Figure 1.4: An example utterance processed by the open vocabulary system. In this example, the word “Fujimori" was not in the 83,000 word vocabulary, however the proposed open-vocabulary system is able to: (a) transcribe this word using sub-lexical units, (b) identify when this new word was spoken, (c) recover its correct spelling, (d) correctly label it as a PERSON’s name (d), and (e) find all instances of this word in the audio when searching for “Fujimori".

Chapter 2

Background

This chapter provides the general background relevant to this dissertation. It is divided into three parts. In the first part we describe the main components of a Large Vocabulary Continuous Speech Recognition (LVCSR) system. The second part explains the Attila speech recognition system developed by IBM (Soltau et al., 2010), and the two corpora we used in our experiments: Broadcast News and MIT Lectures. Finally, we present previous work on modeling of Out-of-Vocabulary words and their effect on downstream speech applications.

2.1 LVCSR Pipeline

The goal of automatic speech recognition (ASR) is to transcribe a spoken utterance into the corresponding string of words. This task is typically evaluated on a number of speech

```
REF:  yugoslav leader ** **** SLOBODAN MILOSEVIC WHO  IS    *** also A serb
HYP:  yugoslav leader TO WHAT I        AM        MOST IMAGE WAS also * serb
EVAL:                 I  I    S        S         S    S     I        D
```

Figure 2.1: Reference-Hypothesis alignment. In this example there are 3 insertions (I), 4 substitutions (S), and 1 deletion (D), resulting in WER $= 100\frac{3+4+1}{9} = 89\%$.

utterances and their corresponding transcriptions (test-set), by computing the **Error Rate** between the automatic transcript (hypothesis) and the human *reference* transcription. The best system is the one that achieves the lowest Error Rate given by Equation 2.1 below:

$$\text{Error-rate} = 100\frac{\#\text{ insertions} + \#\text{ deletions} + \#\text{ substitutions}}{\#\text{units in reference}} \tag{2.1}$$

where the number of insertions, deletions and substitution is with respect to the best possible alignment between the hypothesis and the human transcription. This alignment is the one yielding the smallest number of errors. An example alignment is shown in Figure 2.1. Typically word error rate (WER), and/or phone error rate (PER) are reported.

State-of-the-art systems take a statistical approach to speech recognition (F. Jelinek and Mercer, 1975). The goal is to find the most-likely word string $\hat{\mathbf{W}} = \hat{w}_1, \hat{w}_2, \ldots, \hat{w}_n$ spoken, given the observed acoustic evidence $\mathbf{A}$. This can be written formally as follows:

$$\hat{\mathbf{W}} = \underset{\mathbf{W}\in\mathcal{L}}{\operatorname{argmax}}\, P(\mathbf{W}|\mathbf{A}) \tag{2.2}$$

where $P(\mathbf{W}|\mathbf{A})$ denotes the probability that the words $\mathbf{W}$ were spoken given that the evidence $\mathbf{A}$ was observed. Using Bayes' rule, we can rewrite Equation 2.2 as follows [1]:

$$\hat{\mathbf{W}} = \underbrace{\operatorname*{argmax}_{\mathbf{W} \in \mathcal{L}}}_{\text{search}} \underbrace{P(\mathbf{A}|\mathbf{W})}_{\text{acoustic model}} \times \underbrace{P(\mathbf{W})}_{\text{language model}} \tag{2.3}$$

where we can see the different components of a speech recognizer:

- The **acoustic model** determines the value of $P(\mathbf{A}|\mathbf{W})$, which represents the probability that the acoustic evidence $\mathbf{A}$ will be observed if the word string $\mathbf{W}$ was spoken. This is a statistical model of context-dependent or context-independent phones, and it is typically trained on large amounts of audio with corresponding phonetic transcriptions.

- The **language model** estimates the value of $P(\mathbf{W})$, which represents the prior probability that speaker wishes to utter the words $\mathbf{W}$. It is typically trained on large amounts of text.

- The **search** evaluates the set of all possible words strings in the lexicon $\mathcal{L}$, and returns the one with the highest probability according to the acoustic and language model.

Another important component is the **acoustic processor** (front end), which transforms the pressure waveform into the acoustic data $\mathbf{A}$ with which the recognizer will deal. Fig-

[1] Bayes' rule states that $P(A|B) = \frac{P(B|A)P(A)}{P(B)}$, where $P(B|A)$ is the probability of event B given that is known that event A is known to have happened; and $P(A)$ is the prior probability of event A if nothing else is known. Equation 2.2 is re-written as: $P(W|A) = \frac{P(A|W)P(W)}{P(A)}$. Since the denominator is constant for all hypothesis W, it is ignored in the maximization and we obtain Equation 2.3.

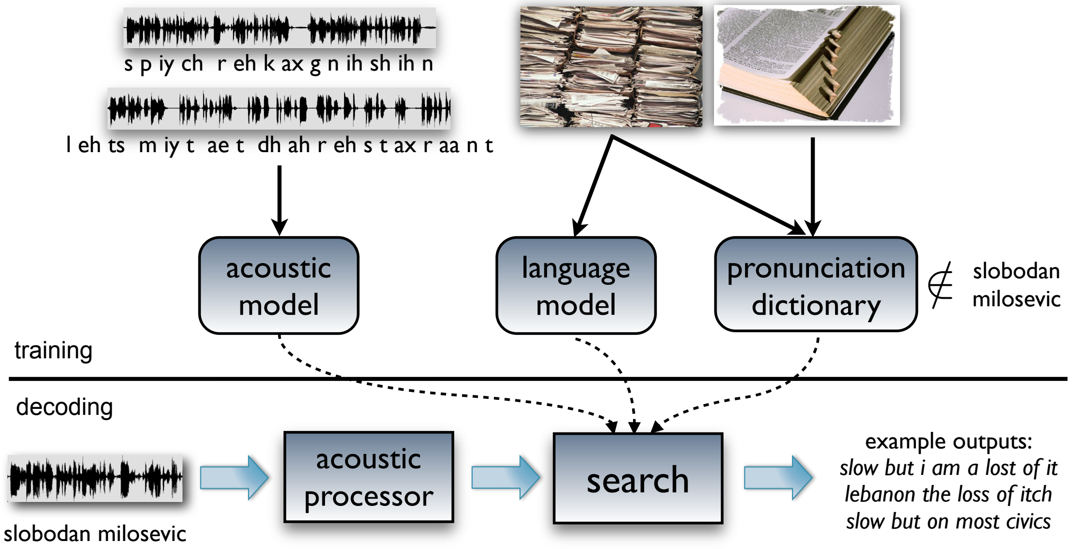

s p iy ch r eh k ax g n ih sh ih n
l eh ts m iy t ae t dh ah r eh s t ax r aa n t
acoustic model
language model
pronunciation dictionary
∉ slobodan milosevic
training
decoding
slobodan milosevic
acoustic processor
search
example outputs:
slow but i am a lost of it
lebanon the loss of itch
slow but on most civics

Figure 2.2: Pipeline for a large vocabulary continuous speech recognition (LVCSR) system. The top and bottom sections correspond to training and decoding, respectively. In this example, both words in the test utterance “slobodan milosevic" were not in the dictionary for the system, and hence they were mis-recognized.

ure 2.2 illustrates the speech recognition pipeline, where the upper and lower parts correspond to training and decoding respectively. Due to the enormity of the search space (i.e. all possible utterances in a language), large vocabulary continuous speech recognition (LVCSR) systems restrict the search to a large but finite word set: **the lexicon** $\mathcal{L}$ (also known as the pronunciation dictionary). This lexicon typically includes all words in the language model training text, or even the set of most frequent words in the training text. In the following sections we explain in detail each of the components of the LVCSR pipeline.

2.1.1 The Acoustic Processor

The speech recognition formulation in Equation 2.3 starts from the acoustic evidence **A**. This symbol represents a sequence of vector observations $\mathbf{A} = a_1, a_2, \ldots, a_T$ for a given input utterance. The goal of the acoustic processor is to transform the raw input

Speech Pre-Processing

The acoustic processor transforms the raw acoustic wave to a sequence of feature vectors. A feature vector is obtained every 10 ms. The most commonly used techniques to obtain these feature vectors are:

- **Mel Frequency Cepstral Coefficients (MFCC)** (Davis and Mermelstein, 1980): these features are derived by first applying the Short Time Fourier Transform (STFT) on speech (or acoustic wave) with an analysis window of length 25 ms and a frame shift of 10 ms. The magnitude square values of the STFT output are warped onto the mel frequency scale and then compressed using a logarithm. Discrete cosine transform (DCT) is applied on the resultant compressed energies to obtain mel cepstrum. MFCC feature vectors are obtained by stacking first 13 mel cepstral coefficients along with the corresponding delta and delta-delta features.

- **Perceptual Linear Prediction (PLP) Cepstral Coefficients** (Hermansky, 1989): these include 39 coefficients derived from the PLP cepstrum. In this case, the magnitude square values of the STFT output are warped onto the Bark frequency scale. Then a sequence of nonlinear compressions are applied (equal loudness, cubic root) to reduce the dynamic range, and the spectral envelope is smoothed by the twelfth order linear prediction analysis. PLP cepstrum is obtained by applying the DCT on smoothed spectral envelope. These transformations are inspired by human speech perception.

Other techniques to supplement the initial preprocessing include:

- **Linear Discriminant Analysis (LDA)** (Haeb-Umbach and Ney, 1992; Saon, Padmanabhan, Gopinath, and Chen, 2000): a matrix transformation of the feature vector to maximize separation between different phone classes.

- **Mean and Variance Normalization**: normalizing the data in a given feature dimension to increase robustness in the features to speaker and channel variation.

- **Vocal Tract Length Normalization (VTLN)** (Cohen, Kamm, and Andreou, 1995; Eide and Gish, 1996): a technique to warp frequencies to cancel speaker variations with respect to the vocal tract length. It reduces gender variation.

- **FMLLR** (Y. Li and Marcheret, 2002): feature space maximum likelihood linear regression, where the feature vectors are linearly transformed to perform feature-based speaker adaptation.

Figure 2.3: Common techniques used in the Acoustic Processor

(i.e. the acoustic wave generated by the speech) into the acoustic evidence $\mathbf{A}$ with which the recognizer will deal. The acoustic wave is typically sampled at 8,000 Hz or 16,000 Hz and quantized. A set of transforms is applied to the signal with an analysis window of 25 ms and a frame shift of 10 ms, resulting in the sequence of feature vectors $\mathbf{A} = a_1 \dots a_T$. Figure 2.1.1 describes the most commonly used features along with techniques to supplement the pre-processing of the speech signal.

2.1.2 Acoustic Modeling

The goal of the acoustic model is to estimate the probability $P(\mathbf{A}|\mathbf{W})$ for any acoustic evidence $\mathbf{A} = a_1, a_2, \dots, a_T$ and hypothesized word string $\mathbf{W} = w_1, w_2, \dots, w_n$, where $\mathbf{A}$ and $\mathbf{W}$ are variable length sequences. Typically w_i is restricted to a finite lexicon $\mathcal{L}$, and $a_i \in \Re^K$ is a feature vector obtained from the acoustic processor. State-of-the-art acoustic models are based on the Hidden Markov Model (HMM). An overview of Hidden Markov Models is given in Jelinek (1997).

The acoustic model consists of different layers of representations as depicted in Figure 2.4. The final model is computed from the hierarchical composition of HMMs in each layer. It works as follows: (Bahl and Jelinek, 1975; Baker, 1975a,b)

- The word string $\mathbf{W}$ is represented by the concatenation of the individual HMM models corresponding to the words w_i, $i \in 1, 2, \dots, n$.

- Each word is in turn represented by the concatenation of HMM models for the sub-word units that compose them. These sub-word units are typically context independent or context dependent phones corresponding to the phonetic representation of a word, also called the baseform $\mathbf{B}$. For example, the word “EITHER" has the baseforms: `AY DH ER` and `IY DH ER` where the pronunciation is obtained from the standard PRONLEX dictionary [2]. Assuming the acoustics $\mathbf{A}$ are independent of the word sequence $\mathbf{W}$ given the phoneme sequence $\mathbf{B}$, the acoustic model can be written as:

$$P(\mathbf{A}|\mathbf{W}) = \sum_{\mathbf{B}} P(\mathbf{A}|\mathbf{B}) P_{\mathcal{L}}(\mathbf{B}|\mathbf{W}) \qquad (2.4)$$

  where L is the pronunciation dictionary containing the most common pronunciations for each word.

- The phoneme sequence is typically converted to context dependent phonemes, such as triphones, and each triphone is represented by a three state left-to-right HMM. Let $q(\mathbf{B})$ represent the HMM state sequence for the triphone sequence in the baseform $\mathbf{B}$. The acoustic model becomes:

$$P(\mathbf{A}|\mathbf{W}) = \sum_{\mathbf{B}} P(\mathbf{A}|q(\mathbf{B})) P_L(\mathbf{B}|\mathbf{W}) \qquad (2.5)$$

  Since all these probabilities are estimated from data, it is often the case that it is not possible to accurately estimate $P(\mathbf{A}|q(\mathbf{B}))$ for all possible triphone sequences.

[2] English pronunciation dictionary produced by LDC, 1995:
http://www.cs.cmu.edu/afs/cs/user/ahlen/www/pronlex.html

Hence, in practice HMM states are tied across (or share among) different sets of triphones which are obtained by clustering their context.

- Finally, within the HMM framework each acoustic vector a_t, $t \in 1, 2, \ldots, T$ is generated by a state in the HMM. To account for variability in the length of the acoustic signal corresponding to a phoneme, the HMM contains a self loop allowing for variable number of acoustic vectors produced by each state. Let $\mathbf{U} = u_1, u_2, \ldots, u_T$ denote the state sequence which generated the output sequence $\mathbf{A}$. Then we write:

$$\begin{aligned} P(\mathbf{A}|q(\mathbf{B})) &= \sum_{\mathbf{U}} P(\mathbf{A}, \mathbf{U}|q(\mathbf{B})) \\ P(\mathbf{A}, \mathbf{U}|q(\mathbf{B})) &= \prod_{t=1}^{T} P_{out}(a_t|u_t) P(u_t|u_{t-1}, q(\mathbf{B})) \end{aligned}$$

  Typically $P_{out}(a_t|u_t)$ is assumed to be described by a mixture of Gaussian densities which provides flexibility for fitting the data:

$$P_{out}(a_t|u_t) = \sum_{j=1}^{N} w_j^{u_t} \mathcal{N}(a_t; \mu_j^{u_t}, \Sigma_j^{u_t}) \tag{2.6}$$

  where $\mathcal{N}(\cdot; \mu_j^{u_t}, \Sigma_j^{u_t})$ represents the Gaussian distribution with mean $\mu_j^{u_t}$ and covariance $\Sigma_j^{u_t}$, which are state and mixture-component dependent.

To summarize, as shown in Figure 2.4, the word sequence $\mathbf{W}$ is represented by the concatenation of the word HMM models. Each word HMM consists of the sequence of phoneme HMM models $\mathbf{B}$, where the phoneme sequence is obtained using the pronunciation dictionary. Finally, each context-dependent phoneme in $\mathbf{B}$ is modeled using a 3-state HMM model $\mathbf{q}$. Over time, the states in this HMM can represent a variable length sequence of acoustic vectors $\mathbf{A}$, which correspond to the feature vectors extracted from the acoustic signal.

For each transition t in a specific context-dependent HMM model q, the parameter estimation task is concerned with the values $\theta = \{P(u_t|u_{t-1}, q(\mathbf{B}), \mu_j^{u_t}, \Sigma_j^{u_t}, w_j^{u_t}\}$. These parameters are learned from large amounts of manually transcribed audio. Given a transcribed utterance, we can build the composite HMM containing all the words in the utterance, where each word is represented by the HMM sequence corresponding to the context-dependent phonemes in its baseform. These parameters are estimated efficiently using the Forward-Backward algorithm (Baum and Petrie, 1966), which is a special case of the Expectation-Maximization algorithm (Dempster, Laird, and Rubin, 1977).

2.1.3 Language Modeling

The language model estimates the prior probability that the speaker wishes to utter the word string $\mathbf{W} = w_1, w_2, \ldots, w_n$. This corresponds to $P(\mathbf{W})$ in Equation 2.3. Using the definition of conditional probability, the probability of a word sequence can be written as follows:

$$P(\mathbf{W}) = P(w_1, w_2, \ldots, w_n) = \prod_{i=1}^{n} P(w_i|w_1, \ldots, w_{i-1}) \tag{2.7}$$

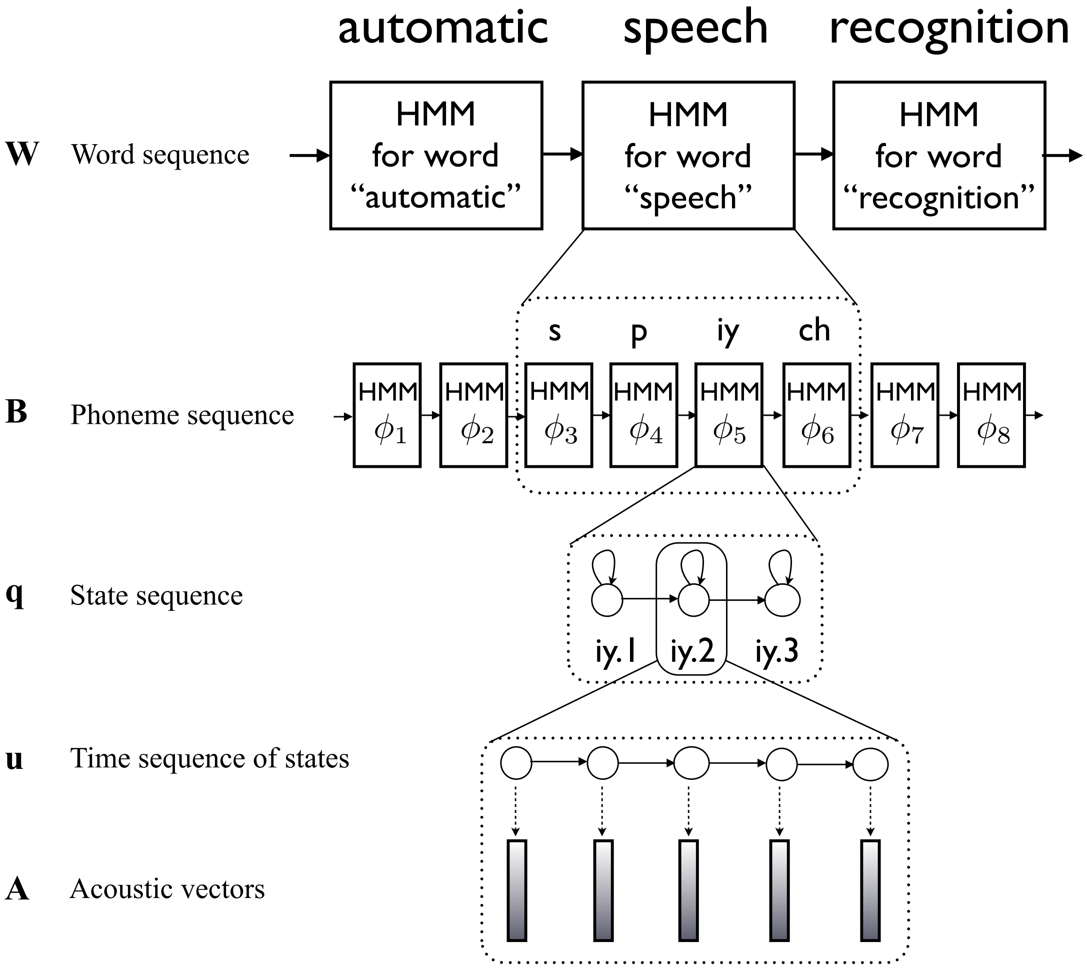


Figure 2.4: The HMM hierarchy of representations in the acoustic model.

Hence the goal of the language model is to estimate the probabilities $P(w_i|w_1, \ldots, w_{i-1})$, for all possible word sequences $w_1, w_2, \ldots, w_i$, where $w_i \in \mathcal{L}$. Even for moderate vocabulary sizes and small values of i, it is not possible accurately estimate this probabilities since many of the word sequences in the history $w_1, w_2, \ldots, w_{i-1}$ will never be observed or will be observed very few times. For example, for $|\mathcal{L}| = 100,000$ and a history size of $i = 3$, the total possible set of word sequences is $100,000^3 = 1x10^{15}$ (one quadrillion), which will need to be accurately estimated and stored. In practice, the vocabulary is typically limited to the most frequent set of words in the training text (so that we can accurately estimate their occurrence), and the history is limited to a finite set M of *equivalent* histories Φ which we can handle.

$$P(\mathbf{W}) \approx \prod_{i=1}^{n} P(w_i|\Phi(w_1, \ldots, w_{i-1})) \tag{2.8}$$

Virtually all state-of-the-art LVCSR systems use a very simple but effective equivalence classification for the history: the N-gram language model, where N is typically 3 (trigram language model). According to this model, histories are equivalent if they end in the same 2 words. Then Eq. (2.8) becomes:

$$\begin{aligned} P(\mathbf{W}) &\approx \prod_{i=1}^{n} P(w_i|w_{N-1}, \ldots, w_{i-1}) && (2.9) \\ &\approx \prod_{i=1}^{n} P(w_i|w_{i-2}, w_{i-1}) && (2.10) \end{aligned}$$

This probability can be easily obtained by counting of all trigrams $c(w_{i-2}, w_{i-1}, w_i)$ and bigrams $c(w_{i-1}, w_i)$ in the LM training text: $P(w_i|w_{i-2}, w_{i-1}) = c(w_{i-2}, w_{i-1}, w_i)/c(w_{i-2}, w_{i-1})$. To avoid assigning zero probabilities for unseen trigrams, it is necessary to *smooth* the trigram frequencies. Many approaches have been proposed: additive smoothing (Lidstone, 1920), linear interpolation (Jelinek and Mercer, 1980), Katz Smoothing (Katz, 1987), Witten-Bell Smoothing (Witten and Bell, 1991), Absolute Discounting (Ney, Essen, and Kneser, 1994), and Kneser-Ney smoothing (Kneser and Ney, 1995), modified Kneser-Ney smoothing (Chen and Goodman, 1998), among others. In our experiments we use modified Kneser-Ney smoothing since it has been shown to outperform all other approaches, and is the most commonly used technique. For more detailed overview of these techniques please refer to Chen and Goodman (1998).

Smoothing alleviates the data sparsity problem for unseen N-gram histories. The probability of predicting a novel word (OOV) is obtained by assigning all rare words in the training text to a special <UNK> symbol which is included in the vocabulary. Other approaches include explicitly modeling the OOVs as we will discuss in Section 2.3.

2.1.4 The Lexicon

The lexicon or pronunciation dictionary contains a list of words with associated pronunciation(s). Although most words have a single pronunciation, multiple pronunciations are allowed to account for pronunciation variability (Hazen, Hetherington, Shu, and Livescu, 2005). Many state-of-the-art LVCSR systems use as a dictionary the PRONLEX dictionary

(LDC 1995) designed for speech recognition. It contains pronunciations for 90,694 wordforms, covering all words over many years of *Wall Street Journal* and the *Switchboard Corpus* (Godfrey, Holliman, and McDaniel, 1992), and uses a 39 ARPABET-derived[3] phone set.

A table look-up is used to find the pronunciation associated with each word. Note that novel words (Out-of-Vocabulary words) have no way of being transcribed since their pronunciation cannot be mapped to an orthographic representation in this approach.

2.1.5 Search (decoding)

The IBM speech recognition toolkit Attilla uses pre-compiled static decoding networks that allow extremely fast recognition. These networks integrate the acoustic information (provided by the HMM), the lexical information (provided by the pronunciation dictionary), and the language model information (provided by the statistical N-gram language models). Decoding is performed by searching the most likely word string on this network using the Viterbi Algorithm (Viterbi, 1967).

[3] The ARPABET is a selection of symbols used within the Advanced Research Projects Agency, Speech Understanding Research (ARPA SUR) (Shoup, 1980)

2.2 Corpora, Task, and System Definition

2.2.1 Attila Speech Recognition System

For LVCSR, we used the IBM Speech Recognition Toolkit Attila (Soltau et al., 2010). As in most LVCSR systems, it operates in a series of steps described below:

1. Front-End preprocessing: the speech utterance is chunked into 20 ms frames with a frame-shift of 10 ms. Each frame is represented by 19-dimensional PLP features, and their mean and variance is normalized on a per-utterance basis. At each frame, a LDA transformed is applied (considering 4 frames of context on each side) to project the vector down to 40 dimensions. Because the context of frames is used in this transform, delta and delta-delta features are not included.

2. Speaker Independent (SI) acoustic modeling: in this step, each sub-word unit (a phoneme) is represented by a 3-state left-to-right HMM with no skip states. First, maximum likelihood estimation is used to learn the parameters of the HMM from transcribed audio. These context-independent (CI) models produce a set of state-level alignments of the speech against the corresponding audio, which are then used to bootstrap triphone context-dependent (CD) models. To avoid data sparseness problems, the CD triphones are clustered using a top-down decision tree, and data is shared among triphones in the same class. After clustering a set of GMMs is trained for each state.

3. Speaker Dependent (SA) acoustic modeling: the SI models are used to bootstrap training for the SA models. In SA modeling VTLN and feature/model space adaptation is applied. Specifically, after VTLN the warped features are adapted to each speaker using FMMLR. Next the CD models are adapted to each speaker using Maximum Likelihood Linear Regression (MLLR) (Gales, 1998).

4. Although discriminative training based on Boosted Maximum Mutual Information (BMMI) criterion (Povey, Kanevsky, Kingsbury, Ramabhadran, Saon, and Visweswariah, 2008) was also available in this system, it was not used in our experiments.

2.2.2 Experimental Setup and Corpora

For our experiments, the acoustic models use speaker adaptive training and are based on maximum likelihood as described in steps 1-3 in Section 2.2.1. They were trained on 300 hours of Hub4 data (Fiscus, Garofolo, Przybocki, Fisher, and Pallett, 1998). The language model was trained on 400M words from various Broadcast News data sources including (Chen, Kingsbury, Mangu, Povey, Saon, Soltau, and Zweig, 2006): 1996 CSR Hub4 Language Model data, EARS BN03 closed captions, GALE Phase 2 Distillation GNG Evaluation Supplemental Multilingual data, Hub4 acoustic model training scripts (corresponding to the 300 Hrs), TDT4 closed captions, TDT4 newswire, GALE Broadcast Conversations, and GALE Broadcast News.

In order to study the OOV problem, we restrict the dictionary to contain 83,000 words from the PRONLEX dictionary and an average of 1.08 pronunciation variants per word. All LMs used are 4-gram LMs with interpolated modified Kneser-Ney smoothing. The LVCSR system's WER on the standard RT04 BN test set was 19.4%. Note that the vocabulary used is close to most modern LVCSR system vocabularies for English Broadcast News; the resulting OOVs are more challenging but more realistic (i.e. mostly named entities and technical terms).

Finally, we also build several hybrid LVCSR systems, combining word and sub-word units obtained from either the proposed approach in Chapter 3 or a state-of-the-art baseline approach (Rastrow et al., 2009a). Our hybrid system's lexicon has 83K words and 5K or 10K sub-words. The evaluation is conducted on two test-sets from different domains: OOVCORP and MIT Lectures, described below.

OOVCORP

Our primary evaluation set is the data-set constructed by Can et al. (2009) for the evaluation of Spoken Term Detection of OOVs since it focuses on the OOV problem. The corpus contains 100 hours of transcribed Broadcast News English speech from the Hub4 corpus. There are 1290 unique OOVs in the corpus, which were selected with a minimum of 5 acoustic instances per word and short OOVs inappropriate for STD (less than 4 phones) were explicitly excluded. Examples of OOVs include: NATALIE, PUTIN, QAEDA, HOLLOWAY, COROLLARIES, HYPERLINKED. This resulted in roughly 24K (2%)

OOV tokens.

MIT Lectures

In addition we report OOV detection results on a MIT lectures data set (Glass, Hazen, Hetherington, and Wang, 2010) consisting of 3 hours from two speakers with a 1.5% OOV rate. Note that the LVCSR system is trained on Broadcast News data. This out-of-domain test-set help us evaluate the cross-domain performance of the proposed and baseline hybrid systems. OOVs in this data set correspond mainly to technical terms in computer science and math. e.g. ALGORITHM, DEBUG, COMPILER, LISP.

2.3 The OOV Problem: Previous Work

As described in Section 2.1, LVCSR systems fail to recognize any out-of-vocabulary (OOV) word. Instead, new words are transcribed as a sequence of in-vocabulary terms which closely resemble their acoustics. Since OOVs are inevitable, many approaches have been proposed to cope with them in order to build practical applications. Hetherington (1995) first characterized the OOV problem in detail, demonstrating its magnitude across several languages and domains and describing the problems caused by OOV words across a range of speech recognition/understanding tasks. Below are some of the conclusions drawn from nine different corpora varying in languages (English, French, and Italian) as well as domain (human-computer interactions, conversational speech, news domain):

- The new word rate is higher for open ended task domains such as News, and is more relevant in applications intended for a human audience (as supposed to machine dialog systems).

- Although the new-word rate drops with increasing training set and vocabulary size, it does not reach zero. Furthermore, even when the OOV rate can be reduced to below 1%, these words affect nearly 17% of utterances, which is detrimental for any dialog or transcription application.

- Typically OOVs cause 1.5 to 2 word errors since words adjacent are also misrecognized because of the presence of the OOV.

- New words are largely nouns, proper nouns, adjectives, and verbs. Their length in number of syllables or phonemes per word is only slightly longer than in-vocabulary words.

Recent work also demonstrates the necessity of handling new words in morphologically-rich languages such as: Turkish, Finish, Estonian, and Arabic (Creutz et al., 2007). In these languages the "brute-force" approach of growing the word-vocabulary is infeasible even for static domains. In Finish, for example, a training text of 150 million words contains more than 4 million unique word forms; and even including all these words in the lexicon the OOV rate of the test-set in these experiments was about 1.5%. They conclude that sub-words (in their case *morphs*) are capable of modeling a much larger set of words without compromising the performance on the limited vocabulary covered by word models.

In this dissertation, we study the new word problem in four areas: 1) modeling out-of-vocabulary words using sub-lexical units, 2) detecting and locating when an utterance contains an out-of-vocabulary word, 3) learning these new words for transcription, and 4) improving robustness to new words in downstream applications. We now describe previous work relevant to each of these areas.

2.3.1 Modeling OOVs using sub-lexical units

Previous work models Out-of-Vocabulary words by including in the LVCSR system some form of filler or sub-lexical representation. There is significant work in this area, varying in terms of the approach used to integrate the sub-words in the LVCSR system, and the nature of the sub-lexical units derived.

Flat versus Hierarchical OOV Models

In **hierarchical OOV models**, the LVCSR lexicon is expanded to include one or multiple generic word models W_{OOV}, which allow for arbitrary phone sequences during recognition. As depicted in Figure 2.5a, the generic OOV word model (W_{OOV}) is considered in parallel with all other words during recognition, and it is essentially a phonetic recognizer which can transcribe any new word.

These systems are called hierarchical because they have a sub-word language model embedded within a word-based language model. The former can be trained on phonetic transcriptions of text or a dictionary, and the units used can also be more complex than just phones (Bazzi, 2002). To avoid large number of false alarms (sub-words predicted when in-vocabulary words are spoken), these systems typically include a penalty/cost to enter the W_{OOV} model which is tuned on a development set. Examples of hierarchical hybrid systems include Asadi, Schwartz, and Makhoul (1989); Suhm, Woszczyna, and Waibel (1993); Scharenborg and Seneff (2005); Bazzi and Glass (2001); Bazzi (2002).

Flat hybrid models directly include both words and sub-word units in the LVCSR's lexicon. Contrary to hierarchical systems, there is a single language model which is able to predict both words and sub-word sequences as shown in Figure 2.5b. To learn this language model the training text is preprocessed to replace all OOVs (words outside a fixed word lexicon) as sub-words creating a word/sub-word training text. Then standard language modeling techniques are used on the hybrid text. Examples of flat hybrid models include Klakow, Rose, and Aubert (1999); Galescu (2003); Yazgan and Saraclar (2004); Bisani and Ney (2005); Rastrow et al. (2009a); Choueiter (2009).

The main advantages of flat hybrid systems with respect to hierarchical approaches are:

- The sub-word portion of the language model is trained on the least frequent words.

- Dependencies between words and sub-words are better captured (no forced back-off to phone LM within OOV word).

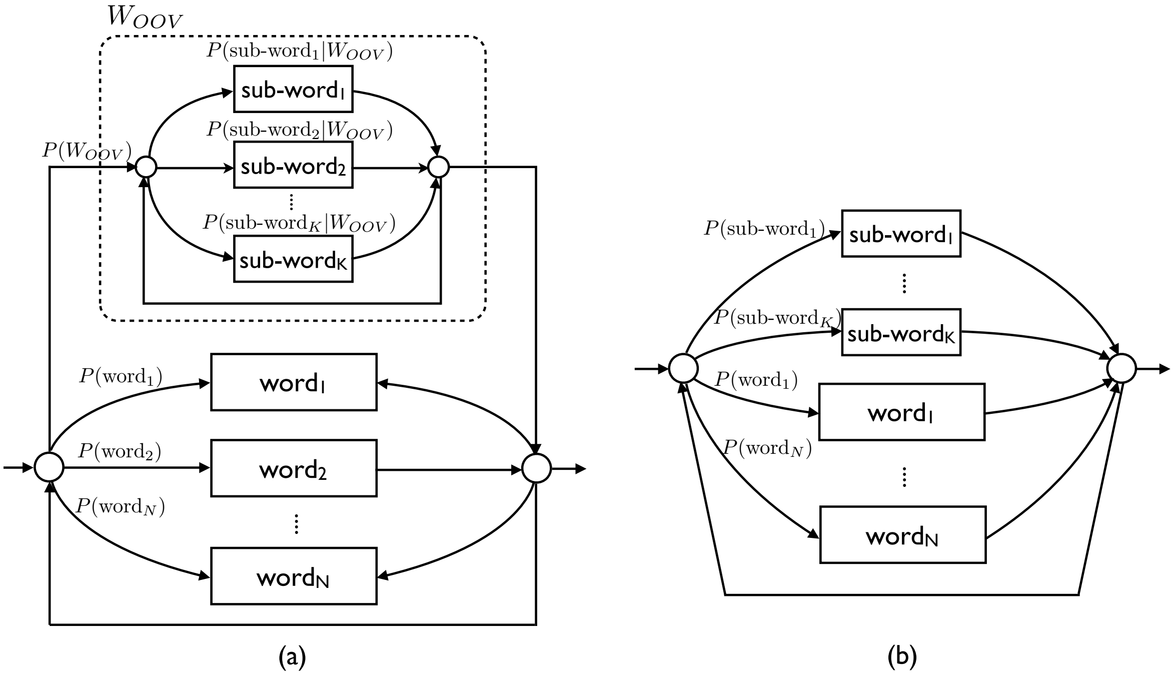


Figure 2.5: The hybrid search network. (a) Hierarchical hybrid: OOV model is explored in parallel to all other words. (b) Flat Hybrid: all words and sub-words are explored in parallel.

- There is no need to adjust a cost/penalty to enter/leave the W_{OOV} network in order to avoid false alarms.

A shortcoming of flat models is that it leaves undetermined the location of word-boundaries if multiple OOVs are spoken in sequence. However, this can be solved by having distinct word-beginning sub-words, and it is not a problem if the main task is OOV detection or if a two-pass system is used to recover the spelling of new words. In this work, we use a flat hybrid approach to integrate our proposed sub-words in the LVCSR system.

Sub-lexical units

Previous work on sub-lexical representations for modeling OOVs can be clustered into two groups: knowledge-driven vs data-driven approaches.

Knowledge driven sub-word units include phones (Asadi et al., 1989; Suhm et al., 1993), syllables (Yazgan and Saraclar, 2004), morphological units[4] (Arisoy et al., 2009), and sub-syllabic units (Choueiter, 2009). Yazgan and Saraclar (2004) found that using word-phone hybrid models yield better detection of OOVs than word-syllable models in an English Switchboard task. Since syllables were considered too long to represent new words Choueiter (2009) proposed sub-syllabic units designed using a context-free grammar, however the resulting system was not compared with other hybrid models. Finally, Arisoy et al. (2009) explore using morph-based units: stem+ending sub-words, concluding however that statistical morph-like units learned in a data-driven approach using *Morfessor* (Creutz and Lagus, 2005) are more effective for Turkish ASR performance, than using a morphological parser to obtain these units.

Data driven sub-word units are typically preferred to single phones, morphemes, or syllables since they 1) achieve better performance (Bazzi, 2002; Arisoy et al., 2009), 2) do not require language-specific knowledge, 3) provide more control over the size of the sub-word lexicon.

The most common type of data-driven sub-word unit are variable-length phone sequences, for example the word "dictionary" can be written as: `d_ih_kd sh_ax_n`

[4] Example morphological units include: stem, morphemes associated with tense, aspect, modality, etc.

eh_r_iy. Approaches to generate these units include: an iterative approach to merge units with Maximum Mutual Information (MMI) (Bazzi, 2002; Klakow et al., 1999) or selecting the most frequent phone N-grams as sub-words (Rastrow et al., 2009a). Other data-driven approaches for unsupervised morphological segmentation (Deligne, Yvon, and Bimbot, 1995; Creutz and Lagus, 2005; Poon, Cherry, and Toutanova, 2009) also obtain sub-word units which can be included directly in a hybrid recognizer (Arisoy et al., 2009).

Other types of data-driven units include variable length letter/phone sequence pairs, for example the word "dictionary" can be written as: dic/d_ih_kd tion/sh_ax_n ary/eh_r_iy. These units have the advantage that they include the orthographic representation of the sub-word, which can be useful to recover the orthography of the OOV word in non-phonetic languages such as English. These sub-words can be learned from data: using grapheme-to-phoneme conversion approaches based on the EM algorithm as suggested by Galescu (2003); Bisani and Ney (2005); Wang (2009) (called it *graphones*), or learned using context-free rules that encode sub-syllabic linguistic knowledge, such as positional, phonological, and stress information as proposed by Choueiter (2009) (called it *spellnemes*).

In this dissertation we consider how to optimally create sub-word units to include in a hybrid system for open-vocabulary recognition. We use variable-length phone sequences as units, although in principle our work can be used with other unit types. We introduced a probabilistic model to *learn* sub-word units optimized for open-vocabulary recognition.

2.3.2 Detection of Out-of-Vocabulary words

OOV detection aims to identify regions in the LVCSR output where OOVs were uttered. Previous work on OOV detection can be categorized into two broad groups: 1) *hybrid (filler) models*, which explicitly model the OOVs as described in Section 2.3.1; and 2) *confidence-based approaches*: which label unreliable regions as OOVs based on different confidence scores, language models, and lattice scores.

Confidence based approaches combine multiple features from the recognizer output lattices to classify each region as in-vocabulary (IV) or OOV. Several confidence metrics have been proposed. In Sun, Zhang, f. Zheng, and Xu (2001) word-level normalized log-likelihood scores, number of active paths in the search space, number of similar paths, number of fillers in the sentence, and similar information from the previous and next word are combined using Fisher Linear Discriminative Analysis to detect OOVs. Lin et al. (2007) performs a joint alignment between independently generated word and phone lattices. Using a similarity measure between phones, they can locate highly misaligned regions and labeled them as candidate OOVs. Burget, Schwarz, Matejka, Hannemann, Rastrow, White, Khudanpur, Hermansky, and Cernocky (2008) compare phone posteriors from two systems: weakly-constrained (phonetic-based) and strongly-constrained (word-based) recognizers using a neural-network (NN). They achieve substantial improvements over standard confidence estimators, such as C_{max}[5]. White, Zweig, Burget, Schwarz, and Hermansky (2008) extends this work to consider string-based comparisons between weak and strong

[5] $C_{max} = \max_{t \in (t_s, t_e)} p(w_i|t)$ (Wessel, Schluter, Macherey, and Ney, 2001) is the confidence of hypothesized word w_i spanning time (t_s, t_e)

recognizers at the phonetic and word level rather than at the frame-level using a maximum entropy model achieving further improvements.

As we discussed in Section 2.3.1, hybrid recognizers provide a straight-forward way to detect new words, namely the presence of sub-word units in the recognition output indicates an OOV was uttered. However, sub-word units matching the acoustics of in-vocabulary words can also be predicted outside OOV regions generating false-alarms. Combining hybrid models with confidence-based methods reduces false alarms and achieves improved performance over each of these methods alone. Some systems combinining hybrid and confidence models include: Bazzi (2002); Rastrow et al. (2009a); Sun et al. (2001).

In this dissertation we combine confidence-based methods with a hybrid system to achieve improved OOV detection performance. As described here, previous work addresses the OOV detection problem as a binary classification task, where each region is independently classified using local information. We show that jointly predicting OOV regions, and including contextual information from each region leads to substantial improvements in OOV detection.

2.3.3 Recovering the spelling of OOV words

Assuming we have identified that a new word was spoken, we need to learn its spelling for transcription. If the sub-word unit encodes the phonetic and graphemic information as in *graphones* and *spellnemes* (Galescu, 2003; Bisani and Ney, 2005; Choueiter, 2009), the orthography can be obtained by direct concatenation of the graphemes. Alternatively, we can obtain word spellings from the phonetic units using:

- A phoneme to grapheme converter (P2G), also known as a sound-to-letter model (S2L) which outputs the spelling of a word given its pronunciation. (Meng, Seneff, and Zue, 1994; Deligne et al., 1995; Decadt, Duchateau, Daelemans, and Wambacq, 2002; Bisani and Ney, 2008)
- A very large pronunciation dictionary to select words with closest pronunciation to the hypothesized phonetic sequence. This large dictionary can be either a fallback lexicon (Scharenborg and Seneff, 2005; Rastrow et al., 2009b), or a set of words retrieved from an alternative source of knowledge (large amounts of text, the web, etc) (Huang, 2005; Oger et al., 2009).

Using graphones/spellnemes or a letter-to-sound model to obtain the spelling assumes the decoded sequence is free of recognition errors, while using an external source of knowledge (a word list) can match the phonetic sequences in the lattice to the closest word in the list. Also, it ensures the recovered words are legitimate words in the language (as long as the list of words is reliable). Since a larger fallback lexicon is not always available, one can use large amounts of text or the web to find relevant words.

Oger et al. (2009) proposed an approach to retrieve words relevant to the utterance's topic from the web. The retrieved words were included in a locally augmented lexicon and in the language model, and the utterances were re-decoded. Different approaches can be used to incorporate new words in the language model, including using the part-of-speech (POS) tag of the word, using a default OOV token score (Oger et al., 2009), or estimating the novel-word probabilities by their similarity to a *synonym* word class (Jelinek, Mercer, and Roukous, 1990).

In this dissertation we propose an approach to recover the correct orthography of the OOV word from the Web; and incorporate the OOV terms in LVCSR output by re-decoding with an augmented lexicon, or using a spoken term detection framework which does not require modifying the LVCSR system.

2.3.4 Effect of OOVs in downstream applications

OOVs represent an important source of error in LVCSR systems. These words cause recognition failures which propagate through pipeline systems impacting the performance of downstream applications, such as: spoken document retrieval, spoken term detection, information extraction, and speech translation.

Spoken document retrieval and spoken term detection are key technologies that allow for search for documents or specific word locations within large collections of audio. The most common approach for these technologies is to use a LVCSR system to obtain word lattices and then apply text information retrieval techniques to retrieve relevant documents. OOVs are an important problem in these applications because: (Woodland, Johnson, Jourlin, and Jones, 2000)

- OOV words will not be encountered in the LVCSR transcripts producing misses when searching for OOV queries.
- They cause false alarms due to spurious in-vocabulary terms caused by the recognizer incorrectly substituting for OOV words.
- They can cause missing term relations, which affect expansion sets.

Furthermore, OOVs tend to be information rich nouns like named-entities or technical terms which often correspond to query terms. This has been empirically observed in live audio indexing systems for the web, such as Logan, Moreno, Thong, and Wittaker (1996), where OOVs were found to represent about 15% of search queries. A common approach to combat OOV queries is to search for their pronunciation in a phonetic index. However, this approach typically yields low recall and high false alarm rates for OOV queries compared to in-vocabulary terms (Can et al., 2009; Mamou et al., 2007; Arisoy et al., 2009).

Since OOVs typically correspond to information rich nouns, including named entities, they affect the performance of information extraction tasks on speech such as Named-Entity Recognition (Huang, 2005). However this problem is largely un-addressed.

Chapter 3

Learning Sub-Word Units for Open Vocabulary Speech Recognition

3.1 Introduction

Hybrid recognizers include both words and sub-words in the lexicon. This allows them to predict a sequence of *sub-word units* in place of OOV words. A few questions come to mind when designing such systems: how should we select the set of words and sub-words to be included in the hybrid lexicon? How relevant is the lexicon selection to performance?

In this chapter we consider how to optimally create sub-word units for a hybrid system. These units are variable-length phoneme sequences, although in principle our approach can be used for other unit types. Previous methods for creating the sub-word lexicon have relied on simple statistics computed from the phonetic representation of text (Klakow et al., 1999; Bazzi, 2002; Rastrow et al., 2009a). These units typically represent the most frequent phoneme sequences, or the phoneme pairs with highest mutual information[1] in a given corpus. However, it isn't clear why these units produce the best hybrid output.

Instead, we introduce a probabilistic model for *learning* the optimal units for a given task. Our model learns a segmentation of a text corpus given some side information: a mapping between the vocabulary and a label set; learned units are predictive of class labels. This model can be used for any un-supervised segmentation task, where the segmentation depends on the label assigned to a word (e.g. OOV/IV, POS tag, topic, etc.). We only require that the labels assigned to each word in the text belong to a finite set $y \in \{0, 1, \ldots, c\}$. Moreover, we assume that there exists an underlying segmentation of the text that can help predict these labels. Our approach is unsupervised: it does not require a segmented training corpus for learning.

In this dissertation, we apply our model to learn the optimal labels for open-vocabulary recognition. The goal is to learn sub-word units such that the recognizer outputs them only for OOV regions while preferring to output a complete word for in-vocabulary regions. Our model can be applied to this task by using a dictionary $\mathcal{L}$ to label words as IV ($y_i = 0$ if $w_i \in \mathcal{L}$) and OOV ($y_i = 1$ if $w_i \notin \mathcal{L}$). This results in a labeled corpus where the sequence Y indicates the presence of out-of-vocabulary words.

[1] For two units x_1 and x_2, the mutual information is defined as: $MI(x_1, x_2) = P(x_1, x_2) \log \frac{P(x_1,x_2)}{P(x_1)P(x_2)}$, where $P(x_i)$ is the marginal probability of x_i, $i \in 1, 2$, and $P(x_1, x_2)$ denotes the probability that x_2 follows x_1 in the training text. Mutual Information is a metric of the amount of uncertainty that is reduced in random variable x_1 when knowing x_2. Hence $MI(x_1, x_2) = 0$ indicates that x_2 never follows x_1 in the text.

3.2 Learning Sub-Word Units

Given raw text, our objective is to produce a lexicon of sub-word units that can be used by a hybrid system for open vocabulary speech recognition. Rather than relying on the text alone, we also utilize side information: a mapping of words to classes so we can optimize learning for a specific task.

The provided mapping assigns labels Y to the corpus. We maximize the probability of the observed labeling sequence Y given the text W: $P(Y|W)$. We assume there is a latent segmentation S of this corpus which impacts Y. The complete data likelihood during training becomes:

$$P(Y|W) = \sum_{S} P(Y, S|W) \tag{3.1}$$

Since we are maximizing the observed Y, segmentation S must discriminate between different possible labels.

We learn variable-length multi-phone units by segmenting the phonetic representation of each word in the corpus. Resulting segments form the sub-word lexicon.[2] Learning input includes a list of words to segment taken from raw text, a mapping between words and classes (side information indicating whether token is IV or OOV), a pronunciation dictionary $\mathcal{L}$, and a letter-to-sound (L2S) converter. A L2S model converts a word from its spelling (e.g. `recognition`) to its pronunciation (e.g.

[2] Since sub-word units can expand full-words, we refer to both words and sub-words simply as units.

`r,eh,k,ax,g,n,ih,sh,ih,n`). In our experiments we used the L2S model presented by Stanley F. Chen (2003).

The corpus W is the list of types (unique words) in the raw text input. This forces each word to have a unique segmentation, shared by all common tokens. Words are converted into phonetic representations according to their most likely dictionary pronunciation; non-dictionary words use the L2S model. This model can also be naturally extended to account for multiple pronunciations as we describe in Section 3.3.1.

3.2.1 Model

Inspired by the morphological segmentation model of Poon et al. (2009), we assume $P(Y, S|W)$ is a log-linear model parameterized by Λ:

$$P_\Lambda(Y, S|W) = \frac{1}{Z(W)} u_\Lambda(Y, S, W) \tag{3.2}$$

where $u_\Lambda(Y, S, W)$ defines the score of the proposed segmentation S for words W and labels Y according to model parameters Λ. Sub-word units σ compose S, where each σ is a phone sequence, including the full pronunciation for vocabulary words; the collection of σs form the lexicon. Each unit σ is present in a segmentation with some context $c = (\phi_l, \phi_r)$ of the form $\phi_l \sigma \phi_r$. Features based on the context and the unit itself parameterize u_Λ.

In addition to scoring a segmentation based on features, we include two priors suggested by Poon et al. (2009). The **lexicon prior** favors smaller lexicons by placing an exponential prior with negative weight on the length of the lexicon $\sum_\sigma |\sigma|$, where $|\sigma|$ is the length of the unit σ in number of phones. Minimizing the lexicon prior favors a trivial lexicon of only the phones. The **corpus prior** counters this effect, an exponential prior with negative weight on the number of units in each word's segmentation, where $|s_i|$ is the segmentation length and $|w_i|$ is the length of the word in phones.

These priors are inspired by the Minimum Description Length (MDL) (Rissanen, 1989) principle. This principle states that the best way to capture regular features is to construct a model which allows for the shortest description of the data. In our model, the lexicon prior favors the shortest description of the lexicon, while the corpus prior favors the shortest description of the corpus. Learning strikes a balance between the two priors.

Using these definitions, the segmentation score $u_\Lambda(Y, S, W)$ is given as:

$$\begin{aligned} u_\Lambda(Y,S,W) = \exp\Bigg(&\sum_{\sigma,y} \lambda_{\sigma,y} f_{\sigma,y}(S,Y) + \sum_{c,y} \lambda_{c,y} f_{c,y}(S,Y) \\ &+ \alpha \cdot \sum_{\sigma \in S} |\sigma| \\ &+ \beta \cdot \sum_{i \in W} |s_i|/|w_i|\Bigg) \end{aligned} \tag{3.3}$$

where $f_{\sigma,y}(S, Y)$ are the co-occurrence counts of the pair (σ, y) where σ is a unit under segmentation S and y is the label. $f_{c,y}(S, Y)$ are the co-occurrence counts for the context c and label y under S. The model parameters are $\Lambda = \{\lambda_{\sigma,y}, \lambda_{c,y} : \forall \sigma, c, y\}$. The negative weights for the lexicon (α) and corpus priors (β) are tuned on development data. The

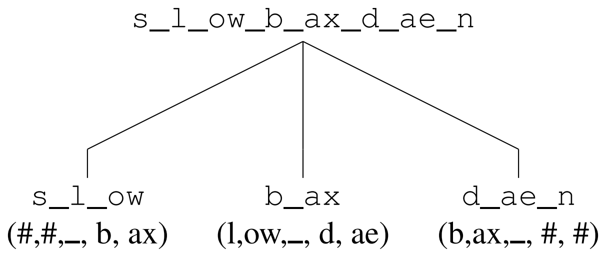


Figure 3.1: Units and bigram phone context (in parenthesis) for an example segmentation of the word “slobodan".

normalizer Z sums over all possible segmentations and labels:

$$Z(W) = \sum_{S'} \sum_{Y'} u_\Lambda(Y', S', W) \tag{3.4}$$

Consider the example segmentation for the word “slobodan" with pronunciation s,l,ow,b,ax,d,ae,n (Figure 3.1). The bigram phone context as a four-tuple appears below each unit; the first two entries correspond to the left context, and last two the right context. The example corpus (Table 3.1) demonstrates how unit features $f_{\sigma,y}$ and context features $f_{c,y}$ are computed.

3.3 Model Training

Learning maximizes the log likelihood of the observed labels Y^* given the words W:

$$\ell(Y^*|W) = \log \sum_{S} \frac{1}{Z(W)} u_\Lambda(Y^*, S, W) \tag{3.5}$$

Labeled corpus:	**Segmented corpus**:
president/$y = 0$ milosevic/$y = 1$	p_r_eh_z_ih_d_ih_n_t/0 m_ih/1 l_aa/1 s_ax/1 v_ih_ch/1

Unit-feature:Value	**Context-feature:Value**
p_r_eh_z_ih_d_ih_n_t/0:1	(#/0,#/0,_,#/0,#/0):1
m_ih/1:1	(#/0,#/0,_,l/1,aa/1):1
l_aa/1:1	(m/1,ih/1,_,s/1,ax/1):1
s_ax/1:1	(l/1,aa/1,_,v/1,ih/1):1
v_ih_ch/1:1	(s/1,ax/1,_,#/0,#/0):1

Table 3.1: A small example corpus with segmentations and corresponding features. The notation m_ih/1:1 represents unit/label:feature-value.

We use the Expectation-Maximization algorithm (Dempster et al., 1977), where the *expectation step* predicts segmentations S given the model's current parameters Λ (Section 3.3.1), and the *maximization step* updates these parameters using gradient ascent. The partial derivatives of the objective Eq. (3.5) with respect to each parameter λ_i are:

$$\frac{\partial \ell(Y^*|W)}{\partial \lambda_i} = E_{S|Y^*,W}[f_i] - E_{S,Y|W}[f_i] \tag{3.6}$$

The gradient takes the usual form, where we encourage the expected segmentation from the current model given the correct labels to equal the expected segmentation and expected labels. The next section discusses computing these expectations.

3.3.1 Inference

Inference is challenging since the lexicon prior renders all word segmentations interdependent. Consider a simple two word corpus: cesar (s,iy,z,er), and cesium (s,iy,z,iy,ax,m). Numerous segmentations are possible; each word has 2^{N-1} possible segmentations, where N is the number of phones in its pronunciation (i.e., $2^3 \times 2^5 = 256$). However, if we decide to segment the first word as: {s_iy, z_er}, then the segmentation for "cesium":{s_iy, z_iy_ax_m} will incur a lexicon prior penalty for including the new segment z_iy_ax_m. If instead we segment "cesar" as {s_iy_z, er}, the segmentation {s_iy, z_iy_ax_m} incurs double penalty for the lexicon prior (since we are including two new units in the lexicon: s_iy and z_iy_ax_m). This dependency requires joint segmentation of the entire corpus, which is intractable. Hence, we resort to approximations of the expectations in Eq. (3.6).

One approach is to use Gibbs Sampling (Koller and Friedman, 2009): iterating through each word, sampling a new segmentation conditioned on the segmentation of all other words. The sampling distribution requires enumerating all possible segmentations for each word (2^{N-1}) and computing the conditional probabilities for each segmentation: $P(S|Y^*, W) = P(Y^*, S|W)/P(Y^*|W)$ (the features are extracted from the remaining words in the corpus). Using M sampled segmentations $S_1, S_2, \dots S_m$ we approximate $E_{S|Y^*,W}[f_i]$ as follows:[3]

$$E_{S|Y^*,W}[f_i] \approx \frac{1}{M} \sum_j f_i[(S, Y^*)_j] \tag{3.7}$$

[3] This approximation follows from the Weak Law of Large Numbers where the empirical mean approximates the true mean of a random variable as the number of samples goes to infinity.

Similarly, we take M joint samples of segmentation and label for each word (S, Y) according to the joint probability of $P(Y, S|W)$ for each segmentation-label pair using Eq. (3.2). We can approximate $E_{S,Y|W}$ as follows:

$$E_{S,Y|W}[f_i] \approx \frac{1}{M} \sum_j f_i[(S, Y)_j] \tag{3.8}$$

A sampled segmentation can introduce new units, which may have higher probability than existing ones.

Using the approximations Eq. (3.7) and Eq. (3.8) in Eq. (3.6), we update the parameters using gradient ascent:

$$\begin{aligned} \bar{\lambda}_{new} &= \bar{\lambda}_{old} + \gamma \nabla \ell_{\bar{\lambda}}(Y^*|W) \\ &= \bar{\lambda}_{old} + \gamma \left[E_{S|Y^*,W}[f_i] - E_{S,Y|W}[f_i] \right] \end{aligned}$$

where $\gamma > 0$ is the learning rate.

To obtain the best segmentation, we use deterministic annealing. In deterministic annealing, sampling operates as usual except that the parameters are divided by a value, which starts large and gradually drops to zero. To make burn in faster for sampling, the sampler is initialized with the most likely segmentation from the previous iteration. To initialize the sampler the first time, we set all the parameters to zero (only the priors have non-zero values) and run deterministic annealing to obtain the first segmentation of the corpus.

3.3.2 Efficient Sampling

Sampling a segmentation for the corpus requires computing the normalization constant (Eq. (3.4)), which contains a summation over all possible corpus segmentations. Instead, we approximate this constant by sampling words independently, keeping fixed all other segmentations. Still, even sampling a single word's segmentation requires enumerating probabilities for all possible segmentations.

We sample a segmentation efficiently using dynamic programming. We can represent all possible segmentations for a word as a weighted finite state machine (WFSM). Figure 3.2 illustrates a WFSM representing all segmentations for the word ANJANI, where the bold path corresponds to the segmentation: AA_N, JH_AA, N_IY. Similarly other paths correspond to other segmentations. In practice each label encodes a sub-word and its context (e.g. "JH, [AA,N, AA,N]") which are the two features required by the model. The arcs weights arise from scoring the segmentation's features. This weight is the negative log probability of the resulting model after adding the corresponding features and priors. A brief overview on finite state automata, including definitions, notations, and common operations, is given in Appendix A.

However, the lexicon prior poses a problem for this construction since the penalty incurred by a new unit in the segmentation depends on whether that unit is present elsewhere in that segmentation. For example, consider the segmentation for the word ANJANI: AA_N, JH, AA_N, IY. If none of these units are in the lexicon, this segmentation yields the lowest

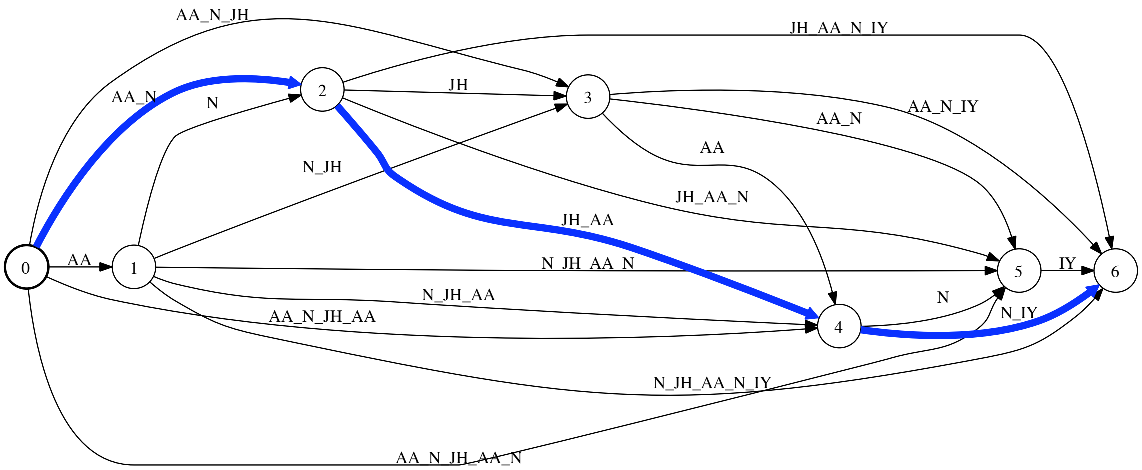


Figure 3.2: FSM representing all segmentations for the word ANJANI with pronunciation: AA,N,JH,AA,N,IY. The bold path corresponds to the segmentation: AA_N, JH_AA, N_IY. Similarly other paths correspond to other segmentations.

prior penalty since it repeats the unit AA_N. [4] This global dependency means paths must encode the full unit history, making computing forward-backward probabilities inefficient.

Our solution is to use the *Metropolis-Hastings* algorithm[5], which samples from the true distribution $P(Y, S|W)$ by first sampling a new label and segmentation (y', s') from a simpler proposal distribution $Q(Y, S|W)$. The new assignment (y', s') is accepted with probability:

$$\alpha(Y', S'|Y, S, W) = \min\left(1, \frac{P(Y', S'|W)Q(Y, S|Y', S', W)}{P(Y, S|W)Q(Y', S'|Y, S, W)}\right) \tag{3.9}$$

We choose the proposal distribution $Q(Y, S|W)$ similar to $P(Y, S|W)$ (Eq. (3.2)) but omitting the lexicon prior, as shown below. We also repeat here the original distribution

[4] Splitting at phone boundaries yields the same lexicon prior but a higher corpus prior.

[5] A *Markov Chain Monte Carlo* (MCMC) method. This framework provides a general approach to generate samples from a posterior distribution in the case where we cannot efficiently sample from the posterior directly (in this case $P(Y, S|W)$. In the Metropolis-Hastings MCMC methods, we sample instead from a *proposal* distribution $Q(Y, S|W)$ and correct for the resulting error. (Koller and Friedman, 2009)

$P(Y,S|W)$ for comparison. Removing the lexicon prior eliminates the challenge for efficient computation, since this is the only term capturing dependencies between repeated sub-words in a segmentation.

$$P_\Lambda(Y,S|W) = \frac{1}{Z(W)} \exp\left(\sum_{\sigma,y} \lambda_{\sigma,y} f_{\sigma,y}(S,Y) + \sum_{c,y} \lambda_{c,y} f_{c,y}(S,Y) + \underbrace{\alpha \cdot \sum_{\sigma\in S} |\sigma|}_{\text{lexicon prior}} + \underbrace{\beta \cdot \sum_{i\in W} |s_i|/|w_i|}_{\text{corpus prior}}\right)$$

$$Q_\Lambda(Y,S|W) = \frac{1}{Z(W)} \exp\left(\sum_{\sigma,y} \lambda_{\sigma,y} f_{\sigma,y}(S,Y) + \sum_{c,y} \lambda_{c,y} f_{c,y}(S,Y) + \beta \cdot \sum_{i\in W} |s_i|/|w_i|\right)$$

Replacing $P(Y,S|W)$ and $Q(Y,S|W)$ in Eq. (3.9), the probability of accepting a sample becomes:

$$\alpha(Y',S'|Y,S,W) = \min\left(1, \frac{\sum_{\sigma\in S'} |\sigma|}{\sum_{\sigma\in S} |\sigma|}\right) \tag{3.10}$$

We sample a path from the FSM by running the forward-backward algorithm, where the backward computations are carried out explicitly, and the forward pass is done through sampling, i.e. we traverse the machine only computing forward probabilities for arcs leaving the sampled state.[6] Once we sample a segmentation (and label) we accept it according to Eq. (3.10) or keep the previous segmentation if rejected.

[6] We use OpenFst's RandGen operation (http://www.openfst.org/twiki/bin/view/FST/RandGenDoc). This operation relies on an *ArcSelector* object for randomly selecting an outgoing transition at a given state in the input FST. In our case we create the *BetaArcSelector* which selects an outgoing transition according to the probability of the outgoing arc times the total probability of leaving the destination state for that arc (typically known as β in the forward backward algorithm.)

Alg. 1 shows our full sub-word learning procedure, where `sampleSL` (Alg. 2) samples a segmentation and label sequence for the entire corpus from $P(Y, S|W)$, and `sampleS` samples a segmentation from $P(S|Y^*, W)$.

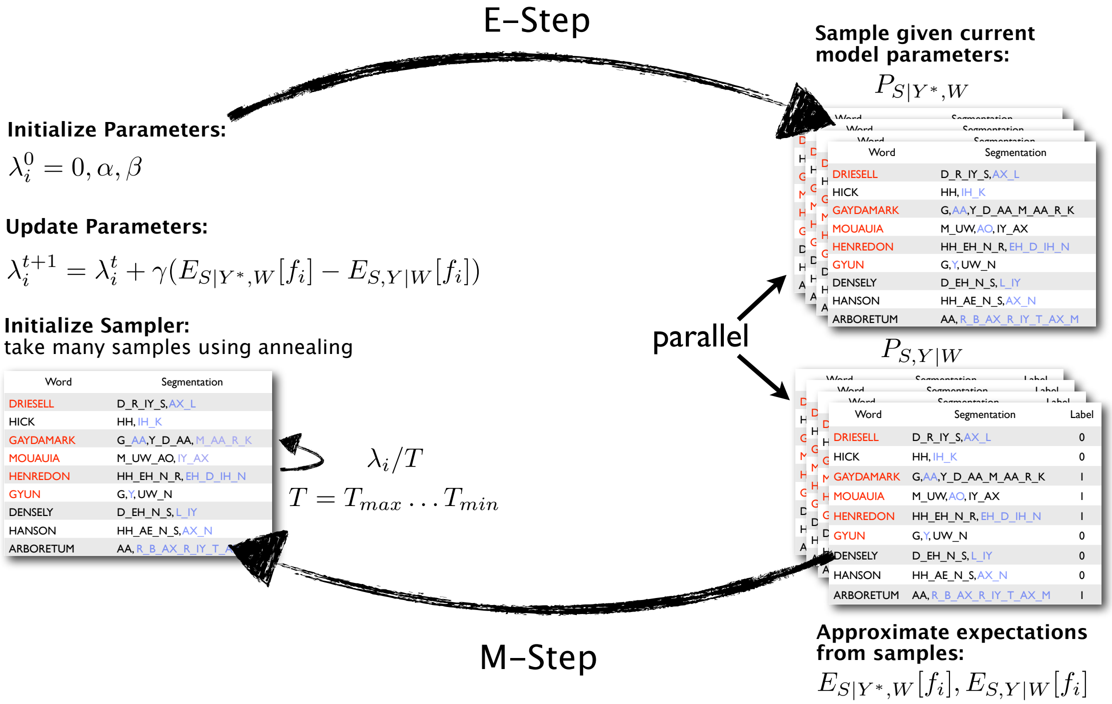
Algorithm 1 Training

```
Input: Lexicon $L$ from training text $W$, Dictionary $D$, Mapping $M$, L2S pronunciations, Annealing temp $T$.
Initialization:
  Assign label $y^*_m = M[w_m]$. $\bar{\lambda}_0 = \bar{0}$
  $S_0$ = random segmentation for each word in $L$.

  for $i = 1$ to K
    /* E-Step */
    $S_i$ = bestSegmentation(T, $\lambda_{i-1}$, $S_{i-1}$).
    for $k = 1$ to NumSamples
      $(S'_k, Y'_k)$ = sampleSL($P(Y, S_i|W)$,$Q(Y, S_i|W)$)
      $\tilde{S}_k$ = sampleS($P(S_i|Y^*, W)$,$Q(S_i|Y^*, W)$)
    end for

    /* M-Step */
    $E_{S,Y|W}[f_i] = \frac{1}{NumSamples} \sum_k f_{\sigma,l}[S'_k, Y'_k]$
    $E_{S|Y^*,W}[f_{\sigma,l}] = \frac{1}{NumSamples} \sum_k f_{\sigma,l}[\tilde{S}_k, Y^*]$
    $\bar{\lambda}_i = \bar{\lambda}_{i-1} + \gamma \nabla L_{\bar{\lambda}}(Y^*|W)$
  end for

  S = bestSegmentation(T, $\lambda_K$, $S_0$)
  Output: Lexicon $L_o$ from $S$
```

Algorithm 2 sampleSL$(P(S,Y|W), Q(S,Y|W))$

```
for $m = 1$ to M (NumWords)
    $(s'_m, y'_m)$ = Sample segmentation/label pair for word $w_m$ according to $Q(S,Y|W)$
    $Y' = \{y_1 \dots y_{m-1} y'_m y_{m+1} \dots y_M\}$
    $S' = \{s_1 \dots s_{m-1} s'_m s_{m+1} \dots s_M\}$
    $\alpha$=min $\left(1, \frac{\sum_{\sigma \in S'} |\sigma|}{\sum_{\sigma \in S} |\sigma|}\right)$
    with prob $\alpha : y_{m,k} = y'_m, s_{m,k} = s'_m$
    with prob $(1-\alpha) : y_{m,k} = y_m, s_{m,k} = s_m$
end for
return $(S'_k, Y'_k) = [(s_{1,k}, y_{1,k}) \dots (s_{M,k}, y_{M,k})]$
```

Figure 3.3 summarizes the algorithm pictorially. We start with a default value for the model parameters $\Lambda = \lambda_i, \alpha, \beta$, where $i \in \{1, 2, \dots, (|\sigma| + |c|)\}$, where λ_i represent the weights associated with each one of the features (unique sub-words and unique contexts). The hyper-parameters α and β are tuned on a development set. Given a value for our parameters, we initialize the sampler using annealing, varying the temperature from T_{max} to T_{min} (known as *burn-in* period). In the *E-step*, we generate two sets of samples according to distributions $P_{S|Y^*,W}$ and $P_{S,Y|W}$. $E_{S|Y^*,W}[f_i]$ and $E_{W,Y|W}[f_i]$ are computed from these samples using Eq. (3.7) and Eq. (3.8) respectively. These expectations are used in the *M-step* to update the model parameters using gradient ascent. After M iterations, we take the most likely segmentation of the corpus given the parameters Λ, and extract all sub-words σ_i associated with OOVs. These sub-words are predictive of the OOV class-label.


E–Step
Initialize Parameters:
$\lambda_i^0 = 0, \alpha, \beta$
Update Parameters:
$\lambda_i^{t+1} = \lambda_i^t + \gamma(E_{S|Y^*,W}[f_i] - E_{S,Y|W}[f_i])$
Initialize Sampler:
take many samples using annealing
λ_i/T
$T = T_{max} \dots T_{min}$
Sample given current model parameters:
$P_{S|Y^*,W}$
parallel
$P_{S,Y|W}$
Approximate expectations from samples:
$E_{S|Y^*,W}[f_i], E_{S,Y|W}[f_i]$
M–Step

Word	Segmentation
DRIESELL	D_R_IY_S, AX_L
HICK	HH, IH_K
GAYDAMARK	G_AA, Y_D_AA, M_AA_R_K
MOUAUIA	M_UW_AO, IY_AX
HENREDON	HH_EH_N_R, EH_D_IH_N
GYUN	G, Y, UW_N
DENSELY	D_EH_N_S, L_IY
HANSON	HH_AE_N_S, AX_N
ARBORETUM	AA, R_B_AX_R_IY_T_A

Word	Segmentation
DRIESELL	D_R_IY_S, AX_L
HICK	HH, IH_K
GAYDAMARK	G, AA, Y_D_AA_M_AA_R_K
MOUAUIA	M_UW, AO, IY_AX
HENREDON	HH_EH_N_R, EH_D_IH_N
GYUN	G, Y, UW_N
DENSELY	D_EH_N_S, L_IY
HANSON	HH_AE_N_S, AX_N
ARBORETUM	AA, R_B_AX_R_IY_T_AX_M

Word	Segmentation	Label
DRIESELL	D_R_IY_S, AX_L	0
HICK	HH, IH_K	0
GAYDAMARK	G, AA, Y_D_AA_M_AA_R_K	1
MOUAUIA	M_UW, AO, IY_AX	1
HENREDON	HH_EH_N_R, EH_D_IH_N	1
GYUN	G, Y, UW_N	0
DENSELY	D_EH_N_S, L_IY	0
HANSON	HH_AE_N_S, AX_N	0
ARBORETUM	AA, R_B_AX_R_IY_T_AX_M	1

Figure 3.3: The training procedure to learn sub-words. We start with a default value for the model parameters $\Lambda = \lambda_i, \alpha, \beta$, where $i \in \{1, 2, \dots, (|\sigma| + |c|)\}$, where λ_i represent the weights associated with each one of the features. Given a value for our parameters, we initialize the sampler using annealing. In the *E-step*, we generate two sets of samples according to distributions $P_{S|Y^*,W}$ and $P_{S,Y|W}$. $E_{S|Y^*,W}[f_i]$ and $E_{W,Y|W}[f_i]$ are computed from these samples and used in the *M-step* to update the model parameters using gradient ascent. After M iterations, we take the most likely segmentation of the corpus given the parameters Λ, and extract all sub-words σ_i associated with OOVs. These sub-words are predictive of the OOV class-label.

3.4 Hybrid Models

In this section we describe how to integrate the learned sub-words in a hybrid recognizer. Our model (Section 3.2.1) can be applied to model OOVs by using a dictionary $\mathcal{L}$ to label words as IV ($y_i = 0$ if $w_i \in \mathcal{L}$) and OOV ($y_i = 1$ if $w_i \notin \mathcal{L}$). This results in a labeled corpus, where the labeling sequence Y indicates the presence of out-of-vocabulary words (OOVs). For comparison we evaluate a baseline method (Rastrow et al., 2009a) for selecting units.

Given a sub-word lexicon, the word and sub-words are combined to form a hybrid language model (LM) to be used by the LVCSR system. This hybrid LM captures dependencies between word and sub-words. In the LM training data, all OOVs are represented by the smallest number of sub-words which corresponds to their pronunciation. Pronunciations for all OOVs are obtained using grapheme to phone models (Stanley F. Chen, 2003).

A greedy search algorithm converts OOVs to sub-words: iteratively assign the longest possible matching sub-word to cover the OOV term. For example, if the word, HAMDI has pronunciation /HH/AE/M/D/IY and sub-words HH_AE_M and D_IY are in the sub-word inventory but HH_AE_M_D and HH_AE_M_D_IY are not, then the sub-word representation for the term would be /HH_AE_M D_IY/.

The output can be the one-best transcripts, lattices or confusion networks. While lattices contain more information, they are harder to process; confusion networks offer a trade-off between richness and compactness (Mangu, Brill, and Stolcke, 1999). Confusion networks represent compact representations of the recognizer's hypotheses. For an utterance the

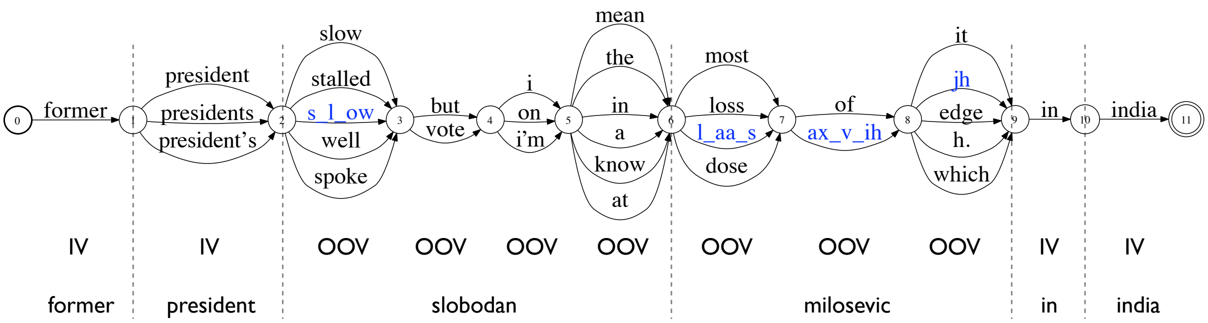


Figure 3.4: Example confusion network from the hybrid system with OOV regions. Hypothesis are ordered by decreasing value of posterior probability. Best hypothesis is the concatenation of the top word/fragments in each bin. We omit posterior probabilities due to spacing.

confusion network is composed of a sequence of *confused regions*, indicating the set of most likely word/sub-word hypotheses uttered and their posterior probabilities.[7].

Figure 3.4 depicts a confusion network decoded by the hybrid system for a section of an utterance in our test-set. Below the network we present the reference transcription. In this example, two OOVs were uttered: “slobodan" and “milosevic" and decoded as four and three in-vocabulary words, respectively. A *confused region* (also called “bin") corresponds to a set of competing hypothesis between two nodes. The goal is to correctly label each of the “bins" as OOV or IV. Note the presence of both fragments (e.g. `s_l_ow`, `l_aa_s`) and words in some of the hypothesis bins.

[7] $P(w_i|A)$: posterior probability of word i given the acoustic signal. This posterior includes language model and acoustic model scores normalized over the set of competing words in that particular time interval (cohort), as described in Mangu et al. (1999)

3.5 OOV Detection

To evaluate our model for learning sub-word units, we consider the task of out-of-vocabulary (OOV) word detection. The sub-words produced detect the presence of OOVs directly. Once identified, OOVs can be flagged for annotation and addition to the system's vocabulary, or OOV segments can be transcribed phonetically and their orthography automatically recovered (see Chapter 5), creating an open vocabulary LVCSR system. Identified OOVs prevent error propagation in the application pipeline as we demonstrate in Chapter 6 and 7.

3.5.1 Baseline OOV detector

Our baseline system is the Maximum Entropy model with features from hybrid and confidence estimation models proposed by Rastrow et al. (2009a). Based on hybrid models, this approach models OOVs by constructing a hybrid system which combines words and sub-word units. Since sub-words represent OOVs while building the hybrid LM, the existence of sub-words in ASR output indicate an OOV region. A simple solution to the OOV detection problem would then be reduced to a search for the sub-words in the output of the ASR system.

This approach also includes properties from confidence estimation systems. Using a hybrid LVCSR system, they obtain *confusion networks*. For any bin of the confusion network, Rastrow et al. combine features from that region using a binary Maximum Entropy classifier. Two effective indications of OOVs are the existence of sub-words (Eq. 3.11) and high entropy in a network region (Eq. 3.12), both of which are used as features in their model:

$$\text{Sub-word Posterior} = \sum_{\sigma \in t_j} p(\sigma|t_j) \tag{3.11}$$

$$\text{Word-Entropy} = -\sum_{w \in t_j} p(w|t_j) \log p(w|t_j) \tag{3.12}$$

where t_j is the current bin in the confusion network and σ is a sub-word in the hybrid dictionary. Improving the sub-word unit lexicon, improves the quality of the confusion networks for OOV detection.

We obtained confusion networks for a standard word based system and the hybrid system described above. We re-implemented the above features, obtaining nearly identical results to Rastrow et al. using Mallet's MaxEnt classifier (McCallum, 2002). [8] All real-valued features were normalized and quantized using the uniform-occupancy partitioning described in White, Droppo, Acero, and Odell (2007).[9] The MaxEnt model is regularized using a Gaussian prior ($\sigma^2 = 100$), but we found results generally insensitive to σ.

3.5.2 Baseline Unit Selection

We used Rastrow et al. (2009a) as our baseline unit selection method. To select sub-words, they proposed a data driven approach where the language model training text (de-

[8] Small differences are due to a change in MaxEnt library.

[9] All experiments use 50 partitions with a minimum of 100 training values per partition.

scribed in Section 2.2.2) is converted into phones using the dictionary or a letter-to-sound model for OOVs. A N-gram phone LM is estimated on this data and pruned using a relative entropy based method. The selected sub-words represent the most likely phone N-grams in the training text – ranging from unigrams to 5-gram phones. The hybrid lexicon includes resulting sub-words and the 83K word lexicon.

3.6 Experimental Setup

For our experiments, we used the 100 hours English Broadcast News OOVCORP data set described in Section 2.2.2. The LVCSR system used was the IBM Speech Recognition Toolkit (Soltau et al., 2010) described in our experimental setup section (Section 2.2.2). The 100 hours were excluded from training and divided into 5 hours of training for the OOV detector and 95 hours of test. Note that the OOV detector training set is different from the LVCSR training set.

We also use a hybrid LVCSR system, combining word and sub-word units obtained from either our approach or a state-of-the-art baseline approach (Rastrow et al., 2009a) (Section 3.5.2). Our hybrid system's lexicon has 83,000 words and 5,000 or 10,000 sub-words. Note that the word vocabulary is common to both systems and only the sub-words are selected using either approach. 1290 words are OOVs to both the word and hybrid systems.

In addition we report OOV detection results on a MIT lectures data set described in Section 2.2.2, consisting of 3 hours with a 1.5% OOV rate. These were divided into 1 Hr for training the OOV detector and 2 hours for testing. Note that the LVCSR system is trained on Broadcast News data. This out-of-domain test-set help us evaluate the cross-domain performance of the proposed and baseline hybrid systems. OOVs in this data set correspond mainly to technical terms in computer science and math. e.g. ALGORITHM, DEBUG, COMPILER, LISP, while in Broadcast News (OOVCORP) they correspond mainly to named-entities (e.g. NATALIE, PUTIN, QAEDA, HOLLOWAY, COROLLARIES, HYPERLINKED).

3.6.1 Learning parameters

For learning the sub-words we randomly selected from training 5,000 words which belong to the 83K vocabulary and 5,000 OOVs[10]. For development we selected an additional 1,000 IV and 1,000 OOVs. This was used to tune our model hyper parameters (set to $\alpha = -1$, $\beta = -20$). There is no overlap of OOVs in training, development and test sets. All feature weights were initialized to zero and had a Gaussian prior with variance $\sigma = 100$. Each of the words in training and development were converted to their most-likely pronunciation using the dictionary for IV words or the L2S model for OOVs. In this work we ignore pronunciation variability and simply consider the most likely pronunciation

[10] This was used to obtain the 5K hybrid system. To learn sub-words for the 10K hybrid system we used 10K in-vocabulary words and 10K OOVs. All words were randomly selected from the LM training text.

for each word. It is straightforward to extend to multiple pronunciations by first sampling a pronunciation for each word and then sampling a segmentation for that pronunciation.

The learning rate was $\gamma_k = \frac{\gamma}{(k+1+A)^\tau}$, where k is the iteration, A is the stability constant (set to $0.1K$), $\gamma = 0.4$, and $\tau = 0.6$. We used $K = 40$ iterations for learning and 200 samples to compute the expectations in Eq. 3.6. The sampler was initialized by sampling for 500 iterations with deterministic annealing for a temperature varying from 10 to 0 at 0.1 intervals. Final segmentations were obtained using $10,000$ samples and the same temperature schedule. We limit segmentations to those including units of at most 5 phones to speed sampling with no significant degradation in performance. We observed improved performance by dis-allowing whole word units.

3.6.2 Evaluation

We obtain confusion networks from both the word and hybrid LVCSR systems. We align the LVCSR transcripts with the reference transcripts and tag each confusion region as either IV or OOV. The OOV detector classifies each region in the confusion network as IV/OOV.

Previous research reported OOV detection accuracy on all test data. However, once an OOV word has been observed in the training data for the OOV detector, even if it never appeared in the LVCSR training data, it is no longer truly OOV. Therefore, in the sections that follow we report unobserved OOV accuracy: OOV words that do not appear in either the OOV detector's or the LVCSR's training data. While this penalizes our results, it is a more informative metric of true system performance.

We compare the performance of the baseline hybrid system (Section 3.5.2) and the proposed hybrid system with units learned by our model in terms of:

- Hits: sub-word units predicted in OOV regions, and False Alarms: sub-word units predicted for in-vocabulary words

- OOV detection performance. We present results using standard detection error tradeoff (DET) curves (Martin, Doddington, Kamm, Ordowski, and Przybocky, 1997). DET curves measure tradeoffs between misses and false alarms and can be used to determine the optimal operating point of a system. The x-axis varies the false alarm rate (false positive) and the y-axis varies the miss (false negative) rate; lower curves are better.

- Phone Error Rate (PER): this is computed using Eq. (2.1) on page 19, where the reference and the automatic transcription are first converted to their phonetic representation using the dictionary. PER evaluates whether the sub-word units predicted in OOV regions resemble the true pronunciation of the OOV. Furthermore, improving PER is important for downstream indexing applications, specially when the query terms include OOVs. We do not evaluate WER because the units used are phonetic, thus their concatenation does not provide the correct spelling of the OOV uttered.

We present WER results in Chapter 5, after describing our approach to recover the correct spelling of new words for transcription.

3.7 Results

Table 3.2 shows the percent of Hits: sub-word units predicted in OOV regions, and False Alarms: sub-word units predicted for in-vocabulary words in OOVCORP. We can see that the proposed system increases the Hits by roughly 8% absolute, while increasing the False Alarms by 0.3%.

Table 3.3 also shows the performance on MIT Lectures. Note that both the sub-word lexicon and the LVCSR models were trained on Broadcast News data, hence this data-set evaluates the robustness of learned sub-words across domains. The OOVs in these domains are quite different: MIT Lectures' OOVs correspond to technical computer science and math terms, while in Broadcast News they are mainly named-entities. However, similar to the results in OOVCORP, we found that the learned sub-words provide larger coverage of OOV regions in MIT Lectures domain. These results suggest that the proposed sub-words are not simply modeling the training OOVs (named-entities) better than the baseline sub-words, but also describe better novel unexpected words.

Interestingly, the average sub-word length for the proposed units exceeded that of the baseline units by 0.3 phones (Baseline 10K average length was 3.20 with standard deviation 0.82, while that of Learned Units 10K was 3.58 with standard deviation 0.87.[11]

[11] This difference is significant at 99% under the T-Test.

Table 3.4 shows example of the predicted sub-words for the baseline and proposed systems.

Hybrid System	**No. of Sub-words**	**Hits (%)**	**FAs (%)**
Baseline	5k	18.25	1.49
Learned Units	5k	26.78	1.78
Baseline	10k	24.26	1.82
Learned Units	10k	28.96	1.92

Table 3.2: Coverage of OOV regions by sub-words in OOVCORP.

Hybrid System	**No. of Sub-words**	**Hits (%)**	**FAs (%)**
Baseline	5k	17.03	2.33
Learned Units	5k	22.14	2.72
Baseline	10k	21.41	2.55
Learned Units	10k	21.89	2.66

Table 3.3: Coverage of OOV regions by sub-words in MIT Lectures.

OOV detection

We also evaluate OOV detection performance as described in Section 3.5. The model uses two features for each region: Word Entropy and Sub-word Posterior (Eqs. 3.11 and 3.12) (Figure 7.1). Predictions at different FA rates are obtained by varying a probability threshold. Both systems used the same features as input to the MaxEnt model. The only different between these systems is the sub-word lexicon used in the hybrid system to decode the test-set.

At a 5% FA rate, our system (`Learned Units 5k`) reduces the miss OOV rate by 6.3% absolute over the baseline (`Baseline 5k`) when evaluating all OOVs. For unobserved OOVs, it achieves 3.2% absolute improvement. A larger lexicon (`Baseline 10k` and `Learned Units 10k`) shows similar relative improvements. Note that the features

Word	Baseline sub-words	Proposed sub-words
adrianna	ey_d, r_iy, ae_n, ax	ey_d, r_iy_ae_n_ax
yakusuni	y_ax, k_uw, s_uw, n_iy	y_ax_k_uw, s_uw, n_iy
quicktime	k_w, ih_k, t_ay, m	k_w_ih, k_t, ay_m
natascha	n_ax, t_aa, sh_ax	n_ax, t_aa, sh_ax
lieutanant	l_uw, t_ae, n_ih, n_t	l_uw_t, ae_n, ih_n_t

Table 3.4: Example representations of OOVs using the Baseline and Learned Subwords.

used so far do not necessarily provide an advantage for unobserved versus observed OOVs, since they ignore the decoded word/sub-word sequence and only include posterior probability information from the decoded networks.

Figure 3.6 shows the OOV detection results in the MIT Lectures data set. For unobserved OOVs, the proposed system (`Learned Units 10k`) reduces the miss OOV rate by 4% with respect to the baseline (`Baseline 10k`) at a 5% FA rate for unseen OOVs. We further improve performance in the next chapter.

Improved Phonetic Transcription

We consider the hybrid lexicon's impact on Phone Error Rate (PER) with respect to the reference transcription. The reference phone sequence is obtained by doing *forced alignment* of the audio stream to the reference transcripts using acoustic models. This provides an alignment of the pronunciation variant of each word in the reference and the recognizer's one-best output. The aligned words are converted to the phonetic representation using the dictionary.

Table 3.6 presents PERs for the word and different hybrid systems. As previously reported (Rastrow et al., 2009b), the hybrid systems achieve better PER, specially in OOV regions since they predict sub-word units for OOVs. Our method achieves modest improvements in PER compared to the hybrid baseline in OOVCORP. No statistically significant improvements in PER were observed on MIT Lectures. It is also worth mentioning that the PER results show that the output of hybrid systems are richer and more useful for downstream applications such as Spoken Term Detection (STD).

System	**No. of Subwords**	**OOV** (%)	**IV** (%)	**All** (%)
Word	0	1.62	6.42	8.04
Hybrid: Baseline	5k	1.56	6.44	8.01
Hybrid: Baseline	10k	1.51	6.41	7.92
Hybrid: Learned Units	5k	1.52	6.42	7.94
Hybrid: Learned Units	10k	1.45	6.39	7.85

Table 3.5: Phone Error Rate results for OOVCORP.

System	**No. of Subwords**	**OOV** (%)	**IV** (%)	**All** (%)
Word	0	1.50	18.26	19.76
Hybrid: Baseline	5k	1.47	18.36	19.84
Hybrid: Baseline	10k	1.47	18.35	19.82
Hybrid: Learned Units	5k	1.47	18.31	19.78
Hybrid: Learned Units	10k	1.45	18.31	19.76

Table 3.6: Phone Error Rate results for MIT Lectures.

3.8 Related Work

OOV detection for ASR output can be categorized into two broad groups: 1) *hybrid (filler) models*: which explicitly model OOVs using either filler, sub-words, or generic word models (Bazzi, 2002; Schaaf, 2001; Bisani and Ney, 2005; Klakow et al., 1999; Wang, 2009); and 2) *confidence-based approaches*: which label un-reliable regions as OOVs based on different confidence scores, such as acoustic scores, language models, and lattice scores (Lin et al., 2007; Burget et al., 2008; Sun et al., 2001; Wessel et al., 2001).

While confidence scores are popular, it is often difficult to determine if low confidence indicates a recognition error due to an OOV or some other cause. Hybrid models target OOV errors and have other advantages such as achieving lower phone error rates in OOV regions and facilitating recovery of the missing word.

The proposed un-supervised segmentation approach presented in this chapter was inspired by the work of Poon et al. (2009). Their work presents a log-linear model for un-supervised morphological segmentation similar to the one we describe. The main differences between their model and the one proposed here are:

1. Their approach learns the joint probability $P(S, W)$ for a segmentation S and the text W, while ours takes into consideration the label sequence Y, specifically we model $P(Y, S|W)$. This motivates the model to find segmentations predictive of class label Y optimizing the segmentation for a particular labeling task;

2. In our model, the text W is on the right-hand side of the conditioning: $P(S, Y|W)$, which means we don't model dependencies in the text W making inference simpler;

3. We further optimize inference by proposing an efficient inference procedure using Finite State Methods and the Metropolis Hastings algorithm. This achieves up to an order of magnitude speed-up for sampling;

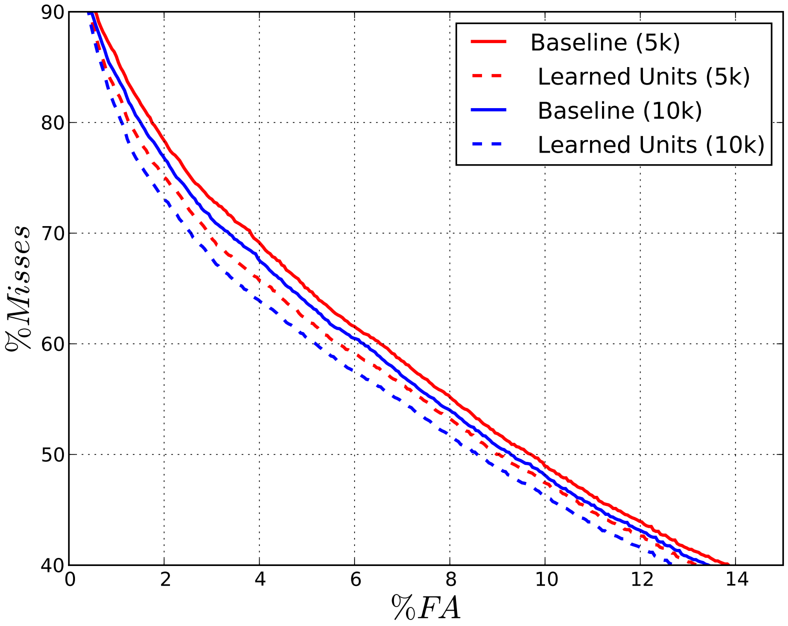
4. The objective function in Poon et al. (2009) is different from ours, since we maximize the likelihood of the observed labeled sequence Y^*, while they maximize the likelihood of the text W;

5. This approach was used for morphological segmentation and not speech recognition.

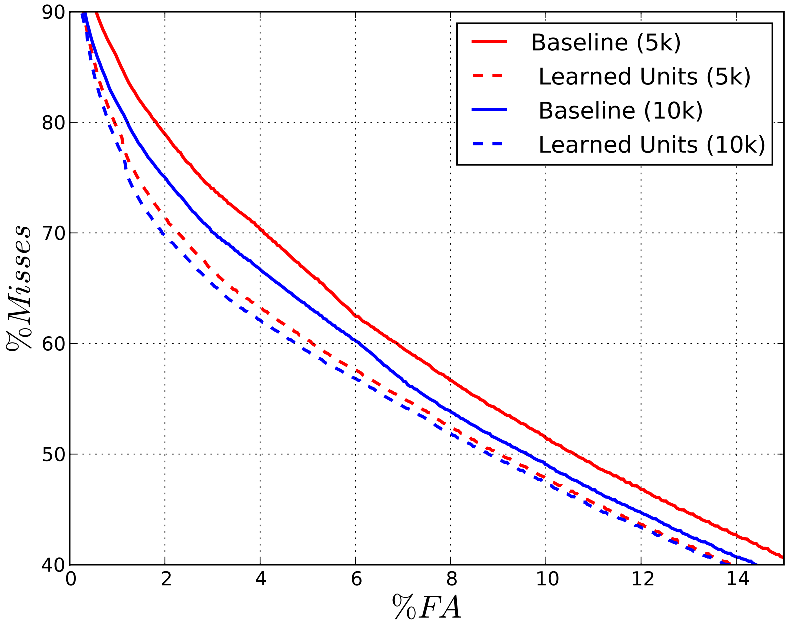
Creutz and Lagus (2002) also proposed an unsupervised segmentation approach for finding morphological units, and applied this model for speech recognition in morphologically rich languages. Their approach slightly simplified maximizes the posterior probability of the lexicon given the corpus: $P(\text{lexicon}|\text{corpus}) \propto P(\text{lexicon})P(\text{corpus}|\text{lexicon}) = \prod_{\text{letters } \alpha} P(\alpha) \cdot \prod_{\text{morphs } \mu} P(\mu)$, where letter and morph probabilities are maximum likelihood estimates. This approach is also inspired by the MDL principle, however it differs from ours in many respects: 1) it is a generative model while ours is discriminative; 2) it does not take into consideration the context of units when deriving a segmentation; and 3) it does not model a labeling sequence Y, so the segmentation is not optimized for a given task.

3.9 Conclusions

Our probabilistic model learns sub-word units for hybrid speech recognizers by segmenting a text corpus while exploiting side information. The learned units improve detection of OOV regions by 6.3% absolute at a 5% FA rate on an English Broadcast News task, and by 4% absolute on an out-of-domain MIT Lectures data-set. Furthermore, we have confirmed previous work that hybrid systems achieve better phone accuracy, and our model makes modest improvements over a baseline with a similarly sized sub-word lexicon. Additionally, we used a simple but effective solution to speed inference using Metropolis-Hastings. This reduces the sampling time by an order of magnitude with no degradation in performance. In the next chapter, we will revise results on OOV Detection for Broadcast News and MIT Lectures after we introduce a novel approach for OOV detection.

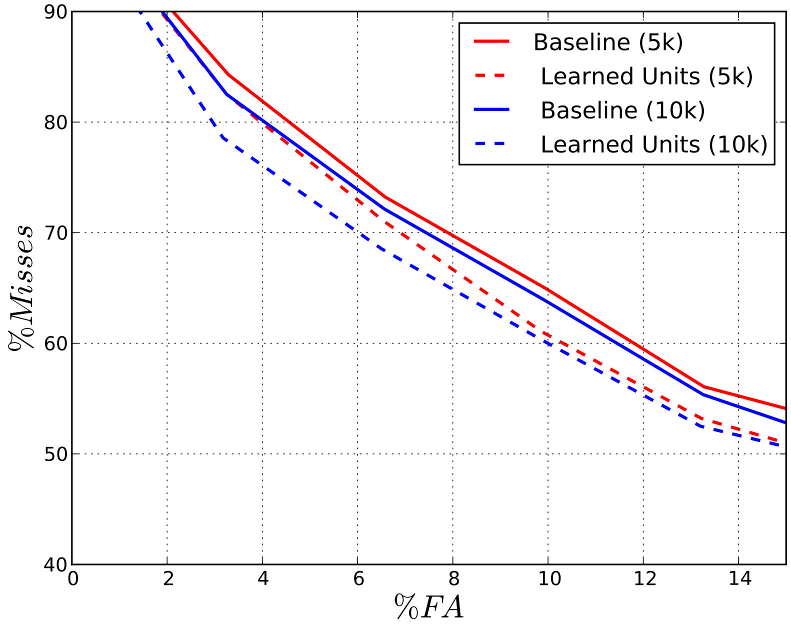


(a)

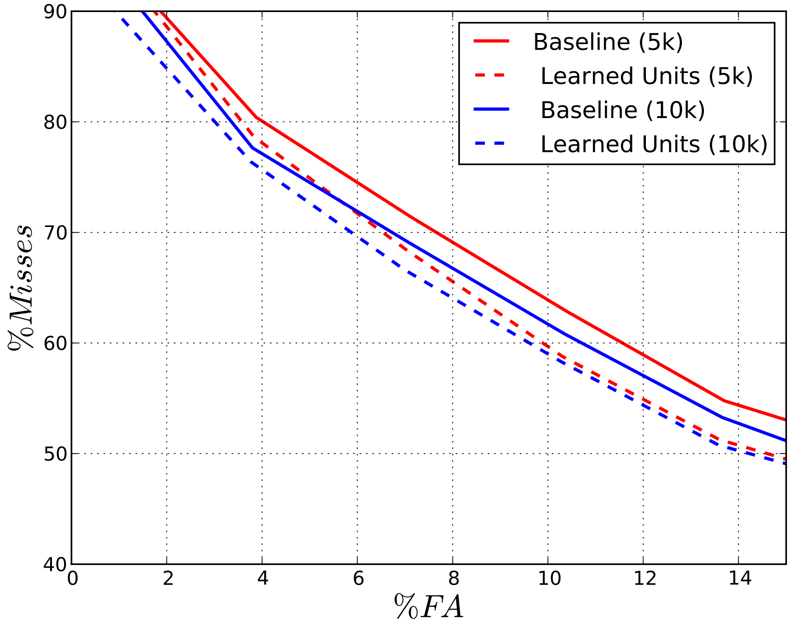


(b)

Figure 3.5: DET curves for OOV detection using baseline hybrid systems for different lexicon size and proposed discriminative hybrid system on **OOVCORP** data set. Evaluation on **un-observed** OOVs (a) and **all** OOVs (b).



(a)



(b)

Figure 3.6: DET curves for OOV detection using baseline hybrid systems for different lexicon size and proposed discriminative hybrid system on **MIT Lectures** data set. Evaluation on **un-observed** OOVs (a) and **all** OOVs (b).

Chapter 4

Exploiting Context for Out-of-Vocabulary Detection

4.1 Introduction

In Chapter 3 we focused on improving the sub-word lexicon to include in a hybrid recognizer. The resulting hybrid output showed improved performance for OOV detection compared to a state-of-the-art hybrid system. In this chapter we assume the hybrid system is fixed and we focus on the model used for detecting OOV regions.

The state-of-the-art OOV detection model introduced in Chapter 3 (Rastrow et al., 2009a) was a Maximum Entropy classifier with features from hybrid and confidence estimation models described in detail in Section 3.5.1. This OOV detection system, as many other confidence based systems (Hazen and Bazzi, 2001; Bazzi, 2002; Lin et al., 2007;

Burget et al., 2008; White et al., 2008), treats OOV detection as a binary classification task; each region is independently classified using local information as IV (in-vocabulary) or OOV (out-of-vocabulary).

In this chapter we move beyond this independence assumption that considers regions independently for OOV detection. We treat OOV detection as a sequence labeling problem and add features based on the local lexical context of each region as well as global features from a language model using the entire utterance. Our results show that such information improves OOV detection and we obtain large reductions in error compared to the best previously reported results. Furthermore, our approach can be combined with any other confidence based metrics.

Our experimental setup and evaluation is identical to the one detailed in Chapter 3, Section 3.6. After briefly reviewing the baseline system, we generalize the framework to a sequence labeling problem, which includes augmenting the features from the local context, lexical context, and entire utterance. Each stage yields additional improvements over the baseline system. We conclude with a review of related work.

4.2 From Maximum Entropy to Conditional Random Fields

The baseline OOV detection system (Rastrow et al., 2009a) introduced in Section 3.5.1, combines features from hybrid and confidence-based models using a Maximum Entropy

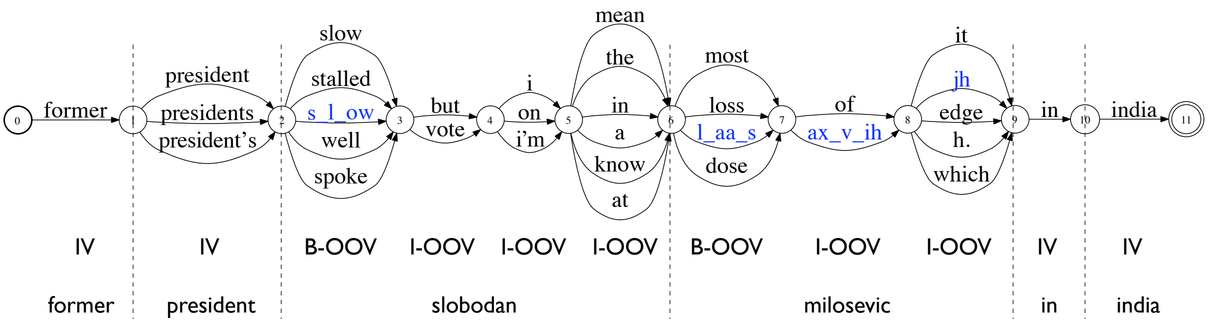


Figure 4.1: Example confusion network from the hybrid system with OOV regions and BIO encoding.

classifier. This approach assigns a label: IV (in-vocabulary) or OOV to each region in the confusion network produced by a LVCSR system.

As a classification algorithm, Maximum Entropy (MaxEnt) assigns a label to each region in the confusion network independently. However, OOV words tend to be recognized as two or more IV words, hence OOV regions tend to co-occur. In our running example (Figure 4.1), the OOV word "slobodan" was recognized as four IV words: "slow vote i mean". This suggests that sequence models, which jointly assign all labels in a sequence, may be more appropriate. Therefore, we begin incorporating context by moving from classification to sequence models.

MaxEnt classification models the target label as $p(y_i|\mathbf{x}_i)$, where y_i is a discrete variable representing the ith label ("IV" or "OOV") and $\mathbf{x}_i$ is a feature vector representing information for position i. The conditional distribution for y_i takes the form

$$p(y_i|\mathbf{x}_i) = \frac{1}{Z(\mathbf{x}_i)} \exp(\sum_{k=1}^{K} \lambda_k f_k(y_i, \mathbf{x}_i)) \ ,$$

$Z(\mathbf{x}_i)$ is a normalization term and $f(y_i, \mathbf{x}_i)$ is a vector of K features, such as those defined in Section 3.5.2. The model is trained discriminatively: parameters λ are chosen to maximize conditional data likelihood.

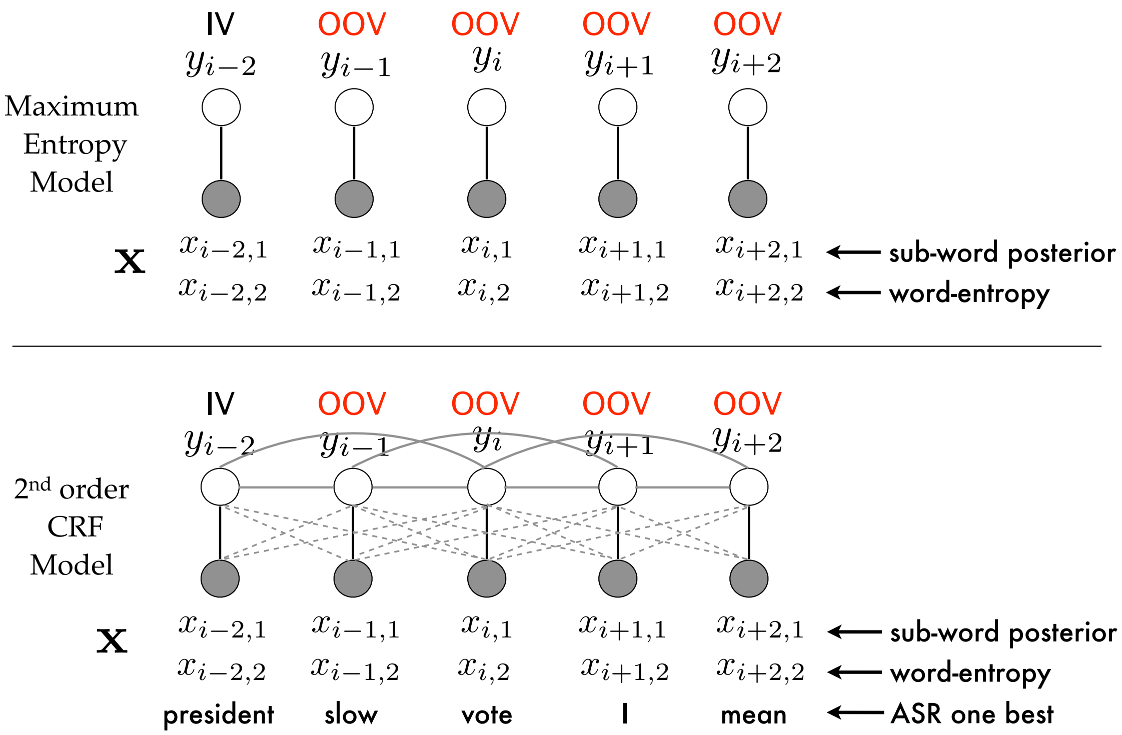
Conditional Random Fields (CRF) (Lafferty, McCallum, and Pereira, 2001) generalize MaxEnt models to sequence tasks. While having the same model structure as Hidden Markov Models (HMMs), CRFs are trained discriminatively and can use large numbers of correlated features. Their primary advantage over MaxEnt models is their ability to find an optimal labeling for the entire sequence rather than greedy local decisions. CRFs have been used successfully used in numerous text processing tasks and, though less popular, in speech it has been applied to sentence boundary detection (Liu, Stolcke, Shriberg, and Harper, 2005), phone classification and recognition (Gunawardana, Mahajan, Acero, and Platt, 2005; Morris and Fosler-Lussier, 2006, 2007), and for transcription using a LVCSR system (Ostendorf, Digalakis, and Kimball, 1996; Zweig and Nguyen, 2009).

A CRF models the entire label sequence $\mathbf{y}$ as:

$$p(\mathbf{y}|\mathbf{x}) = \frac{1}{Z(\mathbf{x})} \exp(\lambda \cdot F(\mathbf{y}, \mathbf{x})) \; ,$$

where $F(\mathbf{y}, \mathbf{x})$ is a global feature vector for input sequence $\mathbf{x}$ and label sequence $\mathbf{y}$ and $Z(\mathbf{x})$ is a normalization term.

Figure 4.2 illustrates the differences between a MaxEnt model (top - Baseline) and a Conditional Random Field model (bottom - Proposed) using their graphical model (GM)



IV
OOV
OOV
OOV
OOV
y_{i-2}
y_{i-1}
y_i
y_{i+1}
y_{i+2}
Maximum Entropy Model
$\mathbf{X}$
$x_{i-2,1}$ $x_{i-1,1}$ $x_{i,1}$ $x_{i+1,1}$ $x_{i+2,1}$ ← sub-word posterior
$x_{i-2,2}$ $x_{i-1,2}$ $x_{i,2}$ $x_{i+1,2}$ $x_{i+2,2}$ ← word-entropy
2nd order CRF Model
president slow vote I mean ← ASR one best

Figure 4.2: Maximum Entropy model (baseline) vs 2^{nd} order CRF model with lexical context (proposed) for OOV detection. Solid gray lines in the bottom section indicate added dependencies on the label sequence $\mathbf{Y}$. Dashed gray lines indicate added dependencies form observed lexical context: $\mathbf{X}$.

representations[1]. The gray nodes in the graph represent the observed random variables at each position i in the confusion network. In this case $x_{i,1}$ represents the sub-word posteriors Eq. (3.11) and $x_{i,2}$ is the word-entropy Eq. (3.12). The white nodes, y_i, represent the unobserved random variables we wish to predict. Each binary random variable y_i indicates the presence of OOV words at position i in the network: $y_i = 0$ (IV) or $y_i = 1$ (OOV). All connections between variables represent dependencies among them.

[1] A graphical model is a probabilistic model for which a graph indicates independence assumptions between the random variables. (Koller and Friedman, 2009)

As can be seen in Figure 4.2, the MaxEnt model only contains dependencies between the label at position y_i and the input variables in that same position $x_{i,1}, x_{i,2}$. That is, it classifies each position independently using only local information. Although in principle MaxEnt models can include dependencies to context input variables, we illustrate the actual dependencies assumed in the baseline model (Rastrow et al., 2009a). The CRF model includes dependencies to the input in the current position and the context (gray dashed lines), as well as dependencies between the labels (gray solid lines). In contrast, the second order CRF depicted in the bottom section of Figure 4.2 includes dependencies between neighboring labels up to two positions, modeling the sequence of unknown variables which are inter-dependent. In this case the value predicted for y_i depends on the values predicted for $\{y_{i-2}, y_{i-1}, y_{i+1}, y_{i+2}\}$.

In what follows, we formulate OOV-detection as a sequence labeling problem and introduce features that model local lexical context, and the entire utterance. Each enhancement is presented along with their additive gains over the baseline system.

4.3 Context for OOV Detection

We begin by including a minimal amount of local context in making OOV decisions: the predicted labels for adjacent confused regions (bins). This information helps when OOV bins occur in close proximity, such as successive OOV bins. This is indeed the case: in the OOV detector training data only 48% of OOV sequences contained a single bin;

sequences were of length 2 (40%), 3 (9%) and 4 (2%). Therefore, we expect that even a minimal amount of context based on the labels of adjacent bins will help.

A natural way of incorporating contextual information is through a CRF, which introduces dependencies between each label and its neighbors. If a neighboring bin is likely an OOV, it increases the chance that the current bin is OOV.

In sequence models, another technique for capturing contextual dependence is the label encoding scheme. In information extraction, where sequences of adjacent tokens are likely to receive the same tag, the beginning of each sequence receives a different tag from words that continue the sequence. For example, the first token in a person name is labeled `B-PER`, all subsequent tokens are labeled `I-PER`, and tokens which are not part of a person's name are labeled as `O` (outside). This is commonly referred to as BIO encoding (beginning, inside, outside). We applied this encoding technique to our task, labeling bins as either `IV` (in vocabulary), `B-OOV` (begin OOV) and `I-OOV` (inside OOV), as illustrated in Figure 4.1. This encoding allows the algorithm to identify features which might be more indicative of the beginning of an OOV sequence. We found that this encoding achieved a superior performance to a simple `IV`/`OOV` encoding. We therefore utilize the BIO encoding in all CRF experiments.

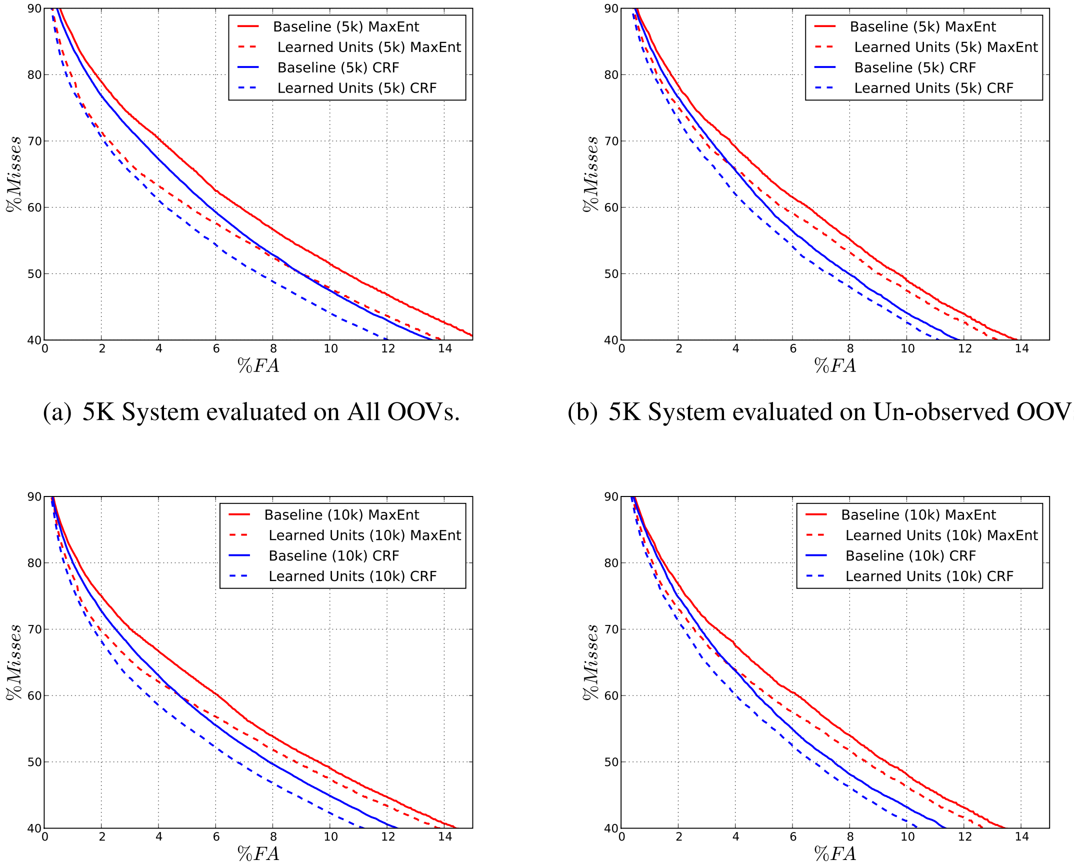
Another means of introducing context is through the order of the CRF model. A first order model ($n = 1$) adds dependencies only between neighboring labels, whereas an n order model creates dependencies between labels up to a distance of n positions. Higher order models capture length of label regions (up to length n). We found that a second order CRF gave the best performance. Higher order models did not provide any improvements.

In order to establish a comparative baseline, we first present results using the same features from the system described in Section 3.5.2 (Word-Entropy and Fragment-Posterior). For the CRF model, all real-valued features were normalized and quantized using the uniform-occupancy partitioning described in White et al. (2007).[2] Quantization of real valued features is standard for log-linear models as it allows the model to take advantage of non-linear characteristics of feature values and is better handled by the regularization term. For the MaxEnt model quantized and continuous features achieve comparable performance.

Figure 4.3 depicts DET curves for OOV detection for the MaxEnt baseline and second order CRFs with BIO encoding on unobserved OOVs and all OOVs in the test data.[3] We present results using the baseline sub-words and the sub-words we proposed in Chapter 3. For MaxEnt we used the predicted label probability and for CRFs the marginal probability of each bin's label. While the first order CRF achieves nearly identical performance to the MaxEnt baseline, the second order CRF shows a clear improvement. The second order model has up to 4.5% absolute improvement at 5% false alarm rate, despite using the identical features as the MaxEnt baseline. Even a small amount of context as expressed through local labeling decisions improves OOV detection. Since the hybrid system including 10K sub-words in the lexicon consistently outperform the one including only 5K sub-words, we present all future experiments using the 10K hybrid system.

[2] All experiments use 50 partitions with a minimum of 100 training values per partition.

[3] CRF experiments used the CRF++ package http://crfpp.sourceforge.net/



(a) 5K System evaluated on All OOVs.

(b) 5K System evaluated on Un-observed OOVs.

(c) 10K System evaluated on All OOVs.

(d) 10K System evaluate on Un-observed OOVs.

Figure 4.3: DET curves for OOV detection using a Maximum Entropy (MaxEnt) classifier vs a 2^{nd} order CRF. Solid curves indicate that the hybrid system was built using the Baseline units (Rastrow et al., 2009a) while dashed curves use Learned Units proposed in Chapter 3. All results are on `OOVCORP`.

4.4 Local Lexical Context

A popular approach in sequence tagging, such as information extraction or part of speech tagging, is to include features based on local lexical content and context. In detecting a name, both the lexical form "John" and the preceding lexical context "Mr." provide clues that "John" is a name. While we do not know the actual lexical items in the speech sequence, the speech recognizer output can be used as a best guess. In the example of Figure 4.1, the words "former president" are good indicators that the following word is either the word "of" or a name, and hence a potential OOV. Combining this lexical context with hypothesized words can help label the subsequent regions as OOVs (note that none of the hypothesized words in the third bin are "of", names, or nouns).

Words from the LVCSR decoding of the sentence are used in the CRF OOV detector. For each bin in the confusion network, we select the word with the highest probability (best hypothesis). We then add the best hypothesis word as a feature of the form: `current_word=X`. These features capture how the LVCSR system incorrectly recognizes OOV words. However, since detection is measured on unobserved OOVs, these features alone may not help.

Instead, we turn to lexical context, which includes correctly recognized IV words. We evaluate the following sets of features derived from lexical context:

- Current bin's best hypothesis. (Current-Word)

- Unigrams and bigrams from the best hypothesis in a window of 5 words around current bin. This feature ignores the best hypothesis in the current bin, i.e., `word[-2],word[-1]` is included, but `word[-1],word[0]` is not. (Context-Bigrams)

- Unigrams, bigrams, and trigrams in a window of 5 words around and including current bin. (Current-Trigrams)

- All of the above features. (All-Words)

- All above features and their stems.[4] (All-Words-Stems)

We added these features to the second order CRF with BIO encoding and baseline features (Figure 4.4). As expected, the current words did not improve performance on unobserved OOVs. When the current words are combined with the lexical context, they give a significant boost in performance: a 4.4% absolute improvement at 5% false alarm rate over the previous CRF system, and 9% over the MaxEnt baseline for un-observed OOVs. Interestingly, only combining context and current word gives a substantial gain. This indicates that OOVs tend to occur with certain distributional characteristics that are independent of the OOV word uttered (since we consider only unobserved OOVs), perhaps because OOVs tend to be named entities, foreign words, or rare nouns.

When evaluating all OOVs, the proposed features achieve 26% absolute improvement, reducing the OOV regions missed from 60% (`CRF`) to 34% (`+All-Words-Stems`). With

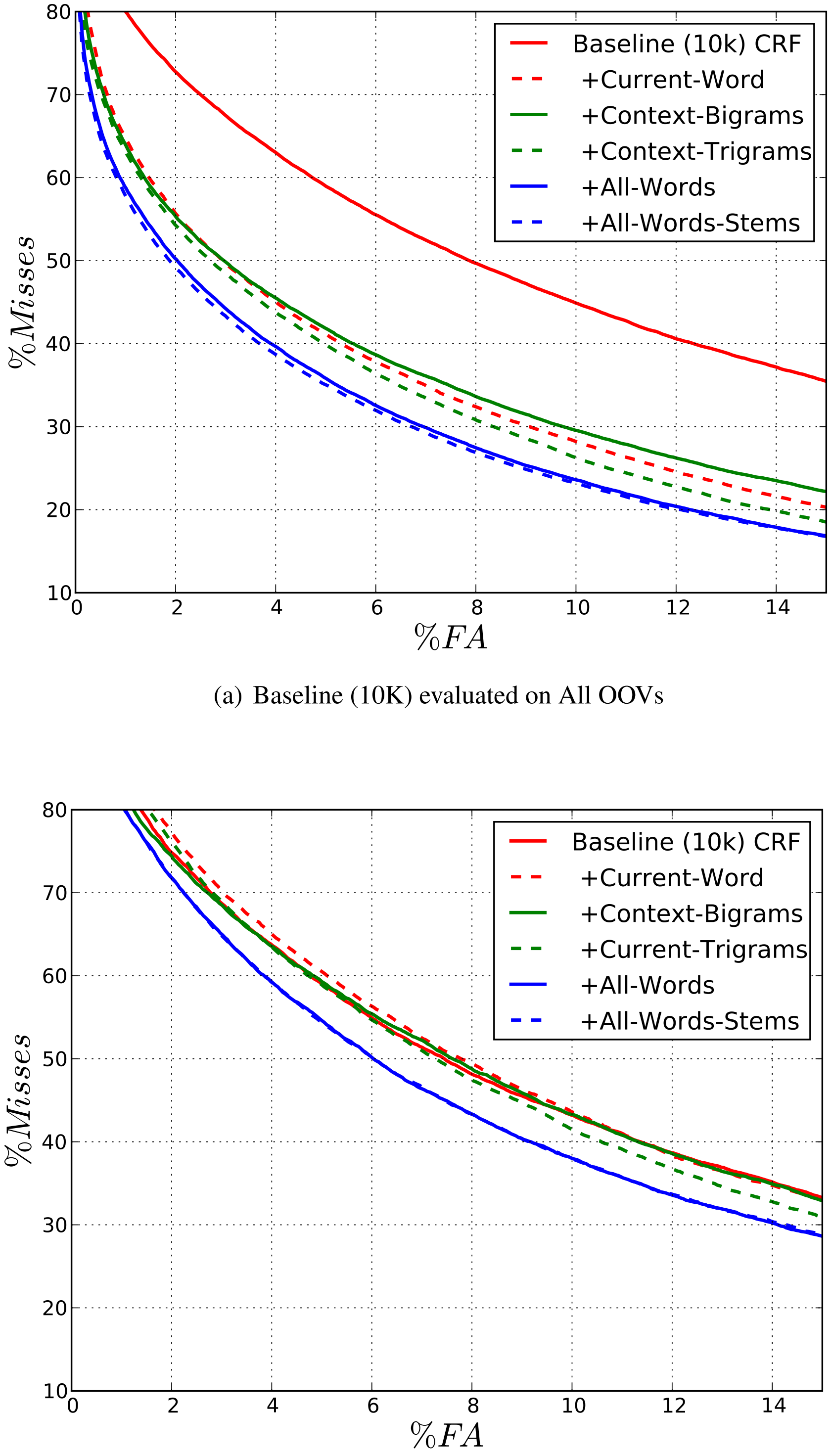
[4] To obtain stemmed words, we use the CPAN package: http://search.cpan.org/~snowhare/Lingua-Stem-0.83.

respect the the MaxEnt baseline the absolute gain is 31%. Since these features include the identify of the decoded words we can see a clear advantage when evaluating all OOVs (including OOVs observed in the OOV detector training set) due to repeated OOVs decoded using the same in-vocabulary word sequences. Adding the *stemmed* versions of the words improves performance by less than 1% absolute.

The importance of distributional features is well known for named entity recognition and part of speech tagging (Pereira, Tishby, and Lee, 1993). Other features such as substrings or baseline features (Word-Entropy, Fragment-Posterior) from neighboring bins did not provide further improvement. To the best of our knowledge previous work did not explore such distributional features for OOV detection.

4.5 Global Utterance Context

We now include features that incorporate information from the entire utterance. The probability of an utterance as computed by a language model is often used as a measure of fluency of the utterance. We also observe that OOV words tend to take very specific syntactic roles (more than half of them are proper nouns in the `OOVCORP` data set), which means the surrounding context will have predictive lexical and *syntactic* properties. Therefore, we use a syntactic language model.



(a) Baseline (10K) evaluated on All OOVs

(b) Baseline (10K) evaluated on Un-observed OOVs

Figure 4.4: A second order CRF (Section 4.3) and additional features including word identities from current and neighboring bins (Section 4.4).

4.5.1 Language Models

We evaluated both a standard trigram language model and a syntactic language model (Filimonov and Harper, 2009a). The syntactic model estimates the joint probability of the word and its syntactic tag based on the preceding words and tags. The probability of an utterance w_1^n of length n is computed by summing over all latent syntactic tag assignments:

$$p(utt) = p(w_1^n) = \sum_{t_1 \ldots t_n} \prod_{i-1}^{n} p(w_i, t_i | w_1^{i-1}, t_1^{i-1}) \tag{4.1}$$

where w_i and t_i are the word and tag at position i, and w_1^{i-1} and t_1^{i-1} are sequences of words and tags of length $i-1$ starting a position 1. The model is restricted to a trigram context, i.e., $p(w_i, t_i | w_{i-2}^{i-1}, t_{i-2}^{i-1})$; experiments that increased the order yielded no improvement.

We trained the language model on 130 million words from Hub4 CSR 1996 (Garofolo, Fiscus, Fisher, and Pallett, 1996). The corpus was parsed using a modified Berkeley parser (Huang and Harper, 2009) and tags extracted from parse trees incorporated the word's POS, the label of its immediate parent, and the relative position of the word among its siblings. [5] The parser required separated contractions and possessives, but we recombined those words after parsing to match the LVCSR tokenization, merging their tags. Since we are considering OOV detection, the language model was restricted to LVCSR system's vocabulary.[6]

[5] The *parent* tagset of Filimonov and Harper (2009a).

[6] Thanks to Denis Filimonov for providing these features.

We also used the standard fourgram LM for reference. It was trained on the same data and with the same vocabulary using the SRILM toolkit[7]. We used interpolated modified Kneser-Ney smoothing.

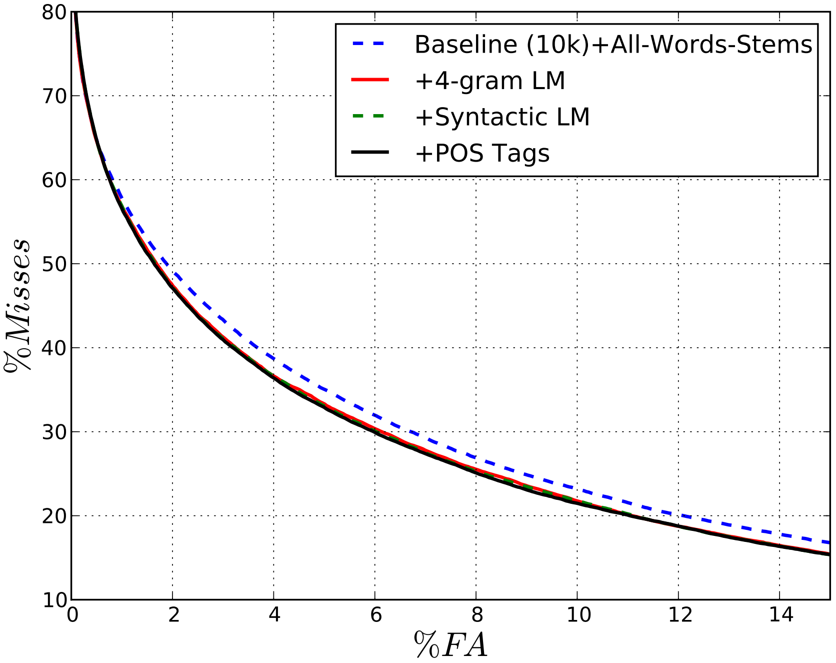
4.5.2 Language Model Features

We designed features based on the entire utterance using the language model to measure how the utterance is effected by the current token: whether the utterance is more likely given the recognized word or some OOV word.

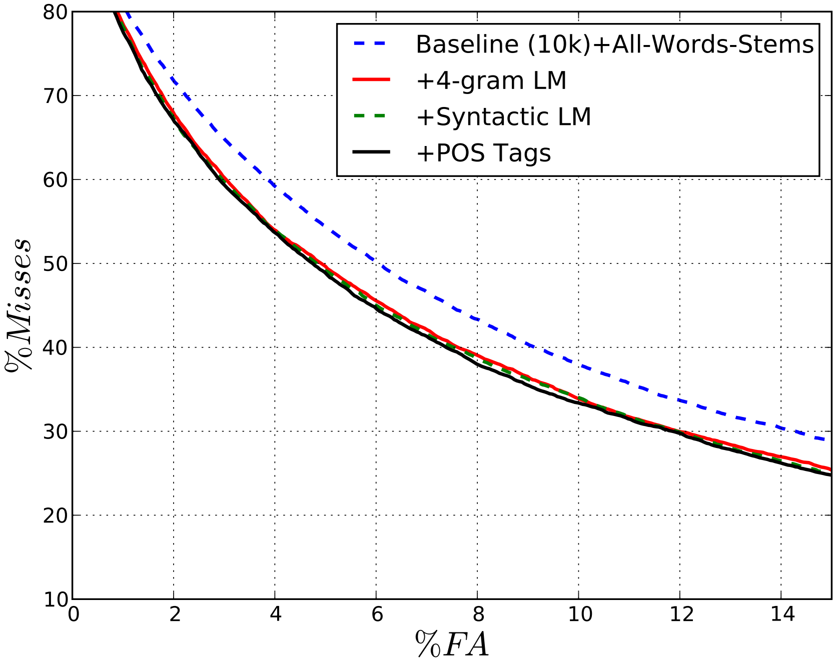
$$\begin{aligned}\text{Likelihood-ratio} &= \log \frac{p(utt)}{p(utt|w_i = \text{unknown})} \\ \text{Norm-LM-score} &= \frac{\log p(utt)}{length(utt)}\end{aligned}$$

where $p(utt)$ represents the probability of the utterance using the best path hypothesis word of the LVCSR system, and $p(utt|w_i = \text{unknown})$ is the probability of the entire utterance with the current word in the LVCSR output replaced by the token `<unk>`, used to represent OOVs. Intuitively, when an OOV word is recognized as an IV word, the fluency of the utterance is disrupted, especially if the IV is a function word. The likelihood-ratio is designed to show whether the utterance is more fluent (more likely) if the current word is a

[7] Available at: http://www.speech.sri.com/projects/srilm/



(a) Baseline (10K) evaluated on All OOVs



(b) Baseline (10K) evaluated on Un-observed OOVs

Figure 4.5: Features from a language model added to the best CRF from Section 4.4 (All-Words-Stemmed).

misrecognized OOV. [8] The second feature (Norm-LM-score) is the normalized likelihood of the utterance. An unlikely utterance biases the system to predicting OOVs.

We evaluated a CRF with these features and all lexical context features (Section 4.4) using both the fourgram model and the joint syntactic language model (Figure 4.5). Most of the gain was obtained using the fourgram language model. Each model improved performance, but the syntactic model provided the largest improvement. At 5% false alarm rate it yields a 4% absolute improvement with respect to the previous best result (All-Words-Stemmed) and 14.2% over the MaxEnt baseline on un-observed OOVs. Higher order language models did not improve.

4.5.3 Additional Syntactic Features

We explored other syntactic features; the most effective was the 5-tag window of POS tags of the best hypothesis.[9] The additive improvement of this feature is depicted in Figure 4.5 labeled `+POS Tags`. With this feature, we achieve a small additional gain.

Other syntactic features we have tried without added benefit include:

- The POS tag for the `<unk>`in the most likely tagging of the best hypothesis with the current word replaced by `<unk>`.

[8] Note that in the standard N-gram LM the feature reduces to $\log \frac{\prod_{k=i}^{i+n-1} p(w_k|w_{k-n+1}^{k-1})}{\prod_{k=i}^{i+n-1} p(w_k|w_{k-n+1}^{k-1}, w_i\text{=unknown})}$, i.e., only n N-grams actually contribute. However, in the syntactic LM, the entire utterance is affected by the change of one word through the latent states (tags) (Eq. 4.1), thus making it a truly global feature.

[9] The POS tags were generated by the same syntactic LM (see Section 4.5.1) as described in Filimonov and Harper (2009b). In this case, POS tags include merged tags, i.e., the vocabulary word *fred's* may be tagged as `NNP-POS` or `NNP-VBZ`.

- $p(t_i = TAG|utt, w_i = \text{unknown})$, where TAG is a tag from a set of up to 14 most frequent tags for unknown words, i.e., the probability of the `<unk>`at the position i to have the tag TAG given the entire utterance[10].

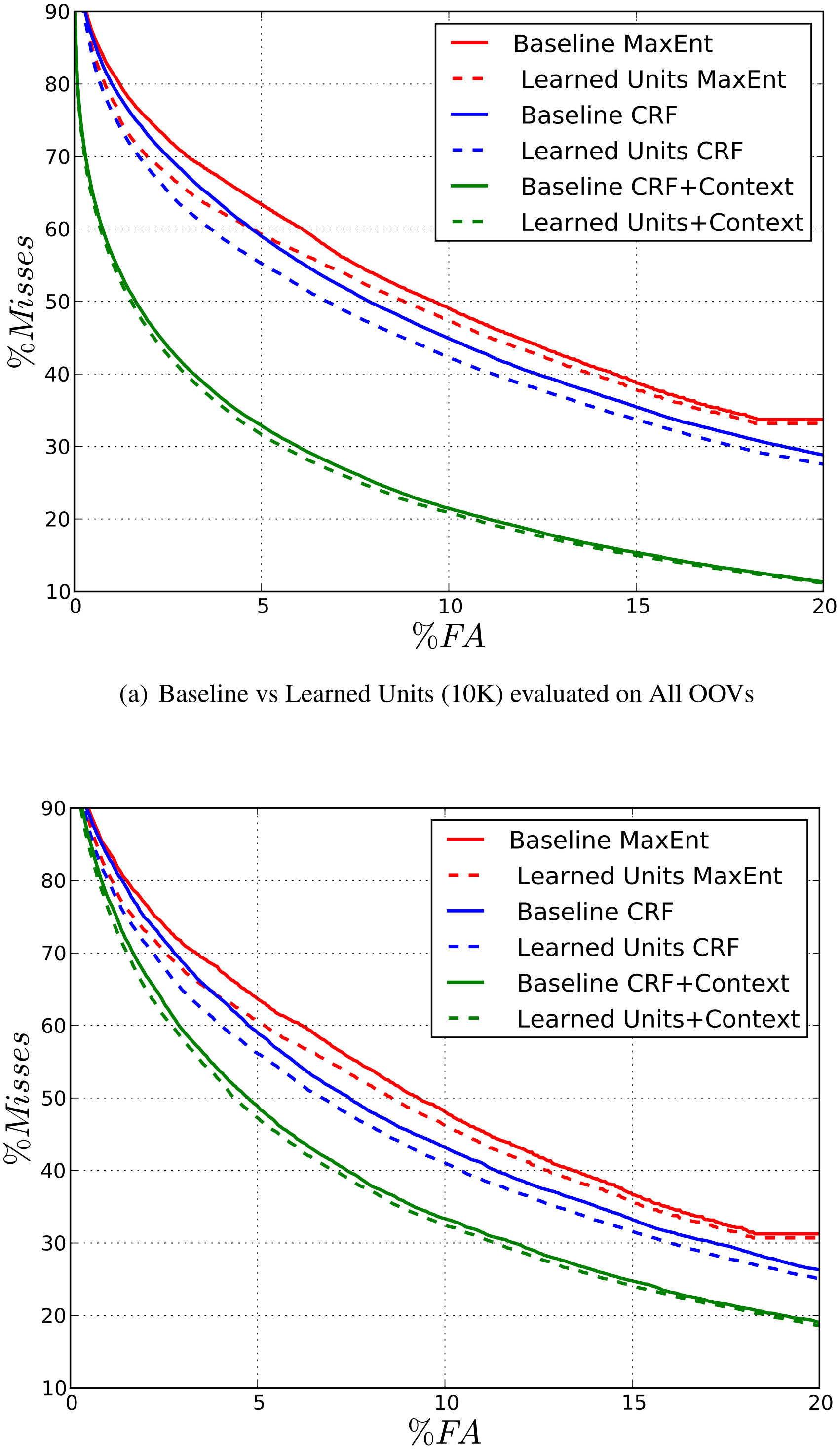
4.6 Final System

Figure 4.6 summarizes all of the context features in a single second order BIO encoded CRF. Results are shown for state-of-the-art MaxEnt (Rastrow et al., 2009a) as well as for the CRF on unobserved, and all OOVs. We include results with baseline sub-words and learned sub-words from Chapter 3. The Learned sub-words with context features (`Learned Units + Context`) still improves over the baseline (`Baseline 10k + Context`), however the relative gain is reduced.

For unobserved OOVs our final system achieves a 14.8% absolute improvement at 5% FA rate by adding context. Including the learned sub-words the total gain is 16.5%. The absolute improvement on All OOVs was 30.5% using context and 31.7% including also new sub-words. The result on all OOVs includes *observed* OOVs: words that are OOV for the LVCSR but are encountered in the OOV detector's training data.

Figure 4.7 shows the OOV detection results in the MIT Lectures data set with different levels of context included. When evaluating on un-observed OOVs in the MIT Lectures

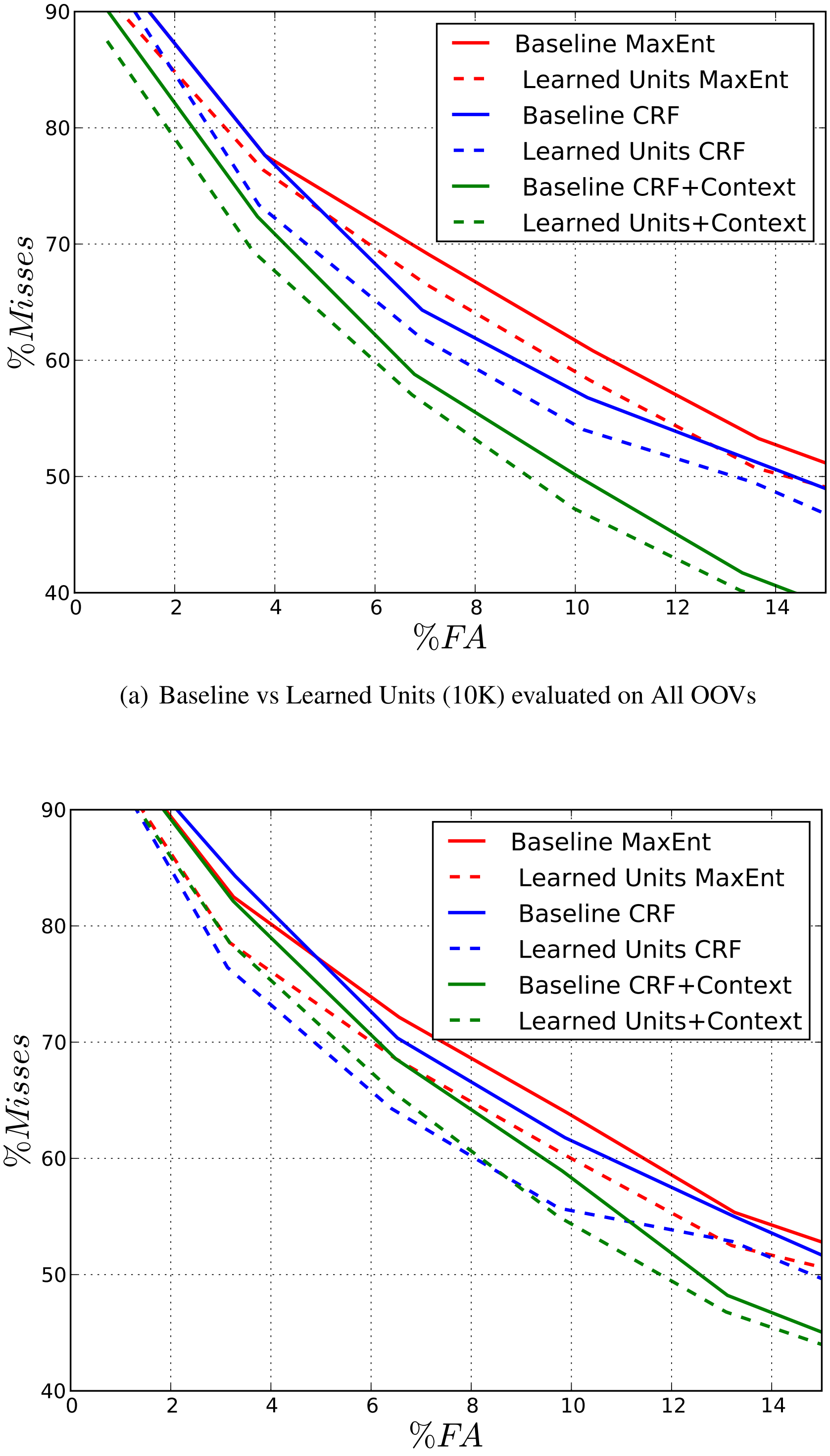
[10] This probability is easy to compute using the syntactic LM: $p(t_i = TAG|w_1^n) = \frac{p(t_i=TAG,w_1^n)}{p(w_1^n)}$. $p(t_i = TAG, w_1^n)$ is estimated similarly to the Eq. 4.1, except we restrict the set of possible values for t_i in the summation term to match TAG.



(a) Baseline vs Learned Units (10K) evaluated on All OOVs

(b) Baseline vs Learned Units (10K) evaluated on Un-observed OOVs

Figure 4.6: Comparing Baseline (Rastrow et al., 2009a) vs Learned Units (Chapter 3) with different context features. Evaluated on `OOVCORP`.



(a) Baseline vs Learned Units (10K) evaluated on All OOVs

(b) Baseline vs Learned Units (10K) evaluated on Un-observed OOVs

Figure 4.7: Comparing Baseline (Rastrow et al., 2009a) vs Learned Units (Chapter 3) with different context features. Evaluated on MIT Lectures corpus.

data set, our final system achieves a 1.4% absolute improvement at 5% FA rate by adding context. Including the learned sub-words the total gain is 7.1%. The absolute improvement on All OOVs was 8.3% using context and 12.3% including also new sub-words.

Similar to Broadcast News, we found that including context achieves improvements in performance. However the gains from context are smaller on the MIT Lectures data set. We conjecture that this is due to the higher WER[11] and the less structured nature of the domain: i.e. ungrammatical sentences, disfluencies, incomplete sentences, making it more difficult to predict OOVs based on context.

The learned sub-words achieve larger gains with respect to the baseline hybrid system in the MIT Lectures data set than in Broadcast News `OOVCORP` data set, specially for unobserved OOVs. These results suggest that the learned sub-words are not simply modeling the training OOVs better than the baseline sub-words, but also describe better novel unexpected words. Recall that these training OOVs were mostly named-entities since it was trained on BN data.

Finally, note that the MaxEnt curve flattens at 18% false alarms, while the CRF continues to decrease. The elbow in the MaxEnt curve corresponds to the probability threshold at which no other labeled OOV region has a non-zero OOV score (regions with zero entropy and no sub-words). In this case, the CRF model can still rely on the context to predict a non-zero OOV score. This helps applications where misses are more heavily penalized than false alarms.

[11] $WER = 32.7\%$ since the LVCSR system was trained on Broadcast News data as described in Section 3.6.

4.7 Related Work

We have shown that combining the presence of sub-word units with other measures of confidence can provided significant improvements, and other proposed local confidence measures could be included in our system as well. Lin et al. (2007) use joint word/phone lattice alignments and classifies high local miss-alignment regions as OOVs. Hazen and Bazzi (2001) combine filler models with word confidence scores, such as the minimum normalized log-likelihood acoustic model score for a word and, the fraction of the N-best utterance hypotheses in which a hypothesized word appears.

Limited contextual information has been previously exploited (although maintaining independence assumptions on the labels). Burget et al. (2008) used a neural-network (NN) phone-posterior estimator as a feature for OOV detection. The network is fed with posterior probabilities from weakly-constrained (phonetic-based) and strongly-constrained (word-based) recognizers. Their system estimates frame-based scores, and interestingly, they report large improvements when using temporal context in the NN input. This context is quite limited; it refers to posterior scores from one frame on each side. Other features are considered and combined using a MaxEnt model. They attribute this gain to sampling from neighboring phonemes. Sun et al. (2001) combines a filler-based model with a confidence approach by using several acoustic features along with context based features, such as whether the next word is a filler, acoustic confidence features for next word, number of fillers, etc.

None of these approaches consider OOV detection as a sequence labeling problem. The work of Liu et al. (2005) is most similar to the approach presented here, but applies a CRF to sentence boundary detection.

4.8 Conclusion

This chapter has presented a novel and effective approach to improve OOV detection in the output confusion networks of a LVCSR system. Local and global contextual information is integrated with sub-word posterior probabilities obtained from a hybrid LVCSR system in a CRF to detect OOV regions effectively. We obtain large improvements from two main sources:

1. We consider the OOV-detection task as a sequence labeling task and use a second-order conditional random fields with BIO encoding for the prediction. CRFs combines the advantages of being a discriminatively trained and being able to model the entire sequence. This approach provides gains despite using the same features as the baseline MaxEnt system.

2. Augmenting the features to include both local lexical context and global information.

These gains were additive when combined with the learned sub-words in Chapter 3.

In the Broadcast News data set, at a 5% FA rate we reduce the missed OOV rate from 63.4% to 31.6%, a 31% absolute error reduction when evaluating all OOVs, and by 16.5% absolute when focusing on un-observed OOVs. Most of the gain was achieved by including contextual information in the OOV detector. On the MIT Lectures data set we achieved 7.1% and 12.2% absolute improvement on un-observed and All OOVs respectively. We found that while context helps, most of the gain in this out-of-domain data-set was achieved by integrating the proposed sub-words proposed in Chapter 3. Context information might have a smaller effect in this data set given that it is a less-structured domain when compared to Broadcast News.

In the following chapters we apply this OOV detector, including proposed sub-words and contextual information, to recover the correct spelling of these novel words for transcription, and to improve robustness of downstream applications to out-of-vocabulary words.

Chapter 5

Recovering Out-of-Vocabulary Words

5.1 Introduction

In previous chapters we focused on identifying when OOVs are spoken and transcribing them using sub-lexical units. For dictation applications this is not sufficient: we need to transcribe the correct spelling of each word. The problem of recovering the correct spelling of a word from its phonetic representation is related to the sound-to-letter problem (Meng et al., 1994; Chen, 2003) where given the correct phonetic sequence we need to obtain the letter sequence. However in this case we need to also account for recognition errors.

In this chapter, our objective is to recover the correct spelling of the identified OOVs in order to correct transcription errors. We assume the audio has been processed by a LVCSR system producing hybrid word/sub-word lattices, such as the one presented in Chapter 3, and that OOVs in the output lattices have been identified using either an oracle or an automatic OOV detector, such as the one presented in Chapter 4.

We propose a novel approach to recover the spelling of OOV terms. For each utterance containing an OOV region, we generate queries for the Google[1] search engine to find words relevant to the utterance topic in the vast and constantly updated World Wide Web. The retrieved documents are processed to extract all words outside of the LVCSR system's vocabulary, forming a candidate-list. This candidate list is used to correct OOV errors in two ways:

- **Spoken Term Detection Framework**: for each candidate OOV retrieved from the Web (query), we use a Spoken Term Detection (STD) system to perform a search over the LVCSR lattices and retrieve all instances (hits) where the query matches the lattice's phonetic sequence.The STD hits replace the best LVCSR hypothesis in each OOV region without the need to re-decode or modify the LVCSR system. In a media monitoring/surveillance/browsing system it is impractical to re-decode and reindex large amounts of data. The proposed solution can be integrated naturally with the speech retrieval architecture with minimal cost.

- **Vocabulary Expansion**: expand the vocabulary of the LVCSR system to include the candidate-list obtained from the Web, and re-decode.

Our approach for identifying relevant OOV terms from the Web is described in Section 5.2. Section 5.3 describes how the STD system is used for OOV recovery, and Sec-

[1] www.google.com

tion 5.4 describes the proposed method for vocabulary expansion. Section 5.5 describes the results analysis and we conclude with a summary of the key findings of this chapter.

5.2 Identifying Relevant OOVs on the Web

Our OOV recovery approach exploits the lexical context of OOV regions to query the Web and retrieve content relevant to the utterance. For each utterance containing an OOV region, we select M relevant words in the utterance and submit the set of words as a single query to a search engine (Google). To identify keywords, we rank the decoded words for that utterance using TF-IDF. This is a common information retrieval score used to evaluate how important word t_i is to document d_j in a collection of documents D, as given in the expression below:

$$\text{TF-IDF}_{i,j} = \frac{n_{i,j}}{\sum_k n_{k,j}} \log \frac{|D|}{|d : t_i \in d|} \tag{5.1}$$

where $n_{i,j}$ is the number of occurrences of term t_i in document (utterance) d_j, $|d : t_i \in d|$ is the number of documents (utterances) where the term t_i appears, and $|D|$ is the number of documents (utterances). We considered only words with confidence[2] larger than a threshold $T = 0.8$, and not present in a standard stop-list. We selected the top M keywords per utterance ($M = 5$) as a query to the search engine and retrieved the top 20 documents. Each retrieved document was scanned to create a list of “potential OOVs" containing all OOV words within a fixed window around the query terms. Some sample queries which we

[2] We used as confidence the posterior probability associated with each word in the output confusion networks.

will refer to in our results (Section 5.5) include "MEDIATORS UNDERMINE EXTREME NATIONALIST BOSNIAN" which helped us to recover the OOV word *Milosevic* and the query "POLLS REFERENDUM SHARON ISRAELIS WEDNESDAY" which helped recover *Netanyahu*.

Table 5.1 shows for various window sizes around the query terms the tradeoff between the size of the retrieved word set versus recall. In our experiments we used a window of 10 words. Note that in this work we consider a global list of potential OOVs rather than an utterance specific list. In our experiments we found that limiting the set of potential OOVs to the words retrieved using the current utterance had a significantly lower recall (30%) than creating a global list. This could be due to the fact that many utterances refer to the same topic, while their automatic transcription performance might vary significantly, yielding noisy or irrelevant keywords.

To reduce the size of the candidate list we remove bad candidates, such as misspelled words, typos, and invented words, by filtering words which appear less than K times in the retrieved documents. Figure 5.1 shows the recall of true OOVs versus candidate list size for the OOVCORP data set. We selected $K = 4$ for our experiments based on development set tuning. This setting reduces the list to 22K while still retrieving 53% of the missing OOVs. We tried several other approaches to further reduce the candidate list, including: scoring candidates with a character letter N-gram model trained on In-Vocabulary (IV) words and excluding concatenated IVs (commonly found on the Web). There was no significant improvement over the simple strategy of shortlisting by Web counts.

Window Size	Recall	List Size
1	7.86	36K
3	22.78	105K
5	38.15	170K
10	71.24	323K
Paragraph	91.55	1.4M

Table 5.1: Recall versus candidate list set size.

Acquiring Web data using automatically generated queries similar to the approach described in this section has been of considerable interest as a means of increasing the amount of language model training data and incorporating new words into the LVCSR vocabulary (Ng et al., 2005; Creutz et al., 2009; Oger et al., 2009). In this approach we focus on using the OOV word list acquired from the Web as a source of query terms for a spoken term detection system, which can then help to identify and incorporate the OOV terms in LVCSR output without re-decoding or modifying the LVCSR system. We also evaluate an approach for augmenting the LVCSR system's vocabulary to re-decode.

5.3 Spoken Term Detection for OOV Recovery

In the previous section, we described a method for obtaining potential spelling forms for uttered OOVs using contextual queries and the Web. With this candidate list a spoken term detection system is used to identify matches in the LVCSR output.

We use a weighted finite state transducer (WFST) based STD system described in detail in Chapter 6, Section 6.2 (Can et al., 2009; Allauzen, Mohri, and Murat, 2004). To build

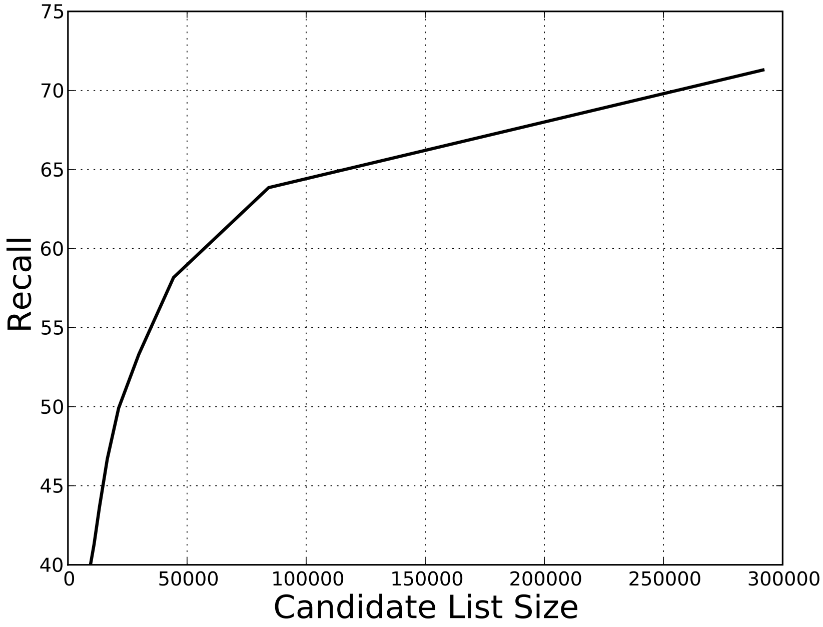


Figure 5.1: Unique OOV recall versus the size of the candidate list for different term frequency thresholds (window of 10).

the index, we process the audio with a hybrid LVCSR system (Chapter 3) to obtain the corresponding phonetic lattices. The resulting phonetic index is used in all of our experiments. At search time each textual query is converted to its phonetic representation using the pronunciations obtained from a letter-to-sound (L2S) system (Chen, 2003).

5.3.1 Decision Thresholds for OOV Recovery

The performance of our OOV recovery approach depends on what candidate hits are accepted from the output of the STD system. Miller, Kleber, lin Kao, and Kimball (2007) presented an optimal decision theoretic approach for selecting a term specific threshold (TST) on STD scores for deciding whether a postulated hit for a term should be included in the search output. The threshold was optimized for the NIST ATWV metric and has been widely adopted in the research community. Here we present an extension of TST targeted towards improving WER by OOV recovery.

Motivated by Miller et al. (2007) we aim to select a threshold that leads to a reduction of WER. Given a hit for term t in region r with confidence score $P(t,r)$[3], we accept a hit if the benefit of correctly retrieving it (B) is larger than the cost of incorrectly retrieving it (C), as shown in Equation 5.2.

$$P(t,r)B - [1 - P(t,r)]C > 0 \tag{5.2}$$

Let T_{OOV} be all missing OOVs, R_{OOV} the set of all OOV regions, then in terms of WER, the benefit of correctly retrieving a missing OOV is $B = 1$ if $r \in R_{OOV}$ and $t \in T_{OOV}$, and cost $C = \beta$ if $r \notin R_{OOV}$, where β the cost of an error. We rewrite:

$$P(t,r)P(r \in R_{OOV})P(t \in T_{OOV}) - (1 - P(t,r))\beta P(r \notin R_{OOV}) > 0 \tag{5.3}$$

Solving for $P(t,r)$ yields:

$$P(t,r) > \frac{1 - P(r \in R_{OOV})}{1 - P(r \in R_{OOV}) + \frac{P(t \in T_{OOV})P(r \in R_{OOV})}{\beta}} \tag{5.4}$$

[3] $P(t,r)$ is the probability of the acoustic match obtained from the STD system.

where $P(r \in R_{OOV})$ is the probability that region r is an OOV region and $P(t \in T_{OOV})$ is the probability that the term t is a missing OOV. Note that if the region r is OOV with probability 1, then we should accept any hit with $P(t, r) > 0$ (all), since no additional errors will be incurred. However if the region is *not* an OOV region, i.e., $P(r \in R_{OOV}) = 0$, then we can only accept hits with probability $P(t, r) > 1$ (none), since an error will always be incurred. As the cost of incurring an error β increases, the threshold tends to 1.

5.3.2 Incorporating STD Matches in LVCSR Output

The matching regions in the LVCSR output are replaced with the OOV word. An alternative is to re-decode the utterance with an augmented vocabulary. Replacing the matched regions is faster as no re-decoding is necessary. For our OOVCORP test-set, incorporating the words from the web to correct errors using the STD framework takes roughly 15 minutes, while re-decoding the 90 hours test-set takes several hours (around 10 hours). Assuming we only replace words in OOV regions, this approach does not tradeoff accuracy of in-vocabulary words for rare words.

5.4 On Demand Vocabulary Expansion

An alternative approach to correct Out-of-Vocabulary errors for transcription is to augment the system's vocabulary (dictionary) and language model to include words relevant to the OOV utterances. We include the recovered words from the web in the dictionary of the LVCSR system using their most likely pronunciations (top 6) obtained from a letter-to-sound model (Stanley F. Chen, 2003). Since OOV words obtained from the Web are not present at training time, we don't have a language model probability for them.[4]

In this work, we incorporate the new words in the language model as unigrams sharing the probability of the <UNK> token. This token typically represents all words in the language modeling training text which were not included in the vocabulary of the system (typically because they had low frequency counts in this text). Specifically, we re-distribute the unigram probability of the <UNK> token among all words from the candidate list and a new <UNK> token.[5] Note that re-distributing the <UNK> unigram probability does not affect the probability mass assigned to seen words during training, hence reducing the risk of trading off accuracy of IV vs OOV words. New words will only be favored by the language model if it needs to back-off to unigram. Once we augment the dictionary and language model, the complete test-set audio is re-decoded.

Oger et al. (2009) presents an approach for on-demand vocabulary expansion similar to the one we proposed here. However they assume OOV regions have been manually identified, and re-decode each segment using a segment specific augmented vocabulary. Our approach employs automatic detection to find OOV utterances, and builds a global OOV candidate list from the web for the entire corpus. Building a global list achieved

[4] We could extract entire documents from the web and adapt/augment our existing language models with the extracted web documents to obtain a LM probability for these words. This is a natural extension to the approach proposed here.

[5] We update the backoff weights to ensure we keep a proper distribution. Specifically, we modify the ARPALM file directly and use SRILM's -renorm flag to obtain a re-normalized LM.

higher recall since OOVs tend to appear in bursts providing multiple contexts from different utterances.

Finally, our approach to integrate new words in the language model differ from that of Oger et al. (2009). The authors assigned a language model probability to each OOV from the web according to its POS tag. They also tried re-decoding including new words as pronunciation variants of the <UNK> word, hence assigning the same <UNK> probability to all words. Instead, we re-distribute the <UNK> unigram probability and do not modify higher order ngrams. Each word in our dictionary has its most likely 6 pronunciations according to the L2S model, they are not pronunciation variants of the <UNK> token. Oger et al. (2009) recovered 7.7% of the missing OOVs in a 6 hour test-set of French Broadcast News, assuming the OOV regions had been manually identified, achieving 0.2% WER improvement only when using the POS tag information in the language model.

5.5 Results

As in previous chapters we evaluate our approach on the OOVCORP data set. The 100 hours from English Broadcast News are divided into 5 hours of training for the OOV detector, 5 hours of development set, and 90 hours of evaluating recovery.

We first consider the case where OOV segments are identified using an oracle. We identify OOV regions by finding time segments in the manual transcripts containing words which are not in the LVCSR system vocabulary. This allows for the direct evaluation of

System	Precision (%)	Recall OOVs (%)	WER (%)
Baseline	n/a	0	15.8
STD recovery	82.55	33.45	14.9

Table 5.2: Recovery results using **oracle** OOV detection. Recall OOVs is the number of unique OOVs found.

the effectiveness of our retrieval strategy as well as the STD framework for integrating the words into the automatic transcripts. We also consider OOV regions that are detected automatically, which represents a more realistic scenario.

5.5.1 STD Recovery with Oracle OOV Detection

We use the STD system presented in Section 5.3 to phonetically match each retrieved word to the corresponding OOV regions in the decoded audio. We built a phonetic index from the lattices obtained from the LVCSR system, and consider the list of 22K words as OOV queries to the STD system. At search time the textual queries are converted to their phonetic representation using the pronunciations obtained from the letter to sound (L2S) system (Chen, 2003). Note that this query list contains 53% of the missing OOVs as described in Section 5.2.

Table 5.2 summarizes our results for searches with the top 6 weighted pronunciations for each query. Assuming OOV regions have been manually identified (oracle), we are able to recover 33.5% of the unique OOVs in the test-set, achieving a 0.9% absolute improvement in WER. The remaining OOVs could not be assigned a recovered term due to mismatches between the hypothesized pronunciation and the phonetic string in the index.

Original Decode		**After Recovery**
the netting yahoo government negotiated	⇒	the NETANYAHU government negotiated
former president slogan I'm a loss of itch	⇒	former president slogan MILOSEVIC

Figure 5.2: Example utterances before and after recovery (automatic OOV detection). Incorrect terms are underlined in the decoded string and corrections are emphasized. The system corrects the OOV in the first string and one of the two OOVs in the second (Slobodan remains incorrect as "slogan"), improving understandability significantly. See Section 5.2 for the Web queries that retrieved these OOVs.

Further improvement could be achieved by modeling phonetic confusability as described in Chapter 6.

5.5.2 STD Recovery with Automatic OOV Detection

We repeated the above experiments using the automatic OOV detection system presented in Chapter 4. The OOV detector's performance on our 90 hour test set is repeated in Figure 5.4. We incorporate OOV detection as a post-processing step to the STD system, by penalizing mismatches between query type (OOV) and false alarms returned from in-vocabulary (IV) regions. The mismatch penalty is tuned on the development set. More details on the post-processing step are presented in Chapter 6.

Table 5.3 shows the results in terms of recall and WER when using automatic OOV detection for different thresholds. Using the standard term specific threshold (TST) (Miller et al., 2007) resulted in degraded WER performance. The TST however, achieves the highest recall (27.71%). The degraded WER and high recall can be explained by the fact that the TST is designed to maximize NIST ATWV metric, which assumes that every query appears in the index at least once. This guarantees that at least one hit is retrieved for every query processed by the STD system. For OOV recovery, the query terms include noise (invalid words) from the web, hence accepting at least one hit per term causes a large number of false alarms and higher recall for true OOVs.

The proposed term-region specific threshold (TRST) reduces this effect, retrieving 12.7% of OOV regions with a small gain in WER. This result is expected since the threshold only allows hits in predicted OOV regions, yielding fewer false alarms. In this work we consider $P(t \in T_{OOV}) = 1$ for Equation 5.4. Including prior knowledge about the a candidate term should improve performance further.

The best recall/WER tradeoff is obtained using the term-specific threshold combined with a hard-threshold (TST + HT), which retrieves 17.9% of the missing OOVs respectively, achieving an improvement of 0.2% in WER, which is statistically significant ($p < 0.001$).[6] Finally, we combined both thresholds (TRST + TST). If the score is higher than a hard-threshold HT, we used the TST. This did not provide further improvements.

From a random sample of 100 OOV words correctly recovered, 92% were named entities. From this set 68% were people, and 24% were locations, organization, or other.[7] As expected, the retrieved words are high information bearing words that improve understanding of the transcription significantly. Figure 5.2 depicts two example utterances from our test set before and after recovery using automatic OOV detection.

[6] For statistical significance, we used the *mapsswe* test.

[7] We manually labeled the retrieved words in the true transcription.

STD Threshold	Precision (%)	Recall OOVs (%)	WER (%)
Baseline	-	-	15.8
TST	11.23	27.71	17.6
TST+HT	29.53	17.98	**15.6**
TRST	21.95	12.71	15.7
TRST+HT	27.12	15.82	**15.6**

Table 5.3: Recovery results using an automatic OOV detector.

5.5.3 Recovery by re-decoding

We also evaluated re-decoding the test-set audio after expanding the vocabulary with the 22K words obtained from the Web. We modified the language model to include these words as detailed in Section 5.4. Table 5.4 summarizes the results. This approach is able to reduce WER by 0.7% absolute, and correctly decode 21.55% of the OOVs in the test-set. We can see that the large gains in WER are explained by the improved precision from the re-decoding approach compared to STD-recovery framework.

We conjecture the improved precision is explained by the fact that, when re-decoding, new words are only predicted by the language model when it needs to back-off to unigrams, i.e. when it realizes the word was never seen during training. In contrast, the STD approach always replaces the novel word if the phonetic string in the recognition output matches the new word's pronunciation, generating more false alarms. Figure 5.3 depicts examples recovered using this approach.

Finally, we evaluated the potential improvement of having language model training text containing the OOVs obtained from the Web. This achieves 2% absolute improvement (RE-DECODE - ORACLE LM), motivating future work on language model adaptation using

System	Precision (%)	Recall OOVs (%)	WER (%)
Baseline	-	0	15.8
re-decode - UNK prob	70.78	21.55	**15.1**
re-decode - **Oracle** LM	74.20	41.50	13.8
re-decode - **Oracle** LM and Vocab	99.6	54.0	13.2

Table 5.4: Recovery results using STD-framework vs re-decoding. Recall OOVs is the number of unique OOVs correctly decoded.

Original Decode		**After Recovery**
celebrates sold on the loss of interest	⇒	yugoslavia celebrates SLOBODAN MILOSEVIC
presidency of where to put your morning	⇒	presidency of ALBERTO FUJIMORI
leader of the has boa guerrillas	⇒	leader of the HEZBOLLAH guerrillas
unibomber suspect theodore can pinsky	⇒	unibomber suspect theodore KACZYNKSKY
the netting yahoo government negotiated	⇒	the NETANYAHU government negotiated
pan am locker the disaster	⇒	pan am LOCKERBIE disaster
this drug called the lender in a cost	⇒	this drug called ALENDRONATE cost
olympic champion get a cappella cough	⇒	olympic champion YEVGENI KAFELNIKOV

Figure 5.3: Example utterances before and after recovery (re-decoding approach). Incorrect terms are underlined in the decoded string and corrections are emphasized.

the Web documents. We also built the full system (RE-DECODE - ORACLE LM AND VOCAB): closed test-set vocabulary and language model, i.e. if the OOV rate was 0. In this case, the performance is 13.2%, a 2.6% absolute improvement, there is clearly room for improvement.

5.6 Conclusion

In this chapter we presented a novel approach to recover Out-Of-Vocabulary words using the Web as a corpus. Our method incorporates the retrieved words using a Spoken Term Detection system or by re-decoding the audio. The former approach is faster since it

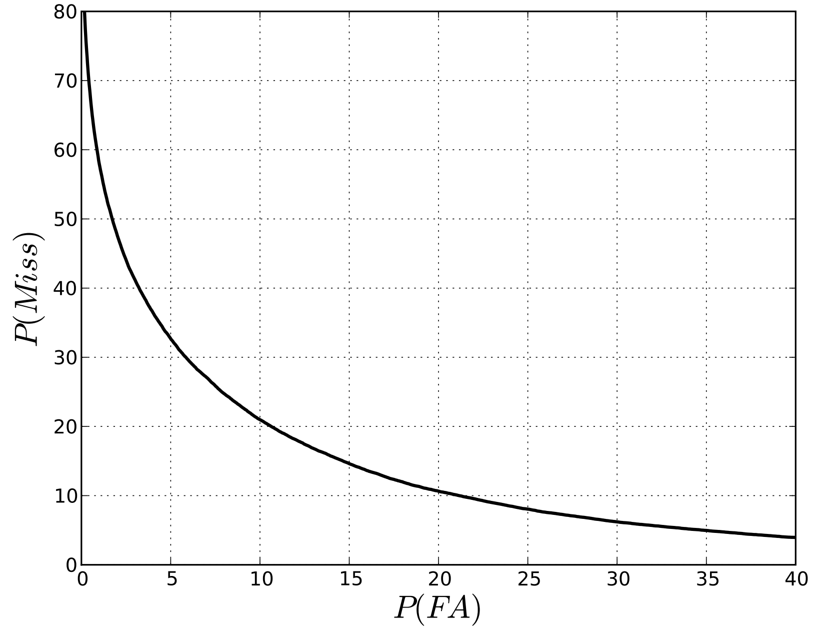


Figure 5.4: Performance of the final OOV detector from Chapter 4 on the `OOVCORP`.

does not require re-decoding the audio. We evaluate our approach when OOVs are automatically identified, the more realistic situation, recovering 17.98% of the missing OOVs and improving WER by 0.2% using the STD-framework approach. Re-decoding the audio achieves a recall of 21.55%, improving WER by 0.7% absolute. Furthermore, the recovered words are 92% named entities, which improves the understanding of the transcription.

Chapter 6

Spoken Term Detection with OOV Queries

6.1 Introduction

The fast growing availability of recorded speech calls for efficient and scalable solutions to index and search this data. Spoken term detection (STD) is a key technology aimed at open vocabulary search over large collections of speech content. Out-of-Vocabulary words pose an important problem in this task, since queries typically relate to information rich nouns, such as named-entities and foreign words, which have poor coverage in the vocabulary. This has been empirically observed in live audio indexing systems for the web, such as (Logan et al., 1996), where OOVs were found to represent about 15% of search queries.

The most common approach to STD is the use of a large vocabulary continuous speech recognition (LVCSR) system to obtain word lattices that are subsequently indexed (J. Mamou and Hoory, 2006). Critically, the search of queries containing OOV terms in the LVCSR processed output will not return any results[1]. A typical solution to OOV queries, is to build a phonetic index. We can then search for the phonetic representation of the OOV term (i.e. its pronunciation) in this phonetic index (Mamou et al., 2007; Can et al., 2009). However, there are many challenges in finding a good operation point for a spoken term detection system that balances false alarms and true hits, particularly when the queries are Out of Vocabulary terms.

In this chapter, we demonstrate the usefulness of the proposed hybrid recognizer and OOV detector (Chapter 3 and 4) to enhance robustness to OOVs in the spoken term detection (STD) task. We show that using a hybrid recognizer to derive the phonetic index increases recall while automatically tagging OOV regions helps reduce false alarms. We also propose incorporating phonetic confusability to further increase recall for Out-of-Vocabulary queries and explore additional features, such as: boosting the probability of a hit in accordance with the number of neighboring hits and query-length normalization.

[1] Since the index is built from the LVCSR lattices, which do not contain any words outside of the vocabulary.

6.2 A Weighted Finite State (WFST) Transducer based Spoken Term Detection System

In this section we describe the overall framework of our STD system and the methods used in our experiments. We assume that the audio to be indexed has been processed with an LVCSR system and the corresponding word or sub-word lattices are available. Phonetic lattices are subsequently derived by mapping all words in the lattice to their phonetic representation using the dictionary. The phonetic lattices are used to build the indexes used in all of our experiments. In this chapter we make extensive use of Weighted Finite State Transducers (WFST). An overview of WFSTs including formal definition, notation, and common operations is presented in Appendix A.

6.2.1 Pre-Processing

Prior to creating the index, the phonetic lattices are preprocessed into weighted finite state transducers (WFST). The word/sub-word is stored in the input label of each arc in the lattice, and the output label is used to store the timing information. An additional normalization step (achieved by weight pushing in the log-semiring) (Mohri, Pereira, and

Riley, 1996) converts the weights into the desired posterior probabilities. All silences and hesitations in the lattices are converted to ϵ arcs.[2]

In essence, each arc in the resulting WFST representing the lattice is a 5-tuple (p, i, o, w, q) where $p \in Q$ is the source state, $q \in Q$ is the destination state, $i \in \Sigma$ is the input label (phone), $o \in \Re$ is the output label (start-time associated with state p), and $w \in \Re^+$ is (neg log of) posterior probability associated with i. Q is a finite set of states and Σ is the input alphabet (words or sub-words).

6.2.2 WFST-based Indexing and Retrieval

We use the algorithm described in (Allauzen et al., 2004) to create a full index of the pre-processed lattices represented as a WFST. The final index maps each substring x (word or sub-word) to the set of indexes in the automata (lattices) in which x appears. Here, the weight of each path gives the within utterance expected counts of the substring corresponding to that path. This algorithm is optimal for search: the search is linear in the size of the query string and the number of indexes of the weighted automata in which it appears (Allauzen et al., 2004).

At search time the textual queries are converted to their phonetic representation using the pronunciations obtained from the L2S system described in (Chen, 2003). This phonetic sequence is represented as a weighted acceptor and using a single composition operation

[2] By convention, ϵ is a special symbol which consumes no symbol and matches all symbols in the input and output alphabets. FST operations effectively ignore ϵ arcs. For example , if the lattice contains the path "`president %HESITATION SIL fujimori`", converting the `%HESITATION` and `SIL` symbols to ϵ allows the search algorithm to match "`president fujimori`" in this region.

(Mohri et al., 1996) with the index we can retrieve the automata (lattice) containing it. Note the flexibility of this framework, since it allows us to search for any weighted finite state acceptor as query, an advantage we exploit.

The construction described above only retrieves the lattice indexes. However it can be used as the first pass in a two-pass STD retrieval system as described in (Parlak and Saraclar, 2008). Essentially, once the lattice indexes have been identified (in the first pass), the second pass loads the relevant lattices and extract the time marks corresponding to the query. Alternatively, the index can be modified to perform 1-pass retrieval (Can et al., 2009), improving search times at the cost of a larger index and with comparable performance. We implemented a 2-pass FST-based indexing system using the OpenFst toolkit (Allauzen, Riley, Schalkwyk, Skut, and Mohri, 2007). We derive the phonetic index from word lattices and achieve comparable performance to that reported by (Can et al., 2009) as shown in Table 6.1. We also explore deriving the phonetic index from the baseline hybrid (Rastrow et al., 2009a) and proposed hybrid (Chapter 3) systems.

6.2.3 Evaluation

To evaluate performance of the spoken term detection system, we present results in terms of the NIST 2006 STD Evaluation criteria (NIST): “Actual Term-Weighted Value” defined below.

$$ATWV = 1 - \frac{1}{|\mathcal{Q}|}\sum_{q\in\mathcal{Q}} P_{miss}(q) + \beta P_{FA}(q) \tag{6.1}$$

with $P_{miss}(q)$ and $P_{FA}(q)$ defined as in Equation 6.2 and 6.3:

$$P_{miss}(q) = 1 - \frac{N_{correct}(q)}{N_{true}(q)} \tag{6.2}$$

$$P_{FA}(q) = \frac{N_{spurious}(q)}{T - N_{true}(q)} \tag{6.3}$$

where:

- $\mathcal{Q} =$ query set
- $N_{true} =$ occurrences in reference
- $N_{spurious} =$ spurious instances retrieved
- $N_{correct} =$ correctly retrieved instances
- $\beta =$ user defined parameter, in STD 06 Eval, $\beta = 999.99$
- $T =$ total number of seconds in audio

The ATWV defined in Eq. (6.1) requires a binary decision for each instance returned by the system. In this work, we use the *Term Specific Threshold* introduced by (Miller et al., 2007) to make a binary decision, since it achieved higher performance in our experiments than a global threshold. The term specific threshold asserts only those candidate hits for query q if the benefit $B(q)$ of correctly accepting them is larger than the cost $C(q)$ of incorrectly retrieving them. That is, we only accept an instance if its probability p ensures:

$$pB(q) - (1 - p)C(q) > 0 \tag{6.4}$$

According to the ATWV metric given by Eq. (6.1) the benefit is $B(q) = 1/N_{true}(q)$ and the cost is $C(q) = \beta/(T - N_{true}(q))$. This yields the following term-specific threshold:

$$p(q) > \frac{N_{true}(q)}{T/\beta + \frac{\beta-1}{\beta} N_{true}(q)} \tag{6.5}$$

Since N_{true} is unknown in the test set, this quantity is approximated by the sum of the posterior probabilities of all candidate detections for that term q in the lattices.

6.3 Experimental Setup

For our experiments we indexed the 100 hours data set OOVCORP described in Section 2.2.2. The query set was the 1290 OOVs in the corpus. Since all queries are out-of-vocabulary words, we build a phonetic index instead of a word index. We then search for the phonetic representation of the query in the phonetic index. In all our experiments we use the letter-to-sound (L2S) model introduced by Stanley F. Chen (2003).

We compare the performance of the word-based system, the baseline hybrid system (Section 3.5.1), and the proposed hybrid system presented in Chapter 3. Since we explore using OOV detection to reduce false alarms and other algorithmic improvements specific to STD, we divided the 100 hours from `OOVCORP` into 5 hours to train the OOV detector (as in Chapter 3 and 4), another 5 hours for development set, and the remaining 90 hours for test set.

Finally, we also presents results on the NIST 2006 Spoken Term Detection Dev06 test set which we will refer to as **DEV06**. This set comprises of around 3 hours of speech and has 1107 query terms. The OOV rate on this query set is low, with only 16 terms being OOV.

6.4 Word versus Hybrid LVCSR for STD

In order to establish a comparative baseline for our STD results, we first present experiments using a phonetic index derived from word lattices. First, the textual queries are converted to their phonetic representation using the reference pronunciations of the OOV queries, which we refer to as *reflex*. Table 6.1 presents the *reflex* results, which achieve comparable performance to those presented by (Can et al., 2009) (they obtain ATWV $= 0.322$) on the same data set when a word-based lattice index is used. Higher values in ATWV are better.

Data	P(FA)	P(miss)	ATWV
Word 1best	0.00002	0.757	0.227
Word lattices	0.00004	0.638	0.325

Table 6.1: Reflex results on OOVCORP data set.

In practice we don't have the true pronunciation for OOVs, hence we also show results using the pronunciations obtained from the L2S model. For the L2S system, performance peaks at 6 pronunciations, when weighted with the normalized L2S scores. Hence we use 6 weighted pronunciations for all remaining results. Table 6.2 presents results using L2S pronunciations for the word system, the baseline hybrid and proposed hybrid systems. We found more useful to examine total number of hits and total number of false alarms when evaluating different systems, so we report total counts instead of $P(miss)$ and $P(FA)$.

Data	Hits	FAs	ATWV
Word Lattices	6349	9464	0.307
Hybrid (baseline) Lattices	7254	10275	0.352
Hybrid (proposed) Lattices	7247	10316	0.351

Table 6.2: N-Best L2S pronunciations on 90 hours OOVCORP data set.

As can be seen in Table 6.2, the hybrid systems produce better STD performance by increasing the number of retrieved instances. We conjecture that this is due to the lower phone error rates observed in OOV regions when using a hybrid recognizer. The baseline hybrid (Section 3.5.2) and proposed hybrid systems from Chapter 3 achieve comparable performance and they both improve over the phonetic index derived from word lattices.

There is a lot of room for improvement. The total number of instances in the reference is 20,172 but we are only retrieving around 7,000. Furthermore, the number of false alarms is larger than the number of hits. This is often the case for OOV queries since we are searching a phonetic index and queries can match across-words or within word phonetic sequences. In the following section we propose further algorithmic improvements for increase the number of hits, and reduce the number of false alarms.

6.5 Reducing False Alarms

6.5.1 OOV-Detection

In this section we explore the benefits of incorporating automatic detection of OOV regions in the indexed audio. To identify these regions we employed the baseline OOV detector (Section 3.5.1) and the improved OOV detector described in Chapter 4. Figure 7.1 depicts the final performance of these OOV detection systems.

We incorporate the OOV-detection as a post-processing step to the STD system described in Section 6.2. Let the score of an occurrence of a query-term q at time interval Δ_t (returned by the system described in Section 6.2) be denoted by $score_q(\Delta_t)$. The updated score is given by the Equation below.

$$score_q(\Delta_t, \gamma_o) = \begin{cases} score_q(\Delta_t) & OOV_{scr}(\Delta_t) > \gamma_1 \\ score_q(\Delta_t) \times \gamma_o & o/w \end{cases}$$

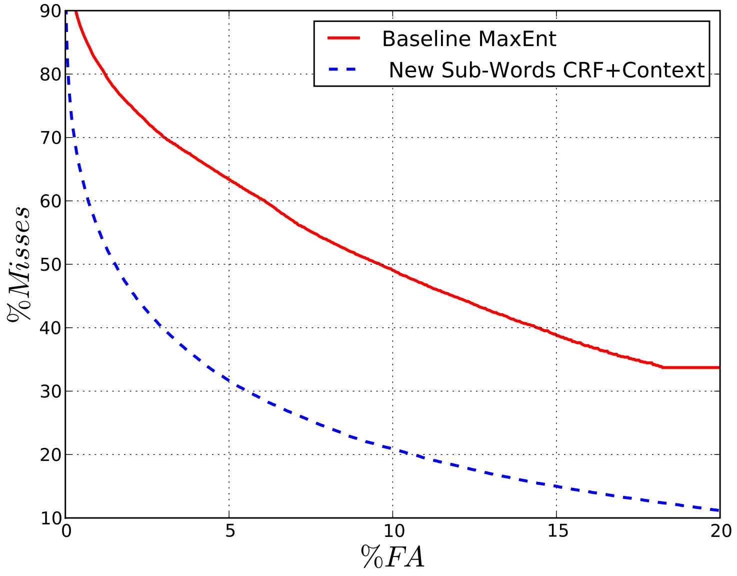


Figure 6.1: DET curve for Baseline OOV detector (`MaxEnt`) and improved OOV detector (`Context`) proposed in Chapter 4

where γ_o is a parameter tuned on the development set, which penalizes the mismatch between query-type (OOV) and false alarms returned from In-Vocabulary (IV) regions. γ_1 selects the operating point for the OOV-detector on the development set.

6.5.2 Query Length Normalization

In order to further reduce False Alarms, we incorporated a normalization penalty based on the length of the query term. Since hits with a longer duration are less likely to be false alarms, we adjust the score in a post-processing step as shown in Equation 6.6 below:

$$score_q(\Delta_t, \gamma_L) = score_q(\Delta_t)^{\frac{\gamma_L}{\Delta_{avg}(q)}} \tag{6.6}$$

where $\Delta_{avg}(q)$ is the average duration of all returned hits for the query-term q, and $\gamma_L \in [0, 1]$ is a parameter tuned on the development set. This approach is similar to the query-length normalization presented by (Mamou et al., 2007), where the authors normalize the scores by a fixed factor $\gamma_L = 1/n$, where n is the number of words in the query phrase.

6.6 Increasing Hits

6.6.1 Incorporating Phonetic Confusions

In this section we address the low recalls for OOV queries. The main cause for low recalls is the mismatch between the phonetic string used to represent the query (obtained from an L2S model) and the phonetic string present in the lattice when that word was spoken. Table 6.3 illustrates a typical OOV query from our test set (`GAVLAK`) which was completely missed by the baseline STD system due to mismatch in some phones between the pronunciation used for the query and phonetic sequence present in the decoded lattice that was indexed. As can be seen in this example, neither the reference pronunciation nor the top 6 pronunciations from the L2S model match any of the phone representations found in the lattice (i.e. `G AE V L [EH or AE] K`).

We propose to compensate for potential differences in deriving index and query representations by augmenting the query representation with phone confusions. Including phone confusability allows for phone substitutions, deletions, and insertions which helps reduce these misses. In this example, allowing substitution of `AA` by `EH` or `AE` in fifth position will be sufficient.

query	GAVLAK					
reference pron	G	AE	V	L	AA	K
L2S 6 best prons	G Y	AE AA AX EY	V	L	AA AY	K
index	**Candidate Hits**					
word decode	GET			LIKE		
hybrid decode	G_AE_V			L_EH_K		
phonetic lattice from hyb	G	AE	V	L	EH AE	K

Table 6.3: Example of various phonetic representations for an OOV query and potential hits in the phonetic indices

In this work we use a phonetic confusion transducer (P2P) generated from a phonetic confusion matrix based on a neural network based acoustic model (Ramabhadran, Sethy, Mamou, Kingsbury, and Chaudhari, 2009). The phonetic confusion matrix was derived from broadcast news development data. Similar results were obtained with an alternate approach to compute a confusion matrix using string alignments between reference and decoded phonetic representations. To augment the query with phonetic confusions we compose the WFSA representation of the query with the P2P transducer and generate n-bests.

More formally, for any string s, let $I(s)$ be the transducer that maps each character is s to itself, let L2S be the transducer that maps any letter string to a set of possible pronunciations, and P2P be the transducer that maps phones to phones, where the weights are obtained from the P2P system explained above. The new query representation is then given by:

$$qfst = bestpathN(I(q) \circ L2S \circ P2P) \tag{6.7}$$

where q is the textual query, $\circ$ and $bestpathN$ correspond to the standard composition and shortest-path operations (in *Tropical* semiring) for WFSTs.

6.6.2 Cache feature

Certain content words, especially rare words, tend to appear in bursts. Inspired by the Cache Language Model (Jelinek, Merialdo, Roukos, and Strauss, 1991), which adapts probabilities of ngrams based on the recently encountered n-grams, we adapt the probability of a potential hit, based on the number of neighboring hits for the same query. Specifically, we boost the score of each hit as shown below:

$$score_q(\Delta_t, \delta) = score_q(\Delta_t)^{1/\#hits \in \Delta_{t \pm \delta}} \tag{6.8}$$

where , $\delta \in [0, 1000] secs$ is again tuned on the development set.

6.7 Experimental Results

Table 6.4 illustrates the relative improvement over the two hybrid systems, obtained when incorporating a P2P transducer in deriving several n-best query representations. In all cases, a significant improvement in ATWV performance is seen by increasing the number of hits. However, as expected, increasing phonetic confusibility also increases false alarms and hence we use OOV detection to reduce this effect. The results for the baseline and proposed hybrid systems are comparable, hence we will focus on the effect of the remaining features using the baseline hybrid system.

	Baseline Hybrid			**Proposed Hybrid**		
P2P-Nbest	**Hits**	**FAs**	**ATWV**	**Hits**	**FAs**	**ATWV**
none	7254	10275	0.352	7247	10316	0.351
10 best	7519	14779	0.375	7502	14815	0.376
20 best	7727	17837	0.388	7700	17766	0.390
100 best	8294	27495	**0.399**	8267	27463	**0.399**

Table 6.4: Phone-to-phone transducer for N-best query representation using baseline and proposed hybrid system on OOVCORP

Table 6.5 shows the effect of including: OOV detection (Section 6.5.1), query-length normalization (Section 6.5.2), and cache (Section 6.6.2). We represent the features used in each experiment with a "X" mark against their respective columns. All queries used the 100-best representations generated after composition with the P2P transducer since it yielded the best result in ATWV. When using OOV detection, we show results using the baseline OOV detector (Section 3.5.1) (`Maxent`), and proposed detector (`Context`) from Chapter 4. The performance of these systems is repeated in Figure 7.1.

oovdet (Oracle)	**oovdet (Maxent)**	**oovdet (Context)**	**length-norm**	**cache**	**Hits**	**FAs**	**ATWV**
					8294	27495	0.399
	x				8138	24502	0.403
		x			8253	24152	0.403
			x		9007	24054	0.409
				x	8294	27495	0.399
	x		x	x	9734	39379	**0.420**
		x	x	x	10066	37107	**0.424**
x			x	x	10994	17663	**0.512**

Table 6.5: OOVCORP results using OOV-detector, Length-normalization, and Cache Features. Search was performed on phonetic lattices obtained from the Baseline Hybrid System (Section 3.5.1).

Both length normalization and oov detection achieve improvements in ATWV performance (Baseline: 0.399, OOV-det: 0.403, length-norm: 0.409) and the gains increase when we combine these features (ATWV = 0.424). Note that the proposed OOV detector (`Context`, ATWV = 0.424) achieves improved performance compared to the baseline (`MaxEnt`, ATWV = 0.420), by allowing more hits and reducing false alarms.

Larger gains were expected from the `Context` oov detector given the large improvement in OOV detection performance compared to the baseline `MaxEnt` (Figure 7.1). Error analysis shows the OOV regions selected by the `Context` oov detector included only 5% FAs, while the `MaxEnt` detector had 15% FAs for the selected operating point in the development set. This difference means that the proposed detector reduced the number of incorrectly labeled OOV regions by 110,000. However, this large reduction in false alarms does not translate into the same reduction of false-alarms for STD, since only regions matching the queries phonetically are relevant for STD. Furthermore, the score in mismatched regions is only penalized and does not imply a change in the decision to retrieve that hit. Lastly, the ATWV metric favors having more hits at the expense of larger number of false alarms. This explains why the final system achieves superior performance despite having more false-alarms.

To understand the effect of increasing hits or reducing false alarms in ATWV, we briefly review this metric here. ATWV combines the effect of hits and false alarms as shown in Eq. (6.1)[3], where the false alarm probability for each query term q is given by:

$$P_{FA}(q) = \frac{N_{spurious}(q)}{T - N_{true}(q)} \tag{6.9}$$

and T is the total number of seconds of audio indexed ($\approx 100 hours = 360,000 secs$). This metric assumes one sec of audio corresponds to one potential false alarm. On the other hand, P_{miss} is given by:

$$P_{miss}(q) = 1 - \frac{N_{correct}(q)}{N_{true}(q)} \tag{6.10}$$

where $N_{true}(q)$ is the total number of true instances for query q in the data set. Hence, FAs have a smaller effect (since they are divided by a large number) when compared to hits in ATWV.

Figure 6.2 and 6.3 show results using a DET curve: the percent of misses at different false-alarm rates for the same systems. This curve is obtained by varying a global threshold and accepting all hits with score above this threshold, allowing the user to adjust the

[3] When computing ATWV, we used the same cost factor for FAs as in the NIST 2006 evaluation: $\beta = 999.99$

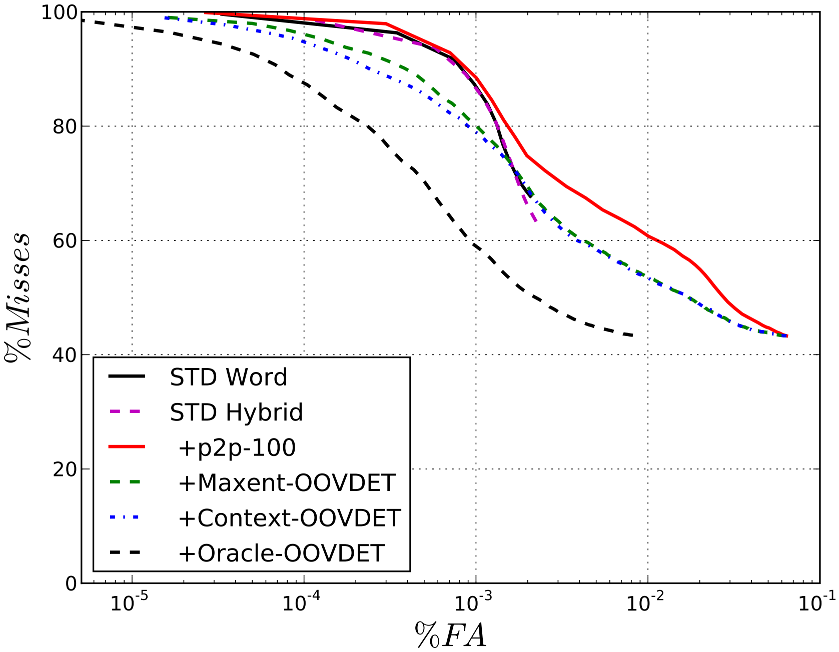


Figure 6.2: STD Det Curve on OOVCORP . Total number of False Alarms is assumed to be the total number of seconds in the audio where the query was not spoken, as in Eq. (6.3).

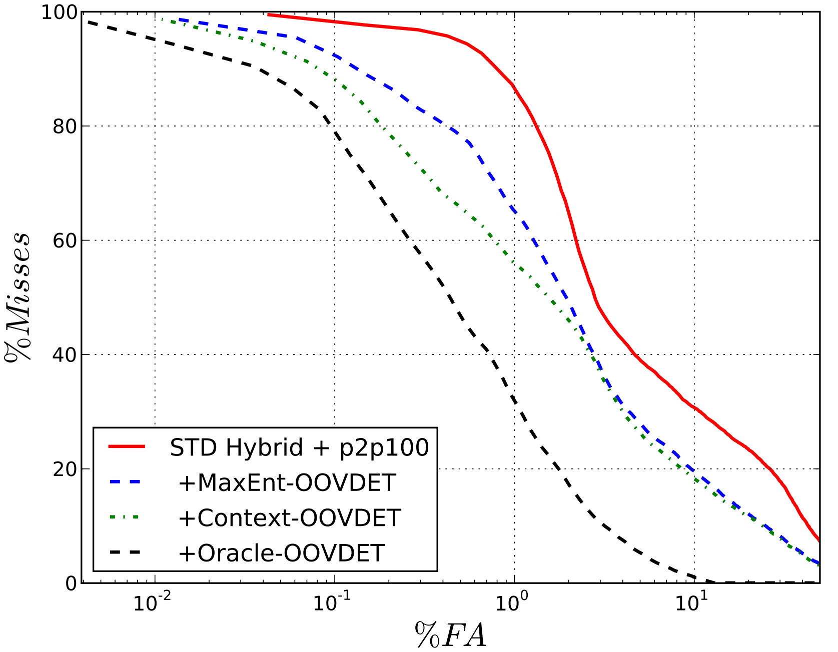


Figure 6.3: STD Det Curve on OOVCORP . Total number of False Alarms is assumed to be the total number of spurious regions matching a query phonetically.

threshold to the application of interest. In Figure 6.2 we assume T is the total number of seconds in the audio (as in ATWV), while in Figure 6.3 we assume T is the total number of regions with a “close” phonetic match to the query set, specifically we use all hits returned by Hybrid + p2p100. The latter allows us to focus on likely false alarms, and appreciate the effect of the OOV detectors in reducing false alarms in phonetically similar regions. We can see a clear improvement by including both OOV detector systems. The proposed detector achieves superior performance for high precision operating points.

We also present results using an oracle OOV detector in Table 6.5, which achieves significant improvements (ATWV = 0.512), when used alone and combined with all other features. This motivates further work in OOV detection. Note that even with oracle OOV detection, false alarms can occur since similar OOV queries can match incorrect query terms in valid OOV regions (i.e. slobodan versus slobodan’s, which are both OOVs).

Lastly, the cache feature (row 5 in Table 6.5) did not provide any gains on their own on this data set. The motivation for the cache feature was to boost scores for rare "content terms” and we expected it to provide significant gains on OOVCORP since OOVs tend to be bursty in nature. Error analysis suggest that OOV terms were not detected closed enough to be exploited by the cache feature. We later realized that this data set was not appropriate for testing trigger or cache features since by design, it contains utterances which are discontinuous for each audio file.

In summary, hybrid systems and the proposed features improve performance for OOV queries in spoken term detection systems. Using the hybrid system to derive the index improved performance from 0.307 to 0.352. Including phonetic confusability further improved to 0.399, and automatic detection and length normalization improved further to 0.424, achieving a total relative improvement of 38%. To the best of our knowledge, OOV detection had not been previously proposed to improve performance of STD systems. Akbacak et al. (2008); Arisoy et al. (2009) proposed a hybrid system to improve STD performance, however their approach first converts the hybrid index to a word-based index and then runs a word-level search. Hence OOVs will only matched if their orthography is correctly recovered.

Dev06 results

Table 6.6 presents the results on the Dev06 data set. While all features provide an incremental gain in ATWV, it can be seen that use of the *cache feature*, i.e. proximity to a potential hit, provides the maximum gain in ATWV by increasing the number of hits. In this data set we are only able to present results using the baseline OOV detector since we did not have available the audio at the time we introduced the new hybrid system or oov-detector. Although very small, the baseline OOV-detector does reduce the false alarms without impacting the number of hits. An oracle experiment that uses true OOV-regions from the reference can reduce the false alarms from 388 to 293, suggesting that more improvements in ATWV can be obtained with better OOV detection.

For Dev06, we did not see any improvements with P2P since there were only 16 OOV terms and the In-Vocabulary terms were most likely captured by the word portion of the hybrid LVCSR system. The highest reported score for the NIST 2006 Spoken Term Detection Evaluation (NIST) on English Broadcast News data (reported here) was 0.85 ATWV (Fiscus, Ajot, Garofolo, and Doddingtion, 2007), achieved by the BBN system.

oovdet (oracle)	oovdet	length-norm	cache	Hits	FAs	ATWV
				4752	388	0.849
	x			4752	383	0.8497
		x		4845	427	0.8520
			x	4907	400	0.8551
	x	x	x	5011	452	**0.8597**
x				4748	232	0.8666
x		x	x	5013	293	**0.8814**

Table 6.6: DEV06 Results using Automatic and Oracle OOV-detector

6.8 Conclusions

In this chapter, we have explored the benefits of using hybrid systems and OOV detection to improve performance of Spoken Term Detection for OOV queries. We also proposed additional features to increase hits and reduce false alarms. All features collectively provide an additive gain on both corpora, resulting in a relative 38% improvement in ATWV on a 100-hour corpus with 1290 OOV-only queries and 2% relative on the NIST 2006 STD task, where only 16 of the 1107 queries were OOV terms.

Using a hybrid system to derive the phonetic index provides significant improvements over a word-based system. While the hybrid system proposed in Chapter 3 performs similar to the baseline hybrid, we do observe significant improvements when including the improved OOV detection system from Chapter 4. Incorporating phonetic confusibility in the query representation also improves performance significantly by increasing the number of retrieved instances.

Chapter 7

OOV Sensitive Named-Entity Recognition in Speech

7.1 Introduction

Named entity recognition (NER) in text, a key step in information extraction, is typically treated as a sequence labeling task in which entities are labeled as people, locations and organizations (Sang and Meulder, 2003). Evaluations have focused on newswire text and manually transcribed broadcast news. However, NER in automatic speech recognition (ASR) produced transcripts is a challenge due to recognition errors and the lack of common named entity markers (punctuation, capitalization, numerals, etc.) Understandably, performance lags behind that of text applications. Attempts to improve speech NER have included transcript normalization (Gravano, Jansche, and Bacchiani, 2009), incorporating speech recognition confidence features (Huang, 2005; Sudoh, Tsukada, and Isozaki, 2006), or tagging LVCSR word lattices (Horlock and King, 2003). A difficult unaddressed problem comes from out-of-vocabulary (OOV) terms. Since many OOVs are proper names (66% of the OOVs in our corpus are named entities) OOV recognition errors are particularly damaging for NER.

We improve speech NER by including features indicative of OOVs obtained from the proposed OOV detection system. This allows us to identify incorrectly decoded sections of speech where a named entity was spoken. Finding such audio regions allows for targeted manual transcription, or automated OOV recovery efforts. To recognize OOV NEs, we augment the features in an NER system to include indications of possible OOVs in the transcript using the OOV detection system presented in Chapter 4. These features yield significant improvements for OOV NEs in particular, as well as NEs in general.

To evaluate our approach, we introduce a new broadcast news speech data set annotated for named entities using Amazon Mechanical Turk. We describe the methods used to create this data set and its properties.

7.2 Named Entity Recognition for OOVs

Out-of-Vocabulary words often include named entities; in our OOVCORP data set 66% of the OOVs are named entities, accounting for 21% of all named entities. This problem is often ignored in NER in speech (Miller et al., 2007; Sudoh et al., 2006); and some cope with OOV entities by adapting the vocabulary and the language model to the specific time interval of the test set (Favre, Bechet, and Nocera, 2005).

To recognize OOV NEs, we augment a standard NE tagger to include features indicative of OOV terms. The tagger should ignore the decoded words for OOV regions and rely on context to identify the named entity. For example, if the tagger sees the string "FORMER PRESIDENT MOST OF IT SAID" it would likely find no named entity. However, "MOST OF IT" is an obvious transcription error (for "MILOSEVIC") and if the tagger knew "MOST OF IT" was OOV, it could focus on context ("Former President X said") and identify the audio corresponding to "X" as a named entity.

Our work is similar to that of Huang (2005) and Sudoh et al. (2006) Huang uses a confidence based approach to identify transcript errors and ignores the decoded word sequence in the error region, using the context to query relevant documents for OOV recovery. He uses features from the recovered word and its context as input for a standard NER system. In this work, we are concerned with identification and not recovery. However, identified named entities in incorrectly transcribed audio could be targeted for recovery using an OOV recovery system such as the one described in Chapter 5. Both Huang and Sudoh et al. rely on the word posterior probability as a confidence metric. Sudoh et al. combine this metric with the decoded word sequence and contextual POS tag information using SVMs to detect unreliable regions. We consider a similar approach (errordet) as a baseline.

Our approach uses the output of an OOV detector as indicative of NE regions. In the next section, we introduce our NE tagger and describe how we incorporate OOV information.

7.2.1 Named Entity Tagger and OOV Detector

We use a conditional random field (CRF) based named entity tagger (McCallum and Li, 2003), with a first order Markov model, BIO encoding (B-ORG, I-ORG, etc.), and a standard set of orthographic features (McDonald and Pereira, 2005) (Baseline). We used the default parameters in Mallet [1] and a Gaussian prior of $\sigma^2 = 10$ (results were generally insensitive to σ.) The number of training iterations was selected using development data. On CONLL 2003 English data (Sang and Meulder, 2003), the tagger achieves an overall development F1 of 88.34 and test F1 of 81.41, which is close to state of the art on this task.

We use the OOV detector presented in Chapter 4 to incorporate OOV information into the tagger. Several of the baseline NER features use part of speech tags. Since most POS taggers are trained for text, we obtained tags for speech by using a syntactic language model (Filimonov and Harper, 2009a,b), which estimates the joint probability of the word and its syntactic tag based on the preceding words and tags (a trigram context.) This tagger can produce tags for suspected OOVs.

We trained the language model on 130 million words from Hub4 CSR 1996 (Garofolo et al., 1996). Tags, extracted from parse trees from a modified Berkeley parser (Huang

[1] http://mallet.cs.umass.edu

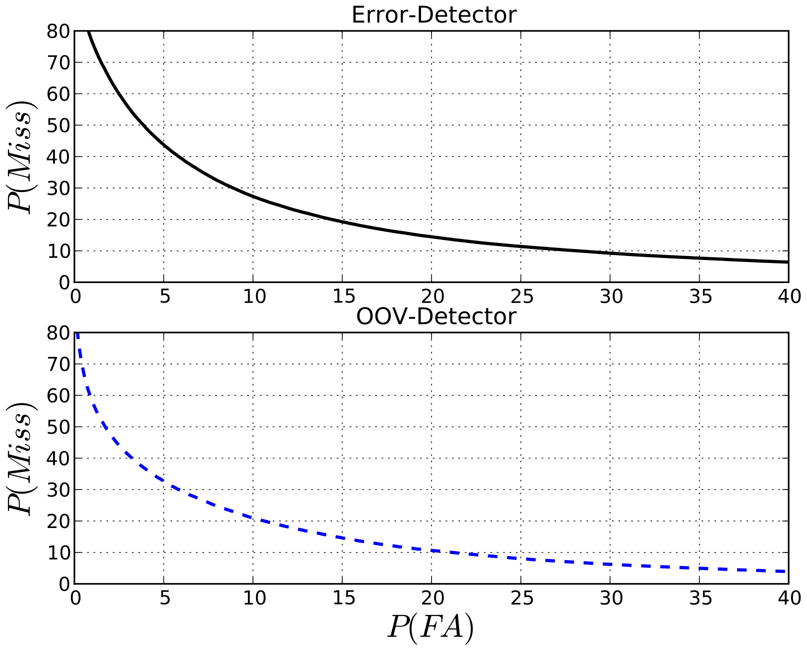


Figure 7.1: Test set (90 hours) performance: Error detector (Sudoh et al., 2006) on errors (top) and OOV detector (Chapter 4) on OOVs (bottom.)

and Harper, 2009), incorporated the word's POS, the label of its immediate parent, and the relative position of the word among its siblings. [2] Since we are considering OOV detection, the language model was restricted to the LVCSR system's vocabulary.

7.2.2 Confidence Baseline

A common approach to improve NER performance on speech is to incorporate a confidence estimate for predicting decoding errors (such as those caused by OOVs) (Huang, 2005; Sudoh et al., 2006) . Therefore, we compare our approach with both a baseline NER

[2] The *parent* tagset of Filimonov and Harper (Filimonov and Harper, 2009a).

feature set described in Section 7.2.1, and with an additional confidence estimate baseline (Figure 7.1.)

The confidence baseline uses the features of Sudoh et al. (Sudoh et al., 2006) to create a CRF error predictor: the decoded word, POS tag, and posterior probability, as well as these features from a ±2 word window. This system shows superior error detection performance to only using the word posterior probability. The training data was obtained from a standard word-based LVCSR system whose errors are known by aligning with the reference transcription. The probability of error, provided by the CRF error predictor, was quantized into 10 bins generating binary features (errordet).

7.2.3 NER with OOVs

We incorporate OOV information into the NER tagger by generating features based on the OOV detector. Our goal was to inform the NER tagger when we suspected that a word may be an OOV, which could make it more likely to be a named entity. We also sought to remove unreliable features, i.e. incorrectly decoded words. We developed three feature sets:

- **oovdet**: The probability of a word being OOV according to the OOV detector.[3]
- **context**: The oovdet confidence feature from a ±2 word window around the current word.

[3] Real valued features were quantized into 10 bins (binary features.)

		Overall			**IV entities**		
OOV	System	F1	Prec	Rec	F1	Prec	Rec
-	Baseline	58.5	64.7	53.3	59.9	64.6	55.9
	errordet	58.6	66.5	52.4	60.2	66.9	54.7
Auto	oovdet	60.1	66.4	54.9	61.7	66.6	57.5
	oovdet+context	59.8	66.1	54.6	61.0	65.8	56.8
	oovdet+replace	59.8	67.9	53.5	61.4	68.3	55.7
	oovdet+context+replace	**60.7**	68.2	54.7	**62.2**	68.4	57.1
Oracle	oov	61.7	68.2	56.4	61.7	70.2	55.0
	oov+oovdet+context+replace	**62.3**	68.0	57.5	**62.1**	68.6	56.8

Table 7.1: Performance for all named entities (Overall) and for entities containing all words in the LVCSR vocabulary (In-Vocabulary or IV entities)

- **replace**: Replace the decoded word with the token `OOV` if the detector has a confidence threshold above 0.9 (tuned on development data.) This explicitly removes the confusion of an incorrectly decoded word, and the system must rely on the context to tag the words, as well as a prior that OOVs may be NEs.

7.3 Experiment Setup

Again, we use the OOVCORP data set to evaluate our system. From the 100 hours in OOVCORP, five hours were used for training the OOV and error detectors, and 48 hours were annotated for named entity training and evaluation. From this set, 25 hours were used for NE tagger training, 5 hours for development, and the remaining 18 hours for testing.

		OOV entities			**Unobserved OOVs**		
OOV	System	F1	Prec	Rec	F1	Prec	Rec
-	Baseline	51.0	63.4	42.7	29.9	41.5	23.4
	errordet	51.3	63.8	42.9	36.8	50.5	28.9
Auto	oovdet	52.9	64.4	44.9	37.5	49.5	30.1
	oovdet+context	54.2	66.6	45.7	33.7	45.8	26.7
	oovdet+replace	52.4	65.1	43.9	35.6	48.0	28.3
	oovdet+context+replace	**53.6**	66.2	45.1	**37.7**	50.5	30.1
Oracle	oov	61.2	62.5	60.0	47.2	47.9	46.5
	oov+oovdet+context+replace	**61.7**	65.9	58.0	**52.0**	54.1	50.0

Table 7.2: Performance for entities containing at least one OOV word (OOV entities), and for OOV entities which are not present in training for NER system (Unobserved OOVs).

7.3.1 Named Entity Annotations

The 48 hours set aside for named entity training and evaluation did not contain named entity annotations. We annotated this data using Amazon Mechanical Turk (MTurk), which provides a platform where human intelligence tasks (HITs) can be given to users (turkers) for annotation. HITs tend to be simple tasks that are easy for humans to accomplish but remain challenging for computers, such as labeling images or translating sentences. Turkers are paid a micro-payment for each HIT, typically a few cents. Studies have shown that high quality annotations are possible by using multiple turkers, simple HITs, and embedded gold standard examples to evaluate quality (Callison-Burch and Dredze, 2010).

We used MTurk to obtain named entity labels for the manual speech transcripts from our corpus. Our annotation guidelines were based on the CONLL shared task (Sang and Meulder, 2003). We used an interface similar to that presented in Finin et al. (Finin, Murnane, Karandikar, Keller, Martineau, and Dredze, 2010), modified to allow users to indicate the presence of two adjacent named entities of the same type, such as “Thanks Jim Sarah reporting live from Boston". Here Jim and Sarah are two different people with a pause in the speech that does not appear in the transcript. Such occurrences are exceedingly rare in text, but common in speech where pauses or speaker turn taking creates sequential named entities. Each HIT contained 5 speech utterances, 1 of which was an utterance chosen from 250 utterances for which we obtained expert NE annotation (provided by the authors) included for quality control. Each HIT was completed by two different turkers at a rate of $0.10 each, yielding a rate of $0.02 an utterance. Utterances completed by unreliable turkers (poor scores on the included gold examples) were resubmitted to obtain additional annotations. Total cost was $530, including repeated annotations and productivity bonuses. Details on annotation instructions are in Appendix B.

For each turker, we evaluated his or her average F1 score on gold utterances and removed annotations by turkers with an F1 score below 0.5. We then compared the two annotations for each utterance and selected labels agreed to by both turkers. In cases of dispute, we select the label assigned by the turker with highest F1. This yielded a total of 9971 annotated utterances (510K tokens). The inter-annotator agreement between turkers computed using Cohen's κ (Fleiss and Cohen, 1973) was .72.[4]

The average F1 score of the final turker annotations on the 257 utterances with gold annotations was 87%. The 9971 utterances contain 34,293 named entities, 14,967 people, 10,680 locations and 8,646 organizations. In this data, 21% of the named entities are OOVs

[4] Kappa coefficient can be interpreted as: $0 - .2$ slight, $.2 - .4$ fair, $.4 - .6$ moderate, $.6 - .8$ substantial and higher is almost perfect (Landis and Koch, 1977).

(38.32% of PER, 6% of LOC, and 14% of ORG) and 66% of the OOVs are named entities.

7.4 Results

We evaluated our named entity tagger with the various OOV feature sets and two baselines: standard NER features (baseline), and the addition of error detection features (errordet). We report F1 results overall, for entities containing all words in the vocabulary (In-Vocabulary or IV entities), and for entities containing at least one OOV word (OOV entities) (Table 7.2). While the error detector (errordet) yields a small improvement (0.15), the equivalent OOV features (oovdet) yield a much larger improvement (1.67). Using all of our OOV features (oovdet+context+replace) achieves a 2.25 improvement over the baseline.[5] Example improvements included the utterance “opposition claims VOJISLAV KOSTUNICA should be declared winner", which is decoded as: “opposition claims BORISLAV CUSTOM ME JUST should be declared winner". The named entity “VOJISLAV KOSTUNICA" is mis-recognized because both words are OOVs, but the improved system correctly labeled it as a person.

Note that in our data set partitioning, the tagger may learn the context of an OOV in the NER training set, which matches the context for that same OOV in test, allowing the system to correctly label it as an entity. However, in a real application, with a constantly changing vocabulary, OOVs seen during the tagger training are likely to be different from those in

[5] Statistically significant at $p = 0.001$ using the paired permutation test.

the test set. To evaluate this scenario, we report performance of “Unobserved OOV" entities: named entities containing words which are not in the recognizer’s vocabulary, and are unobserved in the training set for the OOV detector or NER training and development set. These words only appear in the test set, never in the training or development set. As expected, the performance for these entities is lower than the overall OOV performance, however the proposed system (oodet+context+replace) achieves a 7.8 absolute improvement over the baseline (29.9 to 37.7). Eighteen percent of all OOV entities in the test-set were unobserved.

Additionally, we achieve improvements in performance for IV entities, where there is a large increase in precision (64.58 to 68.42). We attribute this gain to the fact that now the learner is not forced to mark common word strings like “most of it" (for MILOSEVIC) or “custom me just" (for KOSTUNICA) as named entities unless the OOV features indicate otherwise. Furthermore, these incorrectly decoded words are replaced by `OOV`, removing misleading features.

We also sought to determine additional gains we might achieve by improved OOV detection. We replaced the OOV detector by an oracle predictor, manually tagging OOV regions by finding time segments in the manual transcripts containing words which are not in the LVCSR system vocabulary. True OOV regions were marked with a new feature (oov.) The best performance was obtained when adding this oracle OOV feature, replacing the decoded word by “OOV" if this feature fires, and including the context feature described in Section 7.2.3 (oov+oovdet+context+replace). This improves an additional 8.08

points on OOVs over the best OOV predictor results, and 1.61 improvement overall. This demonstrates that our automatic results achieve an almost 60% error reduction towards oracle OOV detection. Additionally, the oracle results achieve similar performance for OOV and IV, indicating that remaining NER errors may not be attributed to OOVs and that given correct OOV predictions, our NE tagger effectively addresses the OOV NER problem.

7.5 Conclusion

We have presented a novel approach for OOV sensitive named entity recognition in automatically transcribed speech, targeting NEs containing words which are not present in the LVCSR system's vocabulary. We augmented the features used by a CRF NER tagger to indicate possible OOVs in the transcript. Our system obtains a statistically significant improvement in overall performance using automatic OOV detection and our automatic results achieve an almost 60% error reduction over the baseline compared to oracle results. Additionally, we show that oracle OOV features close the gap between IV and OOV NER performance.

Chapter 8

Conclusion

In this dissertation we have focused on the out-of-vocabulary (OOV) problem in Large Vocabulary Continuous Speech Recognition (LVCSR) systems. We presented an approach to recover from failures caused by OOVs by automatically identifying when OOVs are spoken and transcribing them using sub-lexical units. This results in a hybrid word/sub-word system which predicts full-words for in-vocabulary terms and sub-lexical units for OOVs.

In Chapter 3 we presented an approach to learn optimal sub-word representations to model OOVs in a hybrid system. The proposed sub-words were combined with confidence estimation methods in Chapter 4 to improve detection of Out-of-Vocabulary words in the LVCSR output. These methods improved OOV detection performance by 32% absolute on a Broadcast News task and by 12.3% on a MIT Lectures data set, and achieved reductions in phone error rate, specially for OOV regions.

The remaining chapters explored the benefits of the improved hybrid system and OOV detection systems in downstream applications. Chapter 5 proposed an approach to recover the orthography of novel words for transcription. Chapter 6 showed significant performance improvements in Spoken Term Detection when employing the proposed hybrid and OOV detection system. Finally, Chapter 7 exploited these systems to improve information extraction tasks on speech, focusing on named entity recognition for OOV words..

There are several novel contributions in this dissertation, which we detailed below.

Contributions of Chapter 3: Learning Sub-Word Units for Open Vocabulary ASR

- Proposed an unsupervised log-linear model to learn sub-lexical representations optimized to a given labeling task.
- Applied this model to open-vocabulary recognition and learned variable-length multi phone units used to model OOVs in a hybrid word/sub-word system. The resulting system improves performance over state-of-the-art in terms of OOV detection and phone-error-rate.
- The proposed model is general and can be applied to any unsupervised segmentation task, where the segmentation depends on a label assigned to each word in the sequence.

Contributions of Chapter 4: Exploiting Context for OOV Detection

- Proposed to treat OOV detection as a sequence labeling task, and showed that jointly predicting OOV regions improves performance.

- Exploited contextual information for OOV detection, both from the local lexical context and from the global context using language models.

- Evaluated the resulting system on two data-sets achieving large reductions in error.

Contributions of Chapter 5: Recovering Out-of-Vocabulary Words

- Proposed a novel approach to recover out-of-vocabulary words using the web as a corpus. This system integrates novel words using a spoken term detection system (no-redecoding) or by re-decoding with an augmented lexicon.

- Evaluated the resulting system on the 90 hours of Broadcast News corpus assuming OOV regions are identified using automatic OOV detection. The system recovers up to 21% of OOVs using automatic detection, achieving 0.7% absolute improvement in WER.

Contributions of Chapter 6: Spoken Term Detection with OOV Queries

- Evaluated the benefits of using hybrid systems and OOV detection to improve performance for OOV queries in a spoken term detection system. Using a hybrid system to obtain the phonetic index retrieves more instances while improved OOV detection can reduce false-alarms.

- Proposed algorithmic improvements to increase hits for OOV queries, such as: including phonetic confusability in the query representation and boosting the score in accordance to the number of neighboring hits.
- Evaluated these features as a post-processing step in a spoken term detection system achieving gains in two corpora: 90 hours of Broadcast News corpus with 1290 OOV queries, and the NIST 2006 STD task where only 16 of the 1107 queries were OOVs.

Contributions of Chapter 7: OOV Sensitive NER in Speech

- Designed a system is capable of identifying incorrectly decoded sections of the speech where an OOV named entity was spoken, improving performance for OOV named entities (improved recall) and IV named-entities (reduced false-alarms).
- Evaluated the benefits of using OOV detection to improve retrieval of OOV named-entities from speech content.
- Introduced a new Broadcast News speech data set annotated for named-entities using Amazon Mechanical Turk.

8.1 Future Directions

While we made significant advances towards identifying and modeling OOVs for open vocabulary recognition, there are many other extensions that can be done to either enhance performance or apply the model to wide range of tasks. In this section we describe some directions for future work.

In Chapter 3 we presented a log-linear model to learn the sub-word lexicon by segmenting the phonetic representation of a corpus. While our approach can find the optimal segmentation for any labeling sequence, we used a simple binary labeling identifying OOVs in the text. This approach essentially models all OOVs as a single class. However, OOVs can be very different types including named-entities, technical terms, rare verb forms, out-of-language terms, etc. Indeed, it has been shown (Bazzi, 2002) that considering multiple OOV models improves detection performance. This could be tested directly in our model by allowing the labeling sequence to take more than 2 values. It is still non-trivial to determine which classes to consider for the different OOV models. One approach is to allow the model to assign different OOVs to different classes by including the class label as a latent variable.

In Chapter 4, we combine the hybrid system from Chapter 3 with confidence estimation features to improve OOV detection. The approach we outline only considers the probability of decoded sub-words as features in the model, completely ignoring the identity of the sub-word predicted. The model used to learn the sub-word lexicon provides a probability estimate for each sub-word as predictive of each class label. This could also be considered as a feature in the OOV detection system. Several other confidence-based features should be explored such as those proposed by Burget et al. (2008). It is straight forward to integrate other confidence metrics.

Other Interesting Directions:

Once an OOV is uttered, it is very likely to be repeated in that same conversation. The models presented in this dissertation do not exploit this property. An interesting research direction is to build a global model that exploits this bursty nature of rare content words to identify and group OOV instances into clusters corresponding to the same word. Each cluster can provide multiple pronunciations and contexts relevant for a given OOV word. This should be useful to improve detection of OOVs and to recover their correct spelling.

Identifying OOV clusters could also be useful for unsupervised online adaptation of the vocabulary and language model, resulting in a LVCSR system that adapts over time. Some previous work in this direction include: self-training for low-resource language modeling (Novotney, Schwartz, and Ma, 2009), and online vocabulary adaptation (Aronowitz, 2008; Oger et al., 2009). These methods rebuild all models on the entire training set instead of adapting them online.

The proposed model for learning the sub-word lexicon is a general segmentation approach optimized for a given task. It would be interesting to evaluate its performance for other tasks, such as spoken term detection, or morphological segmentation with POS-tag information or other type of class-labels.

An interesting application for the proposed segmentation model is word-compounding for language modeling. In this case the goal is to find word compounds (i.e "in_the", "i_am") that are maximally different from their common errorful transcriptions. This approach could segment the text corpus into phrases by maximizing the likelihood of the segmentation and the text (no labels). Similar to the approach proposed by Poon et al. (2009), it could use contrastive estimation (Smith and Eisner, 2005) to simplify inference by limiting negative examples to words in a neighborhood. To ensure selected phrases are maximally different from their common errors, the neighborhood could be obtained from the ASR confusion network cohorts for the phrase in question.

Other interesting directions of research include: identifying other types of non-OOV errors, such as partially spoken words, or errors caused by unexpected noise such as: door-closing, cough, scream.

Appendix A

Finite State Automata Overview

Throughout this dissertation we use finite state automata extensively to efficiently represent sets of strings or string pairs and associated probabilities. Weighted finite state automata are popular tools in speech recognition (Mohri, 1997), and many downstream applications such as speech to speech translation (Knight and Al-Onaizan, 1998), spoken document retrieval (J. Mamou and Hoory, 2006), spoken term detection (Can et al., 2009; et al, 2009; Arisoy et al., 2009), etc.

Finite state automata includes *finite state acceptors* which represent sets of strings and *finite state transducers* which represent mappings between two sets of strings. Both can be weighted or unweighted. A finite state acceptor can be considered as finite state transducer were the input string maps to itself. Therefore we only present here the formal definition for weighted finite state transducers following (Mohri, 2009). In order to define a weighted finite state transducer and allowed operations we need to first define a semiring:

Definition (Mohri, 2009) A system $(W, \oplus, \otimes, \bar{0}, \bar{1})$ is a semiring if $(W, \oplus, \bar{0})$ is a commutative monoid with identity element $\bar{0}$, $(W, \otimes, \bar{1})$ is a monoid with identity element $\bar{1}$, $\otimes$ distributes over $\oplus$, and $\bar{0}$ is an annihilator for $\otimes$. Table A lists several semirings. In this dissertation we only make use of the Log and Tropical semiring.

Definition (Mohri, 2009) A weighted transducer T over a semiring $(W, \oplus, \otimes, \bar{0}, \bar{1})$ is an 8-tuple $T = (\Sigma, \Delta, Q, I, F, E, \lambda, \rho)$ where:

- Σ is a finite input alphabet
- Δ a finite output alphabet
- Q a finite set of states
- $I \subseteq Q$ the set of initial states
- $F \subseteq Q$ the set of final states

Semiring	Set	$\oplus$	$\otimes$	$\bar{0}$	$\bar{1}$
Boolean	$\{0, 1\}$	$\vee$	$\wedge$	0	1
Probability	$\Re_+ \cup \{+\infty\}$	$+$	$\times$	0	1
Log	$\Re \cup \{-\infty, +\infty\}$	$\oplus_{log}$	$+$	$+\infty$	0
Tropical	$\Re \cup \{-\infty, +\infty\}$	min	$+$	$+\infty$	$+$

Table A.1: Example semirings. $\oplus_{log}$ is defined as: $x \oplus_{log} y = -\log(\exp^{-x} + \exp^{-y})$.

- E a finite multi-set of transitions, which are elements of $Q \times (\Sigma \cup \{\epsilon\}) \times (\Delta \cup \{\epsilon\}) \times W \times Q$,

- $\lambda : I \rightarrow W$ an initial weight function

- $\rho : F \rightarrow W$ a final weight function mapping F to W.

Weighted finite state transducers (WFSTs) can be manipulated and combined with other automata using several algorithms. In this dissertation we make use of the following operations which are described in detail in (Mohri, 2009).

- composition (also denoted by the ○ symbol)

- projection

- minimization

- determinization

- epsilon removal

- shortest path

- shortest distance

- randgen

In the context of this dissertation weighted finite state acceptors are used to represent lattices and confusion networks, where each each path in the acceptor represents a hypothesis of the recognizer. Its weight is the negative log probability of that hypothesis. In this context, the shortest path algorithm in the Tropical semiring is often used to obtain the most likely word sequence from a network.

Weighted finite state transducers are used to represent the index in a spoken term detection system in Chapter 6. In this case the input alphabet are the words/sub-words and the output alphabet are the time/utterance id information. The weights in this case represent the expected count of that word/sub-word in the lattice/confusion-network. We also use WFSTs in Chapter 3 to represent compactly all possible segmentations of a word. We use the resulting transducer to do efficient sampling using operations such as shortest distance in the Log semiring to compute forward/backward probabilities; and *randgen* to sample a path through the network according to the distribution of the different segmentations encoded in the weights.

By convention, ϵ is a special symbol which consumes no symbol and matches all symbols in the input and output alphabets. Typically the silence (`SIL`) and hesitation (`%HESITATION`) symbols in the lattice/confusion-networks are converted to ϵ arcs, so that they can be ignored when composing with other automata. For example: when searching for a phrase “`president %HESITATION SIL fujimori`" converting the `%HESITATION` and `SIL` symbols to $epsilon$ allows the search algorithm to match “`president fujimori`”.

Appendix B

Creating NER Dataset: Annotation Guidelines

The following directions were provided to Turkers:

An entity is a object in the world like a place or person. A named entity is a phrase that uniquely refers to an object by its proper name (Hillary Clinton), acronym (IBM) or abbreviation (Minn.). Here are some more examples of named entities for each of the types we are interested in. Note that there is no capitalization in the sentences.

- Person: first, middle, and last names of people, animals and fictional characters.

  Person examples: barack obama; palins; john; terry lewis;

- Organization: companies, brands, political movements, government bodies (councils, courts, ministries), musical companies, public organizations (schools, universities, charities), other collections of people (sport clubs, associations, etc.)

  Organization examples: c.n.n., the white house, congress., valujet; the washington post; oxford university (when considered in context as an organization);

- Place: roads, regions (towns, cities, countries, etc.), natural locations (mountains, valleys, national parks, etc.), public places (squares, museums, airports, stations, hospitals, parks, universities, etc.), commercial places (pubs, restaurants, hotels, etc.), assorted buildings (houses, halls, rooms, castles), abstract places ("the free world")

  Place examples: baltimore, md; washington; dade county, florida; mt. everest; hoover dam; u.s.; oxford university (when considered in context as a location);

When tagging named entities remember to:

- If you find two consecutive entities of the same type (two people mentioned consecutively,) indicate the new person by selecting the This entity is distinct from the preceding entity</strong> checkbox.

- Pronouns (me, I, we, they) should not be tagged.

- You do not have to tag words like "British", which is a nationality.

- Tag words according to their meaning in the context of the sentence.

- Only tag names, i.e. words that directly and uniquely refer to entities.

- Only tag names of the types Person, Organization, and Place.
- You can check the ??? box to indicate something you consider to be ambiguous or that you are uncertain about.

Bibliography

M. Akbacak, D. Vergyri, and A. Stolcke. Open vocabulary spoken term detection using graphone-based hybrid recognition systems. *ICASSP*, 2008.

C. Allauzen, M. Mohri, and Murat. General indexation of weighted automata - application to spoken utterance retrieval. In *North American Chapter of the Association for Computational Linguistics - Human Language Technologies (NAACL-HLT)*, 2004.

C. Allauzen, M. Riley, J. Schalkwyk, W. Skut, and M. Mohri. Openfst: A general and efficient weighted finite-state transducer library. In *Conference on Implementation and Application of Automata (CIAA)*, pages 11–23, 2007.

E. Arisoy, D. Can, S. Parlak, H. Sak, and M. Saraclar. Turkish broadcast news transcription and retrieval. *IEEE Transactions on Audio, Speech, and Language Processing*, 17(5): 874–883, 2009.

H. Aronowitz. Online vocabulary adaptation using contextual information and information retrieval. *Interspeech*, 2008.

A. Asadi, R. Schwartz, and J. Makhoul. Automatic detection of new words in a large vocabulary continuous speech recognition system. *ICASSP*, pages 125–128, 1989.

L. Bahl and F. Jelinek. Decoding for channels with insertions, deletions, and substitutions with applications to speech recognition. *IEEE Transactions on Information Theory*, 1975.

J. Baker. The dragon system-an overview. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, ASSP-23(1):24–29, 1975a.

J. Baker. *Stochastic modeling for automatic speech understanding, Speech Recognition*. Academic Press, 1975b.

L. Baum and T. Petrie. Statistical inference for probabilistic functions of finite state markov chains. *Annals of Mathematical Statistics*, 37:1559–1563, 1966.

I. Bazzi. *Modeling Out-of-Vocabulary Words for Robust Speech Recognition*. PhD thesis, Massachusetts Institute of Technology, 2002.

I. Bazzi and J. Glass. Heterogeneous lexical units for automatic speech recognition: Preliminary investigations. *ICASSP*, pages 1257–1260, 2000.

I. Bazzi and J. Glass. Learning units for domain-independent out-of-vocabulary word modelling. In *Eurospeech*, 2001.

M. Bisani and H. Ney. Open vocabulary speech recognition with flat hybrid models. In *INTERSPEECH*, 2005.

M. Bisani and H. Ney. Joint-sequence models for grapheme-to-phoneme conversion. *Speech Communication*, 50(5):434–451, 2008.

L. Burget, P. Schwarz, P. Matejka, M. Hannemann, A. Rastrow, C. White, S. Khudanpur, H. Hermansky, and J. Cernocky. Combination of strongly and weakly constrained recognizers for reliable detection of OOVS. *ICASSP*, 2008.

C. Callison-Burch and M. Dredze. Creating speech and language data with amazon's mechanical turk. In *Workshop on Creating Speech and Language Data With Mechanical Turk at NAACL-HLT*, 2010.

D. Can, E. Cooper, A. Sethy, C. White, B. Ramabhadran, and M. Saraclar. Effect of pronounciations on OOV queries in spoken term detection. In *ICASSP*, 2009.

S. Chen, B. Kingsbury, L. Mangu, D. Povey, G. Saon, H. Soltau, and G. Zweig. Advances in speech transcription at ibm under the darpa ears program. *IEEE Transactions on Audio, Speech, and Language Processing*, pages 1596–1608, 2006.

S. F. Chen. Conditional and joint models for grapheme-to-phoneme conversion. In *Eurospeech*, 2003.

S. F. Chen and J. Goodman. An empirical study of smoothing techniques for language modeling. In A. Joshi and M. Palmer, editors, *Proceedings of the Thirty-Fourth Annual Meeting of the Association for Computational Linguistics*, pages 310–318, San

Francisco, 1998. Morgan Kaufmann Publishers. URL citeseer.ist.psu.edu/article/stanley96empirical.html.

G. Choueiter. *Linguistically-motivated sub-word modeling with applications to speech recognition*. PhD thesis, Massachusetts Institute of Technology, 2009.

J. Cohen, T. Kamm, and A. Andreou. Vocal track normalization in speech recognition: compensating for systematic speaker variability. *190th Meeting of the Acoustical Society of America*, 97(5):3246–3247, 1995.

M. Creutz and K. Lagus. Unsupervised discovery of morphemes. In *Proceedings of the ACL-02 workshop on Morphological and phonological learning-Volume 6*, pages 21–30, 2002.

M. Creutz and K. Lagus. Unsupervised morpheme segmentation and morphology induction from text corpora using morfessor 1.0. *Computer and information science, Report A*, 81, 2005.

M. Creutz, T. Hirsimaki, M. Kurimo, A. Puurula, J. Pylkkonen, V. Siivola, M. Varjokallio, E. Arisoy, M. Saraclar, and A. Stolcke. Analysis of morph-based speech recognition and the modeling of out-of-vocabulary words across languages. *HLT-Naacl*, pages 380–387, 2007.

M. Creutz, S. Virpioja, and A. Kovaleva. Web augmentation of language models for continuous speech recognition of sms text messages. In *EACL*, 2009.

S. 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.

B. Decadt, J. Duchateau, W. Daelemans, and P. Wambacq. Transcription of out-of-vocabulary words in large vocabulary speech recognition based on phoneme-to-grapheme conversion. *ICASSP*, 2002.

S. Deligne, F. Yvon, and F. Bimbot. Variable-length sequence matching for phonetic transcription using joint multigrams. In *Eurospeech*, pages 2243–2246, 1995.

A. Dempster, N. Laird, and D. Rubin. Maximum likelihood from incomplete data via the em algorithm. *Royal Statistical Society*, 39(1):1–38, 1977.

E. Eide and H. Gish. A parametric approach to vocal track length normalization. *IEEE International Conference on Acoustics Speech and Signal Processing (ICASSP)*, pages 346–348, 1996.

E. C. et al. Web derived pronunciations for spoken term detection. In *SIGIR*, 2009.

L. B. F. Jelinek and R. Mercer. Design of a linguistic statistical decoder for the recognition of continuous speech. *IEEE Transactions on Information Theory*, IT-21:250–256, 1975.

B. Favre, F. Bechet, and P. Nocera. Robust named entity extraction from large spoken archives. In *EMNLP*, 2005.

D. Filimonov and M. Harper. A joint language model with fine-grain syntactic tags. In *EMNLP*, 2009a.

D. Filimonov and M. Harper. Measuring tagging performance of a joint language model. In *INTERSPEECH*, 2009b.

T. Finin, W. Murnane, A. Karandikar, N. Keller, J. Martineau, and M. Dredze. Annotating named entites in twitter data with crowdsourcing. In *Workshop on Creating Speech and Language Data With Mechanical Turk at NAACL-HLT*, 2010.

J. Fiscus, J. Garofolo, M. Przybocki, W. Fisher, and D. Pallett. *1997 English Broadcast News Speech (HUB4)*. Linguistic Data Consortium, 1998.

J. Fiscus, J. Ajot, J. Garofolo, and G. Doddingtion. Results of the 2006 spoken term detection evaluation. In *Interspeech*, 2007.

J. Fleiss and J. Cohen. The equivalence of weighted kappa and the intraclass correlation coefficient as measures of reliability. *Educational and Psychological Measurement*, 33: 613–619, 1973.

M. 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. *Eurospeech*, pages 249–252, 2003.

J. Garofolo, J. Fiscus, W. Fisher, and D. Pallett. *CSR-IV HUB4*. Linguistic Data Consortium, 1996.

J. Glass, T. Hazen, L. Hetherington, and C. Wang. Analysis and processing of lecture audio data: Preliminary investigations. In *North American Chapter of the Association for Computational Linguistics (NAACL)*, 2010.

J. Godfrey, E. C. Holliman, and J. McDaniel. Switchboard: Telephone speech corpus for research and development. In *IEEE International Conference on Acoustics, Speech, and Signal Processing*, pages 517–520, 1992.

A. Gravano, M. Jansche, and M. Bacchiani. Restoring punctuation and capitalization in transcribed speech. In *ICASSP*, pages 4741–4744, 2009.

A. Gunawardana, M. Mahajan, A. Acero, and J. C. Platt. Hidden conditional random fields for phone classification. In *Proceedings of Interspeech*, 2005.

Haeb-Umbach and Ney. Linear discriminat analysis for improved large vocabulary continuous speech recognition. *ICASSP*, 1:13–16, 1992.

T. J. Hazen and I. Bazzi. A comparison and combination of methods for OOV word detection and word confidence scoring. In *Proceedings of the International Conference on Acoustics*, 2001.

T. J. Hazen, I. Hetherington, H. Shu, and K. Livescu. Pronunciation modeling using a finite-state transducer representation. *Speech Communications*, 46(2):189–203, 2005.

H. Hermansky. Perceptual linear predictive analysis of speech. *Journal of Acoustic Society of America*, 87(4):1738–1752, 1989.

I. L. Hetherington. A characterization of the problem of new, out-of-vocabulary words in continuous-speech recognition and understanding. *Thesis*, 1995.

J. Horlock and S. King. Named entity extraction from word lattices. In *Eurospeech*, 2003.

F. Huang. *Multilingual Named Entity Extraction and Translation from Text and Speech.* PhD thesis, Carnegie Mellon University, 2005.

Z. Huang and M. Harper. Self-Training PCFG grammars with latent annotations across languages. In *EMNLP*, 2009.

D. C. J. Mamou and R. Hoory. Spoken document retrieval from call-center conversations. *SIGIR*, 2006.

F. Jelinek. *Statistical Methods for Speech Recognition*. MIT Press, 1997.

F. Jelinek and R. L. Mercer. Interpolated estimation of markov source parameters form sparse data. In *Proceedings of the Workshop on Pattern Recognition in Practice*, Amsterdam, The Neterlands, 1980.

F. Jelinek, R. Mercer, and S. Roukous. Classifying words for improved statistical language models. *ICASSP*, 1990.

F. Jelinek, B. Merialdo, S. Roukos, and M. Strauss. A dynamic language model for speech recognition. In *Human Language Technology Conference (HLT)*, 1991.

S. Katz. Estimation of probabilities from sparse data for the language model component of a speech recognizer. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, ASSP-35(3):400–401, 1987.

D. Klakow, G. Rose, and X. Aubert. OOV-detection in large vocabulary system using automatically defined word-fragments as fillers. In *Eurospeech*, 1999.

R. Kneser and H. Ney. Improved backing-off for m-gram language modeling. *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 181–184, 1995.

K. Knight and Y. Al-Onaizan. Translation with finite-state devices. *Machine Translation and the Information Soup*, pages 421–437, 1998.

D. Koller and N. Friedman. *Probabilistic Graphical Models*. The MIT Press, 2009.

J. Lafferty, A. McCallum, and F. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *International Conference on Machine Learning (ICML)*, 2001.

J. R. Landis and G. G. Koch. The measurement of observer agreement for categorical data. *Biometrics*, 33:159–174, 1977.

G. Lidstone. Notes on the general case of the bayes-laplace formula for inductive or a posteriori probabilities. *Transactions of the Faculty of Actuaries*, 8:182–192, 1920.

H. Lin, J. Bilmes, D. Vergyri, and K. Kirchhoff. OOV detection by joint word/phone lattice alignment. In *ASRU*, pages 478–483, Dec. 2007.

Y. Liu, A. Stolcke, E. Shriberg, and M. Harper. Using conditional random fields for sentence boundary detection in speech. In *ACL*, 2005.

B. Logan, P. Moreno, J.-M. V. Thong, and E. Wittaker. An experimental study of an audio indexing system for the web. *ICSLP*, 1996.

J. Mamou, B. Ramabhadran, and O. Siohan. Vocabulary independent spoken term detection. In *SIGIR*, 2007.

L. Mangu, E. Brill, and A. Stolcke. Finding consensus among words. In *Eurospeech*, 1999.

A. Martin, G. Doddington, T. Kamm, M. Ordowski, and M. Przybocky. The DET curve in assessment of detection task performance. In *Eurospeech*, 1997.

A. McCallum. MALLET: A machine learning for language toolkit. `http://mallet.cs.umass.edu`, 2002.

A. McCallum and W. Li. Early results for named entity recognition with conditional random fields, feature induction and web-enhanced lexicons. In *CONLL*, 2003.

R. McDonald and F. Pereira. Identifying gene and protein mentions in text using conditional random fields. *BMC Bioinformatics*, 6(Suppl 1)(S6), 2005.

H. Meng, S. Seneff, and V. Zue. Phonological parsing for reversible letter-to-sound/sound-to-letter generation. *ICASSP*, pages II1–II4, 1994.

D. Miller, M. Kleber, C. lin Kao, and O. Kimball. Rapid and accurate spoken term detection. In *INTERSPEECH*, 2007.

M. Mohri. Finite-state transducers in language and speech processing. *Computational Linguistics*, 23(2), 1997. URL http://www.research.att.com/~mohri/cl1.ps.gz.

M. Mohri. Weighted automata algorithms. *Handbook of weighted automata*, pages 213–254, 2009.

M. Mohri, F. Pereira, and M. Riley. Weighted automata in text and speech processing. In *In ECAI-96 Workshop*, pages 46–50. John Wiley and Sons, 1996.

J. Morris and E. Fosler-Lussier. Discriminative phonetic recognition with conditional random fields. In *HLT-NAACL*, 2006.

J. Morris and E. Fosler-Lussier. Further experiments with detector based conditional random fields in phonetic recognition. In *ICASSP*, 2007.

H. Ney, U. Essen, and R. Kneser. On structuring probabilistic dependencies in stochastic language modeling. *Computer Speech and Language*, 8:1–38, 1994.

K. Ng. *Subword-based Approaches for Spoken Document Retrieval*. PhD thesis, Massachusetts Institute of Technology, 1990.

T. Ng, M. Ostendorf, M.-Y. Hwang, M. Siu, I. Bulyko, and X. Lei. Web-data augmented language model for Mandarin speech recognition. In *ICASSP*, 2005.

NIST. 2006 std website and evaluation plan. URL http://www.nist.gov/speech/tests/std/.

S. Novotney, R. Schwartz, and J. Ma. Unsupervised acoustic and language mdoel training with small amounts of labelled data. *ICASSP*, 2009.

S. Oger, V. Popescu, and G. Linares. Using the world wide web for learning new words in continuous speech recognition tasks: Two case studies. In *SPECOM*, 2009.

M. Ostendorf, V. Digalakis, and O. A. Kimball. From hmms to segment models: A unified view of stochastic modeling for speech recognition. *IEEE Transactions on Speech and Audio Processing*, 4(5), 1996.

C. Parada, M. Dredze, D. Filimonov, and F. Jelinek. Contextual information improves oov detection in speech. In *NAACL-HLT*, 2010a.

C. Parada, A. Sethy, M. Dredze, and F. Jelinek. A spoken term detection framework for recovering out-of-vocabulary words using the web. *INTERSPEECH*, 2010b.

C. Parada, A. Sethy, and B. Ramabhadran. Balancing false alarms and hits in spoken term detection. In *ICASSP*, 2010c.

C. Parada, M. Dredze, A. Sethy, and A. Rastrow. Learning sub-word units for open vocabulary speech recognition. *ACL*, 2011.

S. Parlak and M. Saraclar. Spoken term detection for turkish broadcast news. In *ICASSP*, 2008.

F. Pereira, N. Tishby, and L. Lee. Distributional clustering of english words. In *ACL*, 1993.

H. Poon, C. Cherry, and K. Toutanova. Unsupervised morphological segmentation with log-linear models. In *NAACL '09*, pages 209–217, 2009.

D. Povey, D. Kanevsky, B. Kingsbury, B. Ramabhadran, G. Saon, and K. Visweswariah. Boosted mmi for model and feature space discriminative training. *ICASSP*, 2008.

B. Ramabhadran, A. Sethy, J. Mamou, B. Kingsbury, and U. Chaudhari. Fast decoding for open vocabulary spoken term detection. In *North American Chapter of the Association for Computational Linguistics - Human Language Technologies (NAACL-HLT)*, June 2009.

A. Rastrow, A. Sethy, and B. Ramabhadran. A new method for OOV detection using hybrid word/fragment system. *ICASSP*, 2009a.

A. Rastrow, A. Sethy, B. Ramabhadran, and F. Jelinek. Towards using hybrid, word, and fragment units for vocabulary independent LVCSR systems. *INTERSPEECH*, 2009b.

J. Rissanen. Stochastic complexity in statistical inquiry theory. *World scientific Series in Computer Science*, 15, 1989.

R. Rosenfeld. Optimizing lexical and n-gram coverage via judicious use of linguistic data. *Eurospeech*, 1995.

T. I. Sadaoki Furui, Masanobu Nakamura and K. Iwano. Why is the recognition of spontaneous speech so hard? *International Conference on Text, Speech, and Dialogue*, pages 9–22, September 2005.

E. T. K. Sang and F. D. Meulder. Introduction to the conll-2003 shared task: Language-independent named entity recognition. In *CONLL*, 2003.

G. Saon, M. Padmanabhan, R. Gopinath, and S. Chen. Maximum likelihood discriminant feature spaces. *ICASSP*, 2000.

T. Schaaf. Detection of OOV words using generalized word models and a semantic class language model. In *Eurospeech*, 2001.

O. Scharenborg and S. Seneff. A two-pass strategy for handling oovs in a large vocabulary recognition task. *INTERSPEECH*, pages 1669–1672, 2005.

J. E. Shoup. *Phonological Aspects of Speech Recognition*. Number 125-138. Prentice-Hall, 1980.

N. A. Smith and J. Eisner. Contrastive estimation: training log-linear models on unlabeled data. In *ACL '05: Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 354–362, Morristown, NJ, USA, 2005. Association for Computational Linguistics. doi: http://dx.doi.org/10.3115/1219840.1219884.

H. Soltau, G. Saon, and B. Kingsbury. The ibm attila speech recognition toolkit. *IEEE Workshop on Spoken Language Technology*, 2010.

Stanley F. Chen. Conditional and joint models for grapheme-to-phoneme conversion. In *Eurospeech*, pages 2033–2036, 2003.

S. Stymne, M. Holmqvist, and L. Ahrenberg. Vs and oovs: two problems for translation between german and english. *Proceedings of the Joint Fifth Workshop on Statistical Machine Translation and MetricsMATR*, pages 183–188, 2010.

K. Sudoh, H. Tsukada, and H. Isozaki. Incorporating speech recognition confidence into discriminative named entity recognition of speech data. In *ACL*, 2006.

B. Suhm, M. Woszczyna, and A. Waibel. Detection and transcription of new words. *European Conference Speech Communication and Technology*, pages 2179–2182, 1993.

H. Sun, G. Zhang, f. Zheng, and M. Xu. Using word confidence measure for OOV words detection in a spontaneous spoken dialog system. In *Eurospeech*, 2001.

A. Viterbi. Error bounds for convolutional codes and an asymmetrically optimum decoding algorithm. *IEEE Transactions on Information Theory*, IT-13:260–267, 1967.

K. Wang. Language processing in the web era. PPT Presentation at CLSP, February 2011.

S. Wang. Using graphone models in automatic speech recognition. Master's thesis, Massachusetts Institute of Technology, 2009.

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), 2001.

C. White, J. Droppo, A. Acero, and J. Odell. Maximum entropy confidence estimation for speech recognition. In *ICASSP*, 2007.

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. *ICASSP*, 2008.

I. Witten and T. Bell. The zero-frequency problem: Estimation the probability of novel events in adaptive text compression. *IEEE Transactions on information theory*, 37(4): 1085–1094, 1991.

P. Woodland, S. Johnson, P. Jourlin, and K. S. Jones. Effects of out of vocabulary words in spoken document retrieval. *SIGIR*, 2000.

Y. G. Y. Li, H. Erdogan and E. Marcheret. Incremental online feature space mllr adapation for telephony speech recognition. *ICSLP*, pages 1417–1420, 2002.

A. Yazgan and M. Saraclar. Hybrid language models for out of vocabulary word detection in large vocabulary conversational speech recognition. *Proceedings of International Conference of Acoustics, Speech, and Signal Processing*, pages 745–748, 2004.

G. Zweig and P. Nguyen. A segmental crf approach to large vocabulary continuous speech recognition. In *ASRU*, pages 152–157, 2009.

Vita

Maria Carolina Parada received the B.S. and M.S. degrees in Electrical Engineering in 2004 and 2006 respectively at Washington State University. She enrolled in the Electrical Engineering PhD program at Johns Hopkins University in September 2006 and has been a member of the Center for Language and Speech Processing and the Human Language Technology Center of Excellence. During her PhD at Johns Hopkins University, she had the opportunity to work as a summer intern at the research groups in Nuance Communications, Google, and IBM working on Automatic Speech Recognition. She has been the recipient of various fellowships, including the Google PhD Fellowship in Speech 2010, Dean's fellowship at Johns Hopkins University and the International Merit Award at Washington State University.

In August 2011, Carolina will start as a Research Scientist at Google in Mountain View CA, working on automatic speech recognition.