

**RESOURCE-DEPENDENT ACOUSTIC AND LANGUAGE
MODELING FOR SPOKEN KEYWORD SEARCH**

A Dissertation
Presented to
The Academic Faculty

By

I-Fan Chen

In Partial Fulfillment
Of the Requirements for the Degree
Doctor of Philosophy in the
School of Electrical and Computer Engineering



Georgia Institute of Technology
May 2016

COPYRIGHT © 2016 BY I-FAN CHEN

**RESOURCE-DEPENDENT ACOUSTIC AND LANGUAGE
MODELING FOR SPOKEN KEYWORD SEARCH**

Approved by:

Dr. Chin-Hui Lee, Advisor
School of Electrical and Computer
Engineering
Georgia Institute of Technology

Dr. Biing-Hwang Juang
School of Electrical and Computer
Engineering
Georgia Institute of Technology

Dr. Mark Clements
School of Electrical and Computer
Engineering
Georgia Institute of Technology

Dr. Gee-Kung Chang
School of Electrical and Computer
Engineering
Georgia Institute of Technology

Dr. Yao Xie
H. Milton Stewart School of Industrial &
Systems Engineering
Georgia Institute of Technology

Date Approved: 11/17/2015

ACKNOWLEDGEMENTS

The completion of this dissertation involves assistance from many people. I would particularly like to thank my advisor, Prof. Chin-Hui Lee, for his guidance in my path to the PhD. It was a great pleasure to be mentored by his valuable direction, complete support and incomparable kindness. I am also grateful for the chance to be in an awesome lab with so many great colleagues to discuss research problems.

Most of the works in this dissertation are the research works I have accomplished in the Babel OpenKWS competitions. I want to show my greatest appreciation to Dr. Haizhou Li and Dr. Nancy Chen from I2R, who led the team very well and provided me the chance to participate the event. Dr. Chen has also given me many constructive comments about my dissertation. I would also like to offer my special thanks to Dr. Alvina Goh, who arranged my visit for Singapore in the OpenKWS competition period. I want to thank all the teammates who share their research experiences with me in the OpenKWS competitions as well. I really enjoyed working with them and learned a lot in the teamwork.

Finally yet importantly, I would like to thank my family. Without their support, I would never have this chance to acquire this PhD degree. My deepest appreciation goes to my wife, Dr. Ya-Shu Huang, who is not only a sweet and thoughtful spouse but also a knowledgeable and wise research partner in my life.

TABLE OF CONTENTS

	Page
ACKNOWLEDGEMENTS	iii
LIST OF TABLES	vii
LIST OF FIGURES	x
SUMMARY	xii
Chapter 1 Introduction	1
1.1 Introduction	1
1.1.1 Problem formulations of ASR and KWS	2
1.1.2 The two modeling problems in the <i>plug-in</i> MAP framework	4
1.1.3 Alleviation of the <i>plug-in</i> MAP modeling problems in ASR	5
1.1.4 Alleviation of the <i>plug-in</i> MAP modeling problems in KWS	6
1.2 Dissertation Organization	9
1.3 Dissertation Contributions	11
1.4 List of Publications	11
Chapter 2 Literature Survey	13
2.1 A Review of Spoken Keyword Search Techniques	13
2.1.1 Keyword-filler based KWS	15
2.1.2 LVCSR-based KWS	16
2.2 A Survey of Recent KWS Research	16
2.2.1 Speech feature representation	17
2.2.2 Rescoring and verification	17
2.2.3 Acoustic, language, and pronunciation modeling	18
2.3 Benchmark Datasets for KWS Research	21
2.3.1 TIMIT Corpus	21
2.3.2 IARPA Babel program and OpenKWS evaluations	22
2.4 Performance Measures for KWS Systems	22
2.4.1 The reference and KWS system outputs	23
2.4.2 Missed Detection Rate	23
2.4.3 False Alarm Rate	24
2.4.4 Actual Term Weighted Value	24
2.4.5 Figure of Merit	25
Chapter 3 Resource-Dependent Keyword Modeling	26
3.1 Introduction	26
3.2 The " <i>Share & Split</i> " training process for keyword modeling	26
3.3 Hierarchical model optimization with <i>split</i> process	27
3.4 DNN-based keyword verification	28
3.4.1 Deep neural network	30
3.4.2 Input features generation for NN models	30
3.4.3 Government phonology features	31

3.5	Experiments	32
3.5.1	Experimental setup	32
3.5.2	Parameter sharing analysis	33
3.5.3	Experimental results for hierarchical keyword modeling	34
3.5.4	Experimental results for keyword verification	36
3.6	Conclusion	38
Chapter 4	Keyword-Aware Language Modeling	40
4.1	Introduction	40
4.2	Spoken Keyword Search Problem	41
4.2.1	Keyword-filler based KWS	42
4.2.2	LVCSR-based KWS	43
4.3	The Prior Underestimation Problem for Multi-word Keywords	43
4.4	Keyword-Aware Language Modeling	45
4.5	Exact Realization of the Keyword-Aware Grammar	47
4.5.1	Preliminary	47
4.5.2	Representation of n -gram LMs with WFSAs	48
4.5.3	Realization of KW-aware grammar with WFSAs	49
4.6	Approximation of the Keyword-Aware Grammar using n -gram LM	49
4.6.1	Keyword-boosted language model	50
4.6.2	Keyword language model interpolation	51
4.6.3	Context-simulated keyword language models	52
4.7	Experiments	53
4.7.1	Experimental setup	53
4.7.2	LM training data analysis for the Vietnamese LLP task	54
4.7.3	Performance of approximate approaches	55
4.7.4	Comparison of KWLM and CS-KWLM interpolation	57
4.7.5	Comparison of the exact and approximate realizations	57
4.7.6	ATWV analysis for keywords of different lengths	60
4.7.7	ATWV analysis for IV and OOV keywords	62
4.7.8	ATWV analysis for seen and unseen keywords	63
4.7.9	Comparison of estimated keyword priors	64
4.7.10	Performance of keyword-aware language model to new keywords	65
4.7.11	OpenKWS13 Vietnamese full language pack task	67
4.8	Conclusion	68
Chapter 5	System Optimization Objectives	69
5.1	Introduction	69
5.2	Keyword-Aware Discriminative Acoustic Modeling	69
5.2.1	State-level minimum Bayes risk (sMBR) training objective	70
5.2.2	Training DNN-HMM hybrid systems with sMBR criterion	71
5.2.3	Keyword-boosted sMBR training objective	72
5.3	Discriminative Keyword-Aware Language Modeling	72
5.3.1	Global linear models	73
5.3.2	The perceptron algorithm	74
5.3.3	Issues in the conventional DLM approaches	76
5.4	Experiments	77

5.4.1	Experimental setup	77
5.4.2	Keyword-aware discriminative acoustic modeling	79
5.4.3	Discriminative keyword-aware language modeling	85
5.5	Conclusion	89
Chapter 6	Data Augmentation	91
6.1	Introduction	91
6.2	Multilingual Acoustic Modeling for Deep Neural Networks	91
6.2.1	Multilingual training for DNN acoustic models	93
6.2.2	Target language fine-tuning for DNN acoustic models	94
6.3	Web Text Augmented Language Modeling	96
6.4	Experiments	98
6.4.1	Experimental setup	98
6.4.2	Analysis of the importance of AM and LM in VLLP tasks	100
6.4.3	Multilingual acoustic modeling	101
6.4.4	Web text augmented language modeling	116
6.5	Conclusion	126
Chapter 7	Conclusion	128
7.1	Summary	128
7.2	Dissertation Contributions	130
7.3	Possible Extensions and Improvements	131
Appendix A		133
Appendix B		134
References		136

LIST OF TABLES

	Page
Table 1. Three research directions and six topics explored in the dissertation.....	8
Table 2. Keywords used in this study. Keywords are divided into three groups according to the count of training instances: 1 – no training instance, 2 – some training instances (<50), 3 – plenty of training instances (>=50).....	33
Table 3. Word- and phone-level parameter sharing factors in the TIMIT corpus.....	34
Table 4. Performance comparison of the baseline and the proposed KWS systems at different optimization levels. Numbers in gray are the performance using keyword models at the lower optimization level since the systems are lacking training data and cannot be optimized on the current level.	35
Table 5. Comparison of the KWS systems with different settings. $n=1$ means NN with a single hidden layer; $n \geq 1$ means NN with one or more than one hidden layers, while the number n is selected empirically based on the performance on the development set.	37
Table 6. Numbers of keywords unseen in the training data and keywords containing OOV words among the given list of 4,065 keywords in the Vietnamese LLP task.....	55
Table 7. WER and ATWV comparison of Vietnamese LLP systems with different language models on the Vietnamese <i>dev2h</i> data.	56
Table 8. WER and ATWV performance of Vietnamese LLP systems with different language models on the Vietnamese <i>evalpart1</i> data.	56
Table 9. Comparison of the three grammar WFSAs in terms of arc number, state number, and file size.	58
Table 10. Performance of the n -gram baseline and the two realizations of the proposed keyword-aware framework on the Vietnamese LLP <i>evalpart1</i> data.....	59
Table 11. Performance of the n -gram baseline and the two realizations of the proposed keyword-aware framework on the Tamil LLP <i>evalpart1</i> data.....	59
Table 12. ATWV performance of all, in-vocabulary (IV), and out-of-vocabulary (OOV) keywords for the baseline LM and CS-KWLM Interpolation systems on the <i>evalpart1</i> data.	63
Table 13. ATWV for seen and unseen keywords in the Vietnamese LLP task.....	64
Table 14. The four keyword lists available in the OpenKWS14 Tamil LLP task.	67
Table 15. ATWV performance of the n -gram baseline and the CS-KWLM interpolation system (enhanced with the Eval keyword list) for the four keyword lists on the Tamil <i>evalpart1</i> data.....	67
Table 16. ATWV performance of the n -gram baseline and CS-KWLM interpolation systems on <i>evalpart1</i> data in the Vietnamese FLP task.	68
Table 17. ATWV (4,065 keywords) of the Vietnamese LLP systems on the <i>evalpart1</i> test set with the three test configurations.	80
Table 18. WER (4,065 keywords) of the Vietnamese LLP systems on the <i>evalpart1</i> test set with three test configurations.	80
Table 19. The 1-bset sentence analysis (correction, substitution, deletion, and insertion) for the <i>orig Lex+LM</i> test configuration on the Vietnamese <i>dev2h</i> data with	

	4,065 keywords. ATWV for sMBR and KW-boosted sMBR systems are 0.2101 and 0.2564.	85
Table 20.	The 1-best sentence analysis for <i>G2P Lex+ CS-KWLM Int</i> test configuration on the Vietnamese <i>dev2h</i> data with 4,065 keywords. ATWV for sMBR and KW-boosted sMBR systems are 0.3546 and 0.3649 respectively.	85
Table 21.	Word Error Rate (WER) and Sentence Error Rate (SER) of DLM systems on the Vietnamese <i>dev2h</i> data set using conventional MAP decoding in the Vietnamese LLP task.....	87
Table 22.	Word Error Rate (WER) and Sentence Error Rate (SER) of DLM systems on the Vietnamese <i>evalpart1</i> data set using conventional MAP decoding in the Vietnamese LLP task.....	87
Table 23.	WER and ATWV of the baseline CS-KWLM interpolation and DLM systems on the Vietnamese <i>dev2h</i> data in the Vietnamese LLP task.	89
Table 24.	WER and ATWV of the baseline CS-KWLM interpolation and DLM systems on the Vietnamese <i>evalpart1</i> data in the Vietnamese LLP task.	89
Table 25.	Training data sets used in the experiments. The FLP training data can be used for the multilingual acoustic modeling. The Vietnamese LLP/VLLP and Tamil VLLP data were used for pure Vietnamese/Tamil system building and the <i>target language fine-tuning</i> stage in the multilingual acoustic modeling recipe. Note that we reserved a tenth of the task data as validation set when doing DNN training.....	99
Table 26.	ATWVs of KWS systems with different combinations of acoustic and language models on Vietnamese <i>dev2h</i> set.	100
Table 27.	WER of KWS systems with different combinations of acoustic and language models on Vietnamese <i>dev2h</i> set.....	101
Table 28.	Details of the training data utilized in each stage of the multilingual acoustic modeling recipe for the three tasks. The validation data were used by the iterative cross-entropy training process to determine whether to halve the learning rate in the next iteration or terminate the training process. Language name abbreviations used here are: C – Cantonese, P – Pashto, Tu – Turkish, Tg – Tagalog, V – Vietnamese, Tm – Tamil.....	103
Table 29.	The ATWVs of the monolingual and multilingual systems in the Vietnamese LLP and VLLP tasks on the <i>evalpart1</i> data. Multilingual acoustic modeling provided a significant performance improvement in the VLLP task.	109
Table 30.	Details of the six languages data used in the experiment. Except for Vietnamese, which is the target language, the amount of training data for each of the 5 assisting languages was set to 56 hours for fair comparison. Information including the number of tones, the spoken area, and the language family of each language is also listed.	112
Table 31.	Number of overlapped phones for the 5 assisting languages with Vietnamese. Both tonal and non-tonal (by removing tone tags) Vietnamese phone sets are considered, where tonal Vietnamese phone set has 236 phones and non-tonal phone set has 41 phones. The one overlapped phone among Vietnamese tonal phone set with Pashto, Turkish, Tagalog, and Tamil is the phone /m/. .	112
Table 32.	Acoustic distances between pairs of the training languages. The acoustic distance is measured by evaluating the KL2 distances between the acoustic	

	feature distributions of the languages. Both V_3hr and V_78 are for Vietnamese language. V_3hr used the 3-hour Vietnamese VLLP data to build the Vietnamese GMM for KL2 distance evaluation, while V_78hr used all the Vietnamese FLP data to build the language GMM.....	113
Table 33.	ATWV performance of Vietnamese VLLP systems with multilingual AM and KW-aware LM on evalaprt1.....	116
Table 34.	Acoustic model configurations of the Vietnamese LLP and VLLP systems in the experiment.	116
Table 35.	Details of the two in-domain and six web-text sources used in the experiments.	117
Table 36.	Evaluation keyword coverage rates of the 8 training text sources used in the experiment.	118
Table 37.	Language model perplexities on keyword list and evaluation sentences for the LMs used in the Vietnamese LLP experiments.....	119
Table 38.	Language model perplexities on keyword list and evaluation sentences for the LMs used in the Vietnamese VLLP experiments.....	120
Table 39.	The ATWVs of the web text augmented LM systems on the <i>evalpart1</i> data in the Vietnamese LLP task.....	121
Table 40.	The WERs of web text augmented LM systems on <i>evalpart1</i> data in the Vietnamese LLP task. (LMWT=12)	121
Table 41.	The ATWVs of web text augmented LM systems on <i>evalpart1</i> data in the Vietnamese VLLP task.....	122
Table 42.	The WERs of web text augmented LM systems on <i>evalpart1</i> data in the Vietnamese VLLP task.....	122
Table 43.	The WERs and ATWVs of the Vietnamese LLP systems with different lexicon and language model configurations on <i>evalpart1</i> data.....	125
Table 44.	The WERs and ATWVs of the Vietnamese VLLP systems with different lexicon and language model configurations on <i>evalpart1</i> data.....	126

LIST OF FIGURES

	Page
Figure 1. Information theoretic view of human speech generation	2
Figure 2. The modern KWS framework.	14
Figure 3. Illustrations of grammars (or language models) in (a) keyword-filler based KWS and (b) LVCSR-based KWS	16
Figure 4. Illustration of the KWS framework with an additional keyword verification stage for keywords with plenty of training samples.	29
Figure 5. ROC curves of the Group 3 keywords for the phone- + word-level optimized keyword hypothesizers and the DNN-based keyword verifiers (rescore the putative hits generated by the hypothesizers).	38
Figure 6. An example of the " <i>incorrect independent event assumption</i> " made by an n -gram LM ($n=2$ in this example) with backoff smoothing when the multi-word keyword is unseen in the LM training data. The incorrect assumption causes serious prior underestimation for multi-word keywords.	44
Figure 7. Keyword probabilities counted directly from the transcriptions of the Babel Vietnamese <i>evalpart1</i> evaluation data (ground-truth) and the probabilities estimated by the n -gram LM ($n=3$) in the Vietnamese LLP task. The n -gram LM seriously underestimated the probabilities of multi-word keywords.	45
Figure 8. Illustrations of (a) the grammar of classic keyword-filler based KWS, (b) the n -gram LM based grammar used by LVCSR-based KWS, and (c) the proposed keyword-aware grammar, which combines the grammars used in the two KWS frameworks.....	47
Figure 9. Pseudo code for the KW-aware grammar WFSM realization	49
Figure 10. An illustration of the keyword-boosted LM training data.....	50
Figure 11. The construction of keyword language model interpolated LM. The final LM is an interpolated LM of the original LM and the keyword LM.....	52
Figure 12. An illustration of the training text for context-simulated keyword language models (CS-KWLM). Note that instead of using real context terms for a keyword, simulated context terms are used here because in most case context information of the keyword is not available (especially in the domain mismatch conditions). The simulated context terms in this research are selected as top M bigrams in the original LM training data. The value of M is chosen to allow the text file size stay in a manageable scale (e.g., less than 2 Gb).	53
Figure 13. ATWV on <i>dev2h</i> with different keyword LM weights α for both KWLM and CS-KWLM interpolation methods in the Vietnamese LLP task.	57
Figure 14. ATWV of keywords with different lengths for the three systems on <i>evalpart1</i> data in the Tamil LLP task. The ATWV drops at $L=5$ in all the systems are due to miss errors caused by underestimated keyword priors. Also, since there are only 3 keywords at $L=5$, the ATWV might not be representative for all the five-word keywords.	61
Figure 15. ATWV for keywords with different length, L , of the baseline and KW-aware systems on the <i>evalpart1</i> data in the Vietnamese LLP task.	62

Figure 16. Prior estimations of keywords with different lengths on the Vietnamese <i>evalpart1</i> data.....	65
Figure 17. A general form of the perceptron algorithm. The value M , which is the number of iteration over the training data, is chosen by the validation set.	75
Figure 18. The perceptron algorithm implemented with WFSA. L_i is the WFSA representation of the i^{th} lattice, and D is the n -gram LM WFSA representation of the linear model parameter \mathbf{w}	76
Figure 19. ATWV (4,065 keywords) of Vietnamese KW-boosted sMBR systems on <i>dev2h</i> with different keyword-boosting weight values of β	81
Figure 20. WER (4,065 keywords) of Vietnamese KW-boosted sMBR systems on <i>dev2h</i> with different keyword-boosting weight values of β	82
Figure 21. ATWV (200 keywords) of Vietnamese KW-boosted sMBR systems on <i>dev2h</i> with different keyword-boosting weight β	83
Figure 22. WER (200 keywords) of Vietnamese KW-boosted sMBR systems on <i>dev2h</i> with different keyword-boosting weight β	83
Figure 23. An illustration of the shared-hidden-layer multilingual DNN (SHL-MDNN) proposed in the reference [19] (the figure is adopted from the paper). In this dissertation, the SHL-MDNN framework is used in the multilingual acoustic modeling recipe.....	92
Figure 24. An illustration of the multilingual acoustic modeling recipe for Vietnamese DNN AMs. The target language (Vietnamese) fine-tuning is followed after the multilingual training stage. The initial learning rates for multilingual training and target language fine-tuning are empirically set to 0.008 and 0.002, respectively.	95
Figure 25. ATWV of multilingual acoustic modeling recipes with different number of fine-tuning layers on <i>dev2h</i> data for the Vietnamese LLP task.	105
Figure 26. WER of multilingual acoustic modeling recipes with different number of fine-tuning layers on <i>dev2h</i> data for the Vietnamese LLP task.	105
Figure 27. Vietnamese VLLP Systems' ATWV at different training stages in multilingual acoustic modeling recipes. Tested on the Vietnamese <i>dev2h</i> data.	107
Figure 28. Vietnamese VLLP Systems' WER at different training stages in multilingual acoustic modeling recipes. Tested on the Vietnamese <i>dev2h</i> data.	107
Figure 29. Tamil VLLP Systems' ATWV at different training stages in multilingual acoustic modeling recipes. Tested on Tamil <i>dev2h</i> data.....	108
Figure 30. Tamil VLLP Systems' WER at different training stages in multilingual acoustic modeling recipes. Tested on Tamil <i>dev2h</i> data.....	108
Figure 31. WER of the 5 bilingual trained Vietnamese VLLP systems at each training stage on the Vietnamese <i>dev2h</i> set. Tagalog and Tamil assisted systems are the two systems with the best WER performance.	114
Figure 32. The ATWVs of the 5 bilingual trained Vietnamese VLLP systems at each training stage on the Vietnamese <i>dev2h</i> set. Tagalog and Tamil assisted systems are the two systems with the best ATWV performance.....	115

SUMMARY

In this dissertation, three research directions were explored to alleviate two major issues, i.e., the use of incorrect models and training/test condition mismatches, in the modeling frameworks of modern spoken keyword search (KWS) systems. Each of the three research directions, which include (i) data-efficient training processes, (ii) system optimization objectives, and (iii) data augmentation, utilizes different types and amounts of training resources in different ways to ameliorate the two issues of acoustic and language modeling in modern KWS systems. To be more specific, *resource-dependent keyword modeling*, *keyword-boosted sMBR (state-level minimum Bayes risk) training*, and *multilingual acoustic modeling* are proposed and investigated for acoustic modeling in this research. For language modeling, *keyword-aware language modeling*, *discriminative keyword-aware language modeling*, and *web text augmented language modeling* are presented and discussed.

The dissertation provides a comprehensive collection of solutions and strategies to the acoustic and language modeling problems in KWS. It also offers insights into the realization of good-performance KWS systems. Experimental results show that the data-efficient training process and data augmentation are the two directions providing the most prominent performance improvement for KWS systems. While modifying system optimization objectives provides smaller yet consistent performance enhancement in KWS systems with different configurations. The effects of the proposed acoustic and language modeling approaches in the three directions are also shown to be additive and can be combined to further improve the overall KWS system performance.

CHAPTER 1 INTRODUCTION

1.1 Introduction

Spoken keyword search (KWS) is a task to detect a set of keywords in continuous speech. It could be considered as an application of automatic speech recognition (ASR) focusing only on the recognition of the keywords¹. Modern ASR-based KWS and ASR technologies are developed based on the information theoretic view of human speech generation, acquisition, transmission, and perception of speech. Figure 1 shows a conceptual model of speech generation [1]. The speech generation process starts from a message source, where a message M consisting of a set of semantic units and concepts is generated. The message M is realized as a sequence of words W through a *linguistic channel*. Sometimes different word sequences will convey the same message. A following *articulatory channel* converts the discrete word sequence into a continuous speech signal S . Speaker effect, which accounts for a major part of the speech variability including accent, dialect, speaker rate, etc., is added at this stage. Additional speech distortion is introduced when the signal passes through the *transmission channel* (which includes speaking environment, interfering noise, transducers and transmitters used to capture and convey the speech signals) before it reaches the receiver as an observed signal O .

For speech recognition, we are interested in recovering and "recognizing" the word sequence W from the given signal O . For spoken keyword search, which is the focus of this dissertation, the goal is similar to speech recognition. However, instead of recognizing the whole word sequence W , we are only interested in recognizing and detecting partial word sequences, namely keywords, from the given signal O .

¹ Note that this perspective of the KWS problem is from ASR-based KWS, which is one of the popular KWS frameworks today. This dissertation adopts the viewpoint of ASR-based KWS for the KWS problem. However, readers should be aware of the fact that there are still other perspectives, which come with different strategies, e.g., detection-based approaches and query-by-example, for the KWS problem.

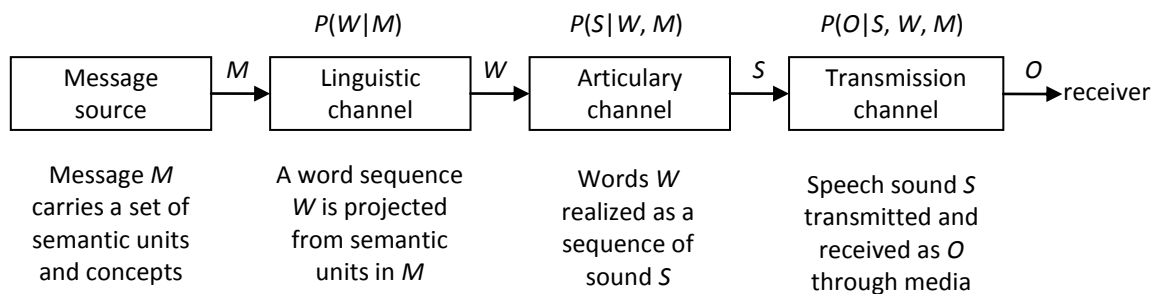


Figure 1. Information theoretic view of human speech generation

1.1.1 Problem formulations of ASR and KWS

The variability introduced in the channels of the speech generation process makes the statistical inference a natural choice for formulating and solving the ASR and ASR-based KWS problems. Since the both problems share a similar formulation, we will start with the formulation of ASR problem and explain the KWS formulation later. In general, given an observed speech signal O , the ASR problem can be solved by maximum *a posteriori* (MAP) decoding:

$$W^* = \arg \max_w P(W | O),$$

where W^* is the recognized sentence. Note that the ideal posterior probability distribution of $P(W|O)$ here is expected to have the following property:

Suppose W_o is the true word sequence of the observed speech signal O , and $L(W, W_o)$ is a loss function² evaluating the loss of a hypothesized word sequence W w.r.t W_o ; for hypothesized word sequences W_1 and W_2 , if $L(W_1, W_o) \leq L(W_2, W_o)$ then $P(W_1|O) \geq P(W_2|O)$.

Namely, $P(W|O)$ is a posterior distribution that correlates with the loss function in the test condition³.

² For example, zero-one lose function.

³ However, in reality, it is very difficult for us to acquire the ideal posterior probability distribution of $P(W|O)$ in the test condition. In most of the situations, we can only approximate the distribution using some parametric distributions trained with a limited amount of training data.

With Bayes' rule, the equation of MAP decoding can be written as

$$\begin{aligned}
 W^* &= \arg \max_w \frac{p(O|W)P(W)}{\sum_{W'} p(O|W')P(W')}, \\
 &\cong \arg \max_w p_{\Lambda^*}(O|W)P_{\Gamma^*}(W),
 \end{aligned} \tag{1}$$

which forms a well-known *plug-in* MAP decoding⁴. In Eq. (1), the posterior probability $P(W|O)$ is decomposed into two components, $p(O|W)$ and $P(W)$, known as an acoustic model and a language model, respectively. The former is the probability of the input signal O given the hypothesized word sequence W , and the latter is the language probability of W . Further, it is assumed that the distributions of $p(O|W)$ and $P(W)$ are parametric probability distributions, i.e., $p_{\Lambda}(O|W)$ and $P_{\Gamma}(W)$, respectively. The parameters Λ and Γ can be estimated from some training data. It is expected that the training data estimated distributions, $p_{\Lambda^*}(O|W)$ and $P_{\Gamma^*}(W)$, can be used to represent the true/ideal distributions of $p(O|W)$ and $P(W)$ in the test condition.

ASR-based KWS shares the same decoding process and the modeling framework in ASR. For a keyword q , Eq. (1) is used for the generation of hypothesized positions of the keyword q in the speech signal O . However, to determine if the keyword q exist in the input speech, a *posterior* probability of the keyword q given O , $P(q|O)$, is needed. The posterior probability, $P(q|O)$, is then used with a preset threshold for the final decision making. Similar to *plug-in* MAP decoding, the posterior probability can be estimated using acoustic and language models:

$$\begin{aligned}
 P(q|O) &= \sum_{W, q \in W} P(W|O) \\
 &\cong \sum_{W, q \in W} \frac{p_{\Lambda^*}(O|W)P_{\Gamma^*}(W)}{\sum_{W'} p_{\Lambda^*}(O|W')P_{\Gamma^*}(W')} .
 \end{aligned} \tag{2}$$

⁴ "*plug-in*" refers to the use of distributions estimated from training data in place of the real test data distributions in the decoding process. Most of the ASR systems today adopt this approach without specifically mentioning it.

In state-of-the-art ASR and KWS systems, the most successful acoustic modeling approach is to use a set of hidden Markov models (HMMs) as the acoustic models of subword or whole-word units [2]. Acoustic models including conventional Gaussian mixture model (GMM) based HMMs [3] and deep neural network (DNN) based HMMs [4] are popular in ASR and KWS systems today. For language modeling, the statistical n -gram models for words and/or word-classes have been proved effective and are widely used in the past decades [5]. Parameters of the above-mentioned models can be estimated using the maximum likelihood (ML) estimation.

1.1.2 The two modeling problems in the *plug-in* MAP framework

Note that in Eq. (1), there are two important assumptions made by the *plug-in* MAP decoding. The first assumption is that the distributions of $p(O|W)$ and $P(W)$ (and thus $P(W|O)$, implicitly) can be represented by parametric probability distributions $p_{\Lambda}(O|W)$ and $P_{\Gamma}(W)$. It is clear that the ML trained HMM-based AMs and n -gram LMs in ASR/KWS systems today do not satisfy the assumption. The distributions of these models do not correlate highly with the loss functions of ASR/KWS in test conditions⁵.

The second assumption is that the probability distributions of $p(O|W)$ and $P(W)$ on test data can be approximated by the parametric probability distributions $p_{\Lambda^*}(O|W)$ and $P_{\Gamma^*}(W)$ estimated from training data. From the theoretic point of view, the estimated $p_{\Lambda^*}(O|W)$ and $P_{\Gamma^*}(W)$ would asymptotically approach the true distribution in the test condition when:

- 1) the training data is in the same domain of test data and is representative enough with respect to the true distribution of test data, and
- 2) there is sufficient amount of training data available.

Unfortunately, training data collection is one of the most time-consuming and expensive efforts in the system-building process. In many cases, there would be only a limited amount of training data available for parameter estimation for an ASR or a KWS task in a

⁵ See Appendix A for more details.

specific domain. For ASR and KWS systems, it is therefore inevitable to have mismatches between training and test conditions and failed to satisfy this assumption made in the *plug-in* MAP decoding framework.

The fact that modern KWS and ASR systems barely satisfy the two assumptions in the *plug-in* MAP framework raises two modeling problems in KWS and ASR: i) the incorrect model problem, and ii) the mismatch between training and test conditions problem. These two problems are the two major causes of the performance degradation in modern ASR-based KWS and ASR systems, which makes the alleviation of these two problems become an important research area for acoustic and language modeling in ASR and ASR-based KWS.

1.1.3 Alleviation of the *plug-in* MAP modeling problems in ASR

In ASR, the incorrect model problem could be alleviated by adopting discriminative training objectives, which correlate with loss functions in ASR (i.e., word or sentence error rates) better than maximum likelihood, for parameter estimation. For acoustic modeling, many training objectives have been proposed for the purpose, e.g., MCE (Minimum Classification Error [6]), MMI (Maximum Mutual Information [7, 8]), bMMI (boosted Maximum Mutual Information [9]), MPE (Minimum Phone Error [8, 10]), MWE (Minimum Word Error [8]), sMBR (state-level Minimum Bayes Risk [11]). There are also various discriminative language modeling (DLM) [12-16] approaches proposed for ASR tasks.

For the mismatch between training and test conditions, two major directions: i) avoid model over-fitting, and ii) collect information about possible distributions in test condition, are adopted for tackling the problem.

When the amount of in-domain training data is too small to represent the test distributions $p(O|W)$ and $P(W)$ and/or the number of parameters in the parametric distributions $p_{\Lambda}(O|W)$ and $P_{\Gamma}(W)$ is too large with respect to the amount of training data, the ML estimated distributions $p_{\Lambda^*}(O|W)$ and $P_{\Gamma^*}(W)$ would tend to over-fit the training data and deviate from the true distribution of the test data. In this situation, the mismatch between the ML estimated and the true distributions could severely degrade the performance of *plug-in* MAP decoding. Parameter sharing, which reduces the number of

parameters in the parametric distributions $p_{\Lambda}(O|W)$ and $P_{\Gamma}(W)$, is a common approach in ASR to attenuate the over-fitting. For acoustic modeling, this can be accomplished by the use acoustic models of subword units (e.g., syllable, phone) instead of whole-word units to reduce the number parameters needed to be estimated in the system [2]. Tying HMM states using decision tree clustering is another popular and effective technique to reduce the number of parameters in acoustic models [17]. For language modeling, smoothing techniques, such as backoff smoothing, are prevalent methods for reducing the number of LM parameters in ASR [5].

Data augmentation is another strategy for dealing with the mismatch problem in ASR. The strategy works by utilizing training data not directly in the test domain, i.e., out-of-domain data, to assist the estimations of parameters Λ and Γ . The out-of-domain data include artificially generated training data (e.g., machine transcribed speech data in semi-supervised training [18]) and training data from other tasks (e.g., speech corpora of foreign languages [19], text data retrieved from the Internet [20, 21]). When out-of-domain data is available, domain-independent models can be first estimated using both the in-domain and out-of-domain data. The domain-independent models are then adapted with the limited in-domain data for better estimation of the distributions $p_{\Lambda}(O|W)$ and $P_{\Gamma}(W)$ in the target task domain.

1.1.4 Alleviation of the *plug-in* MAP modeling problems in KWS

Since ASR-based KWS shares most of the problem formulations with ASR, the solutions in ASR systems mentioned in section 1.1.3 can be directly applied to KWS systems. However, in KWS systems, we know that:

1. for a keyword q , KWS requires $P(q|O)$ to determine the existence of the keyword;
2. sometimes, information of target keywords is available at the model training stage;
3. the number of target keywords is usually much smaller than the vocabulary size.

The first property of KWS provides an additional way for us to tackle the modeling problems in KWS. The second one shows that, in addition to the in-domain and the out-of-domain training data, keyword information might be an important resource that can be utilized in the modeling process. The limited number of target keywords also makes

keyword-specific model optimization practical. These facts make KWS become a speech problem with even more resources than ASR to tackle the two modeling problems.

For the alleviation of the incorrect model problem, when keyword information is available, KWS-performance-related objective functions can be used for model training. Chapter 5 will show that KWS systems with KWS-oriented-objectives trained AMs consistently outperform KWS systems with ML and ASR-oriented-objectives trained AMs. Further, for a keyword q , the incorrect model problem can also be reduced by directly modeling the posterior probability, $P(q|O)$, without the use of Eq. (2). Chapter 3 will show that KWS performance could be significantly improved by using the keyword-specific DNN models for the estimation of $P(q|O)$.

For dealing with the mismatch between training and test conditions, it is apparent that KWS has much more choices of strategies. When keyword information is available, a "*share & split*" training process proposed in Chapter 3 can be used to reduce the mismatch between keyword models and their real pronunciations in the test condition. For language modeling, keyword information is an important knowledge to diminish the domain mismatch between the training data and the test keywords. Chapter 4 will show that keyword information is very helpful for n -gram language modeling in KWS systems, especially for the estimation of unseen keyword prior probabilities. When keyword information is not available, data augmentation can still be used for alleviating the training/test condition mismatch. Chapter 6 will show that KWS systems also benefit from the utilization of out-of-domain data. More importantly, the effects of data augmentation and the keyword-aware approaches are additive and can be combined to further enhance the system performance if both resources are available to a KWS system.

In summary, depending on the availability of keyword information, in-domain and out-of-domain training data, KWS systems can utilize different techniques mentioned above to tackle the two modeling problems in the *plug-in* MAP framework for system AM and LM estimations. This motivates the study of resource-dependent acoustic and language modeling in KWS in this dissertation. Three research directions for the enhancement of acoustic and language modeling in KWS systems are therefore emerged:

Data-Efficient Training Processes – study more efficient and effective ways to use keyword information and limited in-domain data for model-parameter estimations. For acoustic models (AM), data-efficient keyword modeling is proposed and realized by a "*share & split*" training process. For language models (LM), keyword-aware language modeling, which utilizes information of keywords to assist the modeling process, is proposed to enhance LMs in KWS tasks.

System Optimization Objectives – improve performance of KWS systems by replacing conventional maximum likelihood (ML), cross-entropy (CE), and ASR-performance related objectives in ASR systems with KWS-performance-related objective functions for system model training.

Data Augmentation – adopt the data augmentation approaches in ASR, which utilize foreign or out-of-domain data to overcome the mismatch between training and test conditions, for KWS tasks. Research topics, including multilingual acoustic modeling and web text assisted language modeling, are studied.

Table 1 shows the research map of this dissertation.

Table 1. Three research directions and six topics explored in the dissertation.

	Optimization with Keyword Information		Enhancement from Out-of-Domain Data
	Data-Efficient Training Processes	C5. System Optimization Objectives	C6. Data Augmentation
Acoustic Modeling	C3. Resource-Dependent Keyword Modeling	5.2. Keyword-boosted sMBR Training	6.2. Multilingual Acoustic Modeling
Language Modeling	C4. Keyword-Aware Language Modeling	5.3. Discriminative Keyword-Aware Language Modeling	6.3. Web Text Augmented Language Modeling

The dissertation provides a comprehensive collection of solutions to the acoustic and language modeling enhancement in ASR-based KWS. It also offers insights into the realization of good-performance KWS systems and therefore can be a guide for KWS designers to develop training recipes for KWS systems with different available training resources. Experimental results show that the proposed research is promising and significantly enhance the KWS performance.

1.2 Dissertation Organization

An overview of the organization of the dissertation is given below:

Chapter 2 – Literature Survey reviews the spoken keyword search system frameworks and recent KWS research. Development and test databases used in this dissertation and common KWS performance evaluation measures are also explained.

Chapter 3 Resource-Dependent Keyword Modeling proposes a "*share & split*" acoustic modeling procedure for GMM-HMM (Gaussian Mixture Model – Hidden Markov Model) based keyword models. By breaking the state tying and subword level sharing in conventional GMM-HMM acoustic models used by KWS systems, the new training procedure allows the keyword models to be optimized toward their own pronunciations. The optimization level of each keyword model depends on the amount of the training samples of the keyword. Therefore, the proposed approach provides an effective way utilizing the system training data to achieve the best system performance for each keyword. For keywords with plenty of training samples, NN (neural network) based keyword verifiers are also proposed for further KWS accuracy enhancement.

Chapter 4 – Keyword-Aware Language Modeling proposes a language modeling framework for LVCSR (Large Vocabulary Continuous Speech Recognition)-based KWS systems. A "keyword prior underestimation" problem for multi-word keywords caused by incorrect assumptions made in conventional n -gram LMs is discovered and thoroughly studied. The problem leads an LVCSR-based KWS system to high missed detection rate for multi-word keywords. To alleviate the

problem, a set of modeling approaches exploiting the keyword information is proposed. Experimental results show the prior underestimation for multi-word keywords is a problem across languages for n -gram LMs. By alleviating the problem, the proposed framework significantly improve the performance of LVCSR-based KWS.

Chapter 5 – System Optimization Objectives investigates acoustic and language model enhancement for KWS by using better training objectives. For acoustic modeling, Keyword-boosted sMBR training (section 5.2) is proposed for deep neural network (DNN) AMs in LVCSR-based KWS systems. By modifying the conventional sMBR training objective function to emphasize the keyword recognition errors, significant KWS performance improvement is observed. For language modeling, discriminative keyword-aware language modeling (section 5.3) is proposed. The chapter studies the effect of using discriminative language modeling (DLM) technique, which is commonly used in ASR tasks, in the keyword-aware language modeling framework. Difficulties and challenges for applying the DLM approaches to KWS tasks are also discussed.

Chapter 6 – Data augmentation studies the two most common and effective techniques utilizing out-of-domain data for acoustic and language modeling in ASR and KWS research. The combination of the data augmentation techniques with the enhancement approaches proposed in this dissertation is also studied. For acoustic modeling, a refined recipe for multilingual DNN acoustic modeling [19] is proposed in section 6.2. For language modeling, web text augmented language modeling is explored in section 6.3. Data filtering and selection for out-of-domain data are also studied.

Chapter 7 – Conclusion presents the summary and conclusions of the work. A discussion of future research directions is also given in the chapter.

1.3 Dissertation Contributions

The dissertation has generated a number of novel contributions to the field of KWS. These are:

1. A comprehensive blueprint of the enhancement of acoustic and language models in ASR-based KWS systems: three enhancement directions and their combination have been investigated. It can be a guide for future designers to build high performance system acoustic and language models in KWS systems.
2. A thorough investigation of the "keyword prior underestimation" problem for n -gram LMs in LVCSR-based KWS systems: the dissertation shows that the underestimation problem severely degrades detection performance of LVCSR-based KWS systems for multi-word keywords. It also identifies the causes of the problem and provides methods to avoid the problem.
3. The development of a novel "*share & split*" procedure for keyword acoustic modeling: the new training procedure allows keyword AMs to be optimized toward their own pronunciation according to the amount of their training data.
4. The development of the novel keyword-boosted sMBR acoustic modeling process and the study of discriminative keyword-aware language modeling for KWS systems: the new discriminative acoustic modeling process provides significant performance improvement for KWS tasks. The issues of applying discriminative language modeling to KWS systems are also discovered.
5. A novel multilingual acoustic modeling recipe for DNN AMs: detailed studies of combining the proposed keyword-aware language modeling approaches with popular used data augmentation modeling methods are provided.

1.4 List of Publications

- I.-F. Chen and C.-H. Lee, "A Study on Using Word-Level HMMs to Improve ASR Performance over State-of-the-Art Phone-Level Acoustic Modeling for LVCSR," in *Proc. Interspeech*, 2012.
- I.-F. Chen and C.-H. Lee, "A Resource-Dependent Approach to Word Modeling for Keyword Spotting," in *Proc. Interspeech*, 2013.

- I-F. Chen and C.-H. Lee, "A Hybrid HMM/DNN Approach to Keyword Spotting of Short," in *Proc. Interspeech*, 2013.
- I-Fan Chen, Chongjia Ni, Boon Pang Lim, Nancy F. Chen, and Chin-Hui Lee, "A Novel Keyword+LVCSR-Filler Based Grammar Network Representation for Spoken Keyword Search," in *Proc. ISCSLP*, 2014.
- I-Fan Chen, Nancy F. Chen, and Chin-Hui Lee, "A Keyword-Boosted sMBR Criterion to Enhance Keyword Search Performance in Deep Neural Network Based Acoustic Modeling," in *Proc. Interspeech*, 2014.
- Van Tung Pham, Nancy F. Chen, Sunil Sivadas, Haihua Xu, I-Fan Chen, Chongjia Ni, Eng Siong Chng, Haizhou Li, "System and Keyword Dependent Fusion for Spoken Term Detection," in *Proc. IEEE SLT*, 2014.
- Nancy Chen, Chongjia Ni, I-Fan Chen, Sunil Sivadas, et al., "Low-Resource Keyword Search Strategies for Tamil," in *Proc. ICASSP*, 2015.
- I-Fan Chen, Chongjia Ni, Boon Pang Lim, Nancy F. Chen, and Chin-Hui Lee, "A Keyword-Aware Grammar Framework for LVCSR-based Spoken Keyword Search," in *Proc. ICASSP*, 2015.
- I-Fan Chen, Chongjia Ni, Boon Pang Lim, Nancy F. Chen, and Chin-Hui Lee, "A Keyword-Aware Language Modeling to Spoken Keyword Search," in *Journal of Signal Processing Systems*, 2015, DOI: 10.1007/s11265-015-0998-0
- Tze Siong Lau, I-Fan Chen, Chin-Hui Lee, "Tunable Keyword-Aware Language Modeling and Context Dependent Fillers for LVCSR-based Spoken Keyword Search," in *Proc. Interspeech*, 2015.

CHAPTER 2 LITERATURE SURVEY

2.1 A Review of Spoken Keyword Search Techniques

Spoken keyword search (KWS) is a task to detect a set of keywords in continuous speech. It could be considered as an application of automatic speech recognition (ASR) focusing only on the recognition of the keywords⁶. The technology has been widely used in various applications such as spoken term detection [22-25], spoken document indexing and retrieval [26], speech surveillance [27], spoken message understanding [28, 29], and dialog systems [30]. As mentioned in section 1.1, the ASR-based KWS shares a similar formulation with ASR:

Given a speech utterance O and a text-based query q , an ASR-based KWS system detects the query, q , in the utterance, O , by acquiring the best *term* (e.g., word, or keyword/filler) sequence, W^* , representing the utterance. When maximum *a posteriori* (MAP) based decoding is used, the best term sequence acquiring process can be formulated as

$$W^* = \arg \max_w P(W | O). \quad (3)$$

If q exists in W^* , the KWS system reports the occurrence of the query with its location in the utterance, O , and a confidence score for the decision. Usually the confidence score is the posterior probability of q given O , which can be derived from Eq. (2), or the score derived from hypothesis testing.

Note with Bayes' rule, Eq. (3) can be written as

$$W^* = \arg \max_w p(O | W)P(W), \quad (4)$$

⁶ Note that this perspective of the KWS problem is from ASR-based KWS, which is one of the popular KWS frameworks today. This dissertation adopts the viewpoint of ASR-based KWS for the KWS problem. However, readers should be aware of the fact that there are still other perspectives, which come with different strategies, e.g., detection-based approaches and query-by-example, for the KWS tasks.

where $p(O|W)$ is the probability of utterance, O , given the hypothesized *term* sequence, W , and $P(W)$ is the prior probability of the sequence. Usually, the probability of $p(O|W)$ can be computed with acoustic models (AM) [2, 4] (Figure 2 (b)), and $P(W)$ is modeled by system language models (LM) [31, 32] (Figure 2 (c)). The *term* sequence W^* in Eq. (4) can then be searched with Viterbi algorithm. To alleviate the computational burden caused by the large search space, beam search techniques [33, 34] are often used for the search of W^* . Therefore, if a spoken keyword of q is in the utterance O , it is important for the KWS system to preserve the term sequence of q inside the search beam width during decoding. Otherwise, the system would have no chance to detect the keyword.

Note that the "*terms*" in KWS are not necessary "*words*" as in ASR. Depending on the type of a KWS system, the *terms* can be defined in different ways. Considering the decoding grammars used by KWS systems, two popular ASR-based KWS frameworks today are⁷: (i) classic keyword-filler based [33, 35], and (ii) large vocabulary continuous speech recognition (LVCSR) based KWS [22-25] systems.

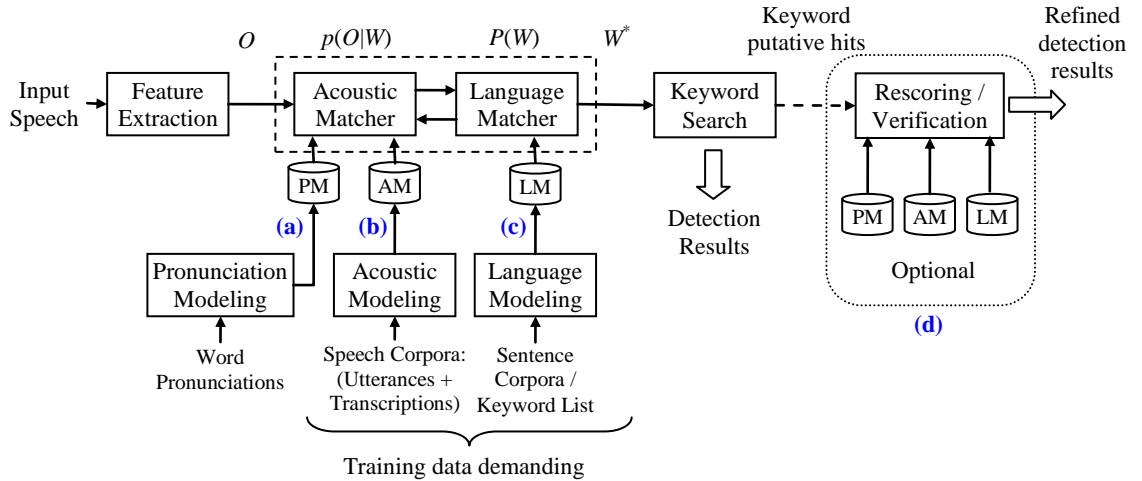


Figure 2. The modern KWS framework.

⁷ These are the two of the popular groups of ASR-based KWS systems today (categorized with the decoding grammars used by the systems). However, Chapter 4 will show another framework that bridges these two KWS frameworks. Further, readers should be aware of the fact that this is not the only way to categorize KWS systems. Some researchers use different categorization and consider KWS systems with and without hypothesis testing (a method for keyword verification) as two different KWS groups.

2.1.1 Keyword-filler based KWS

In a classic keyword-filler based KWS system, the *terms* are defined as a set of keywords and fillers (representing all non-keywords) [33, 35]. The system performs keyword search by decoding input speech into sequences of keywords and fillers with time boundary information. To do so, for each keyword in the system a corresponding keyword AM is built for modeling its acoustic properties, while all non-keywords share the filler acoustic models. For language modeling, a simple keyword-filler loop grammar⁸ (Figure 3(a)) is used. Because of its simplicity, a keyword-filler based KWS system with reasonable performance requires only a small amount of training data. But the system can only be used for detecting the predefined keywords.

Note that despite the high detection rate, keyword-filler based KWS systems sometimes generate lots of false alarms due to inaccurate estimations of $P(q|O)$ for decision making, which could be caused by the over simplified language model for $P(W)$ in Eq. (4). Therefore a second rescoring/verification stage [28, 34, 36-41] is often adopted in keyword-filler based KWS to remove those unwanted false alarms (Figure 2 (d)). The process rescores candidates in the putative list with additional features and models, which allow the systems to estimate $P(q|O)$ more accurately and/or do hypothesis testing so that the system can better determine whether the segments are real query segments. The new scores can be log-likelihood ratios between keyword and background models of the segments [34, 36, 37] or values estimated by complex models such as support vector machine (SVM) [41], multi-layer perceptron (MLP) [39, 40], or deep-neural networks (DNN)[40].

⁸ In this study, a grammar is defined as a search graph or network whose paths from the initial to final nodes represent valid word sequences in a system with corresponding scores, and the graph/network is easily realized by weighted finite-state automata (WFSA).

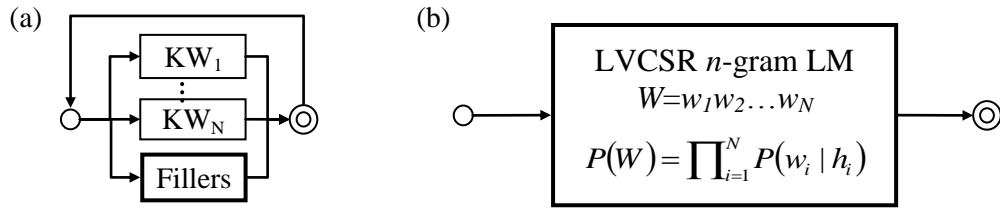


Figure 3. Illustrations of grammars (or language models) in (a) keyword-filler based KWS and (b) LVCSR-based KWS

2.1.2 LVCSR-based KWS

LVCSR-based KWS deals with the keyword search problem from another aspect. Instead of assuming the input speech as a sequence of keywords and non-keywords (fillers), LVCSR-based KWS systems convert input speech into text documents using ASR techniques with n -gram language model [5] based grammar (Figure 3 (b)) first and then perform keyword search on the automatically transcribed documents [22-24]. These text documents can be in different formats, such as N -best sentences or lattices generated by the LVCSR systems at word [22-24] or subword (e.g., syllable [42] or phone [22, 25]) levels. Once speech inputs are converted into text documents, they can be used for searching any keyword, which makes the LVCSR-based KWS systems much more flexible on keyword targets than the conventional keyword-filler based KWS systems. In an LVCSR-based KWS system, n -gram language models play an important role in supporting the system performance. And LVCSR-based KWS systems with well-trained LMs have been shown to outperform keyword-filler based KWS systems [43]. However, a high-performance language model typically requires a significant amount of text training data [31, 44], turning language models into a major performance bottleneck for LVCSR-based KWS in applications where only limited training resources are available.

2.2 A Survey of Recent KWS Research

The keyword-filler based framework was very popular in the early KWS systems [33, 35] as merely a small amount of training data is required to achieve reasonable performance. The computational cost of the systems is also relatively small. These characteristics made keyword-filler based KWS prevalent in the early days. In the 90s, as more powerful machines became available to researchers, and the amount of training speech surging [31,

45], implementing an LVCSR system with good performance was no longer impractical. Since then, LVCSR-based KWS became the mainstream of KWS research [23, 24, 46, 47], especially on languages with plenty of training resources (e.g., English).

The emergence of the LVCSR-based KWS framework makes modern KWS systems more similar to ASR systems. Most of the ASR techniques therefore can be directly applied to KWS systems. Three of the most popular research areas in recent KWS studies are: i) *speech feature representation*, ii) *rescoring and verification*, and iii) *acoustic, language, and pronunciation modeling*. The focus of this dissertation is on the research of acoustic and language modeling. However, to provide a complete background of recent KWS studies, in the following sections a brief introduction for each research area is given.

2.2.1 Speech feature representation

Feature extraction is an important research area in recent KWS research. Extracting more robust and informative features could reduce mismatches between training and testing data and preserve more information in the speech signals for decision making. The performance of the KWS systems can therefore be enhanced. In [48] and [49], it has been shown that in addition to conventional MFCC (Mel-frequency cepstral coefficient) and PLP (perceptual linear prediction) features, tonal features such as pitch and fundamental frequency variation (FFV) can improve KWS performance for both tonal and non-tonal languages. Further, machine-learning based features such as bottle-neck features (BNF), which are extracted with a deep neural network (DNN) structure, have also been shown to provide great performance enhancement to KWS systems [50, 51]. The DNN structure for BNF extraction can be further improved by joint-training with other resource-rich languages. The multilingual BNFs are shown to significantly outperform uni-lingually trained BNFs on KWS tasks [52, 53].

2.2.2 Rescoring and verification

Keyword rescoring and verification (Figure 2 (d)) focuses on improving the estimation of $P(q|O)$ and/or employing hypothesis testing to ameliorate the performance of KWS systems. Common machine-learning-based rescoring and verification tools are HMM [28, 34, 36], logistic-regression (LR) [54-59], SVM [55, 56, 59], MLP [59, 60], and DNN [40]

classifiers. One research direction in this area focuses on making training objectives of the classifiers directly relate to KWS performance measures [58, 59]. The other direction focuses on feature selection [54-57, 60, 61], which has been shown to be a key to building a successful keyword verifier. Note that the machine-learning based rescoring and verification requires yet another model training process. To avoid the additional model training, a graph-based re-ranking approach is proposed in [62] to rescore each putative hit directly with scores of other candidates which are acoustically close to the putative hit. It shows the method achieves similar performance improvement to machine-learning based verification.

Another simple, effective, and popular way to improve the estimation of $P(q|O)$ is system combination. By fusing detection results of a set of complementary KWS systems, the combined KWS performance usually significantly outperforms the individual systems. Note that since scores output from different systems are often in different scales or ranges, score normalization [63, 64] is important before the combination. Complementary systems can be systems with different acoustic features [65], acoustic units [66-68], training approaches [68], and decoding objectives [69].

2.2.3 Acoustic, language, and pronunciation modeling

Acoustic, language, and pronunciation modeling is a research area trying to optimize the training process of parameters Λ and Γ , so that the training data estimated distributions $p_{\Lambda^*}(O|W)$ and $P_{\Gamma^*}(W)$ in Eq. (1) approach the real distributions of $p(O|W)$ and $P(W)$ in the test condition, as mentioned in section 1.1. In this section, recent techniques and research works that have been shown helpful for KWS performance improvement are reviewed.

2.2.3.1 Acoustic modeling

Three of the popular research directions today for acoustic modeling in KWS are: i) discriminative training objectives, ii) new acoustic models, and iii) model training with data augmentation.

Discriminative acoustic modeling has been shown a very successful technique in ASR tasks. By selecting a training objective function that closely relate to the performance metric of the task, the method allows model parameters to be optimized with respect to the task performance directly. In ASR, the training objectives usually relate to

sentence or word error rates. However, in KWS, instead of the whole sentence, only keywords are considered in performance evaluation. In [70], a non-uniform MCE training method is therefore proposed for conventional GMM-HMM systems to address this objective difference in KWS tasks by putting more weight on the error cost function of keyword data. Chapter 5 would also show that similar training objectives can be applied to DNN-HMM systems and provide consistent performance improvement over KWS systems with conventional training criteria [71].

In addition to the discriminative training objectives, some research works focus on the development of new models for the KWS tasks [72, 73]. An interesting work in [73] proposed the use of a point process model (PPM) to replace the conventional hidden Markov model (HMM), which is known for its inaccurate timing event modeling in speech. The PPM for keyword search is a whole-word parametric modeling framework based on the timing of phonetic events rather than the evolution of frame-level phonetic likelihood. Though the PPM-based systems alone received worse KWS performance than HMM-based systems, they provided complementary information to HMM-based KWS systems. KWS performance can be significantly improved after combining detection results from HMM- and PPM-based KWS systems.

Among the three directions, data augmentation [74-79] is the most popular direction in recent KWS acoustic modeling research. The approach tries to increase the amount of training data to alleviate the mismatch between training and test conditions caused by the lack of training data, which is mentioned in section 1.1.2. In *acoustic data perturbation* [74-76], acoustic training data are duplicated and transformed with voice conversion techniques to simulate additional speech data collected from other speakers. The training data size thus increases N times, where N is the number of the simulated speakers. This data perturbation method is effective for alleviating the lack of training data. Another common approach for increasing the amount of training data is semi-supervised training (SST) [75, 77]. The method uses an ASR system to transcribe unlabeled speech data automatically and uses the transcribed data, which may contain errors, with original training data for acoustic model training. However, though the amount of training data is increased in SST, there is no conclusion yet in the community for the effect of SST on performance of KWS. Some research results show SST can slightly improve KWS

performance [77, 78], but some other results show it can only improve performance of ASR and may hurt KWS performance [75, 79].

2.2.3.2 Language modeling

It is interesting that despite the importance of LMs in KWS systems, very few research focuses on language modeling in KWS. To the author's knowledge, among all the hundreds of KWS-related conference publications in the past four years, there are fewer than ten working on language modeling problem in KWS. In reference [21], web text data are used as additional training resources for language modeling. The augmentation has been shown to be effective in improving KWS performance. There has also to be some research that attempts to apply discriminative models such as DNNs for language modeling [80-82]. In [82], a neural-network LM with a training objective emphasizing errors on infrequent words is proposed and shown to provide considerable KWS improvement over conventional n -gram LM based systems. However, a known problem for DNN-based LMs is that they usually require much larger amount of training data than n -gram LMs for model training. DNN-based LMs therefore might not be suitable for all KWS tasks.

2.2.3.3 Pronunciation modeling

For pronunciation modeling, the major research in this area focuses on out-of-vocabulary word (OOV) handling. Similar to ASR, OOVs are major causes of the KWS performance degradation. On the one hand, pronunciations of keywords containing OOVs are unknown to systems since OOVs are not in the system lexicons; the keywords therefore cannot be handled by the systems. On the other hand, for LVCSR-based KWS systems converting input speech into word-level documents for keyword search, keywords with OOVs can never be found in the automatically transcribed documents that consist of only in-vocabulary words. As a result, the OOV keywords would always be missed by the systems. The unknown-pronunciation problem is often handled by grapheme-to-phoneme (G2P) techniques [83, 84], which estimate the pronunciation of OOV keywords from their written forms using rules derived from the original system lexicons. The missed-detection problem for LVCSR-based KWS systems can be alleviated by using keyword proxies [85-88], which replace OOVs in the keywords with their acoustically-similar in-vocabulary words, for the search purpose. Another common way to tackle the missed-

detection problem is to build KWS system in subword levels (e.g., syllable morpheme, and phoneme) and search keywords with their subword-level representations in the automatically transcribed subword-level documents [66, 67, 89]. All the approaches are shown to be effective in reducing the miss rate of OOV keywords and being able to enhance KWS performance.

2.3 Benchmark Datasets for KWS Research

In this section, datasets used in this dissertation are introduced.

2.3.1 TIMIT Corpus

The TIMIT corpus [90] is a read speech dataset designed for acoustic-phonetic studies and for the development and evaluation of automatic speech recognition systems. The data contains broadband recordings of 630 speakers of eight major dialects of American English. For each speaker, ten phonetically rich sentences are recorded. The ten sentences include two dialect sentences (the SA sentences), five phonetically-compact sentences (the SX sentences), and three phonetically-diverse sentences (the SI sentences). The two dialect sentences are meant to expose the dialectal variants of the speakers and are same for all 630 speakers in the training and test sets. Therefore, in general experimental setting for ASR and KWS research, these two SA sentences are not used in either system training nor testing. The TIMIT corpus training set consists of recordings from 462 speakers with a total number of 3696 utterances. The complete TIMIT test set contains a total of 168 speakers and 1344 utterances.

The small corpus size, carefully designed content, and manually annotated transcriptions of the TIMIT corpus make the corpus a very popular dataset for researchers to test new ideas in ASR and KWS. On the one hand, the small data size allows researchers to quickly implement and verify their new ideas without waiting long time for model training. On the other hand, the carefully designed training and test data in the TIMIT corpus provide a wide phonetic and dialectal coverage, which is helpful for researchers to analyze the performance of the new ideas with different test contents. The corpus is therefore selected as one of the benchmark dataset in this dissertation.

2.3.2 IARPA Babel program and OpenKWS evaluations

In addition to the TIMIT corpus, most of the experiments in this dissertation were conducted on the Vietnamese and Tamil corpora provided by the IARPA (Intelligence Advanced Research Projects Activity) Babel program⁹. The Babel program [91] is a project managed by IARPA of the United States since 2012 to address the ASR and KWS problems for new languages with limited resources. The main goal of the program is to bridge the performance gap between rich-resource, well studied languages and limited-resource, lightly studied languages for ASR and KWS systems. The project and its annual open evaluations – OpenKWS13 [92] and OpenKWS14 [93] – have attracted many researchers around the world to this KWS research area. In 2013 and 2014, hundreds of Babel-project-related research papers have been published in various international conferences, e.g., Interspeech, ICASSP (International Conference on Acoustics, Speech and Signal Processing), ASRU (IEEE Automatic Speech Recognition and Understanding Workshop), SLT (IEEE Spoken Language Technology Workshop), and ISCSLP (International Symposium on Chinese Spoken Language Processing).

The audio data of the Babel corpora is conversational speech between two parties over a telephone channel, which can be landlines, cell phones, or phones embedded in vehicles, with the sampling rate set at 8 kHz. Two tasks defined in the corpora are specifically focused in this dissertation: the *limited-language-pack* (LLP) and the *very-limited-language-pack* (VLLP) tasks. In the first task setting, systems have 10 hours of manually transcribed conversational telephone speech data for the entire system construction [92-94] (including both AM and LM training). The second task reduces the training data amount from 10 hours to 3 hours. For both Vietnamese and Tamil corpora, the training, development, and test sets are predefined by IARPA.

2.4 Performance Measures for KWS Systems

In this section, KWS system performance measures used in this dissertation are explained.

⁹ This study uses the IARPA Babel Program Vietnamese and Tamil language collection releases babel107b-v0.7 and IARPA-babel204b-v1.1b with the LimitedLP and VLimitedLP training sets.

2.4.1 The reference and KWS system outputs

The output of a KWS system is a list of putative keyword occurrences in the test data. For each keyword, q , the i^{th} putative occurrence of the keyword reported by the KWS system is a 4-tuple, $O_{q,i}^p = (u_i^p, s_i^p, e_i^p, sc_i^p)$, consists of an utterance identifier (u_i^p), the begin and end time (s_i^p and e_i^p) of the keyword occurrence, and a score (sc_i^p) indicating how likely the keyword q exists at the putative location. Note that usually only putative occurrences with scores above a predefined threshold, thr , are considered as the real output of the system in the performance evaluation. Once the hard-decision (whether the occurrence should be considered) is made with the threshold, the scores, sc_i^p , are no longer needed in the evaluation; the system performance is evaluated specifically at the operating point of the given threshold. However, for other rank-based performance measures, e.g., Figure-of-Merit (FOM)[95], the scores would still be taken into account in the evaluation time.

After a KWS system reports the detection results for the test data, the list of the putative keyword occurrence for the keyword q , $\mathbf{O}_q^p = \{O_{q,i}^p\}$, would then be evaluated against the reference set, $\mathbf{O}_q^r = \{O_{q,j}^r\}$, for the keyword q . A putative occurrence, $O_{q,i}^p$, is considered as a correct detection or a *hit* if the midpoint of the putative occurrence is within the interval from Δ_T before the beginning to Δ_T after the end of a reference occurrence for the keyword q , where Δ_T is a temporal tolerance collar set to 0.5 second in this dissertation. The definitions of hit, missed detection, and false alarm in this dissertation are formulated as follows:

Hit: For a putative occurrence, $O_{q,i}^p = (u_i^p, s_i^p, e_i^p, sc_i^p)$, of keyword q

$$\exists O_{q,j}^r = (u_j^r, s_j^r, e_j^r) \in \mathbf{O}_q^r \quad u_j^r = u_i^p, \quad s_j^r - \Delta_T \leq \frac{s_i^p + e_i^p}{2} \leq e_j^r + \Delta_T \quad \text{and} \quad sc_i^p \geq thr,$$

Missed Detection: if reference $O_{q,j}^r$ is not hit by any putative occurrence whose $sc_i^p \geq thr$,

False Alarm: if putative $O_{q,i}^p$ does not hit any reference and $sc_i^p \geq thr$.

2.4.2 Missed Detection Rate

For a keyword, q , the missed detection rate is defined as

$$P_{Miss}(q) = N_{Miss}(q) / N_{True}(q),$$

where $N_{Miss}(q)$ is the number of missed detection of the keyword q , and $N_{True}(q) = |\mathbf{O}_q^r|$ is the number of reference occurrences of the keyword q .

The overall missed detection rate of a KWS system is defined as the average missed detection rate of all keywords:

$$P_{Miss} = \frac{1}{K} \sum_{q=1}^K P_{Miss}(q) = \frac{1}{K} \sum_{q=1}^K \frac{N_{Miss}(q)}{N_{True}(q)},$$

where K is the number of keywords.

2.4.3 False Alarm Rate

For a keyword, q , the false alarm rate is defined as

$$P_{FA}(q) = N_{FA}(q) / N_{NT}(q),$$

where $N_{FA}(q)$ is the number of false alarms of the keyword q , and $N_{NT}(q)$ is the number of non-target trials for the keyword q . More specifically, $N_{NT}(q)$ is defined as

$$N_{NT}(q) = n_{tps} \times T - N_{True}(q),$$

where n_{tps} is the number of trials per second of speech (and is set to one in this dissertation), and T is the duration of evaluated speech in seconds.

The overall false alarm rate of a KWS system is defined as the average false alarm rate of all keywords:

$$P_{FA} = \frac{1}{K} \sum_{q=1}^K P_{FA}(q) = \frac{1}{K} \sum_{q=1}^K \frac{N_{FA}(q)}{N_{NT}(q)} = \frac{1}{K} \sum_{q=1}^K \frac{N_{FA}(q)}{T - N_{True}(q)}.$$

2.4.4 Actual Term Weighted Value

The Actual Term Weighted Value (ATWV) [46] is a performance measure for spoken keyword search tasks proposed by NIST in 2006. The measure takes both missed detection and false alarm errors of the KWS system into account at the same time. It is a major performance measure for many KWS projects and competitions, e.g., the Spoken Term Detection (STD) evaluation task [46], the Babel project [91], the OpenKWS

competition [92-94], and the Spoken Web Search event [96, 97], in these years. The measure is defined as:

$$\begin{aligned}
 ATWV &= 1 - P_{Miss} - \beta \cdot P_{FA} \\
 &= 1 - \frac{1}{K} \sum_{q=1}^K \left(\frac{N_{Miss}(q)}{N_{True}(q)} + \beta \frac{N_{FA}(q)}{T - N_{True}(q)} \right),
 \end{aligned}$$

where K is the number of keywords, and β is a tunable parameter controlling the penalty weights between miss and false alarm errors. In the Babel program and this dissertation, the β is set as a constant at 999.9. The higher the value of ATWV means the less the miss and false alarm errors for a system, while ATWV=1 indicates a perfect keyword search system with 100% accuracy. Note that the IARPA Babel program set ATWV=0.3 as a benchmark for the performance of successful KWS systems.

2.4.5 Figure of Merit

Another popular performance measure for KWS systems is the Figure of Merit (FOM) [95]. Unlike ATWV, whose performance is evaluated at a specific operating point of a given threshold, the FOM averages the system precisions over the left part of a KWS systems' ROC (receiver operating characteristic) curve. Mover specifically, the FOM is an average keyword precision (hit rate) taken over false alarm rates from 0 FA/kw-hr to 10 FA/kw-hr using Eq. (5).

In the evaluation of FOM, the putative occurrences, $\mathbf{O}_q^p = \{O_{q,i}^p\}$, for each keyword q are first sorted by score from best to worst. Then the system FOM can be evaluated using the formula:

$$FOM = \frac{a \cdot p_{N+1} + \sum_{i=1}^N p_i}{10Hr}, \quad (5)$$

where p_i is the percentage of true hits (precision) found before the i^{th} false alarm across all the keywords, Hr is the duration of evaluated speech in hours, $N = \lfloor 10Hr \rfloor$, and $a = 10Hr - N$ which is a factor that interpolates to 10 false alarms per hour.

CHAPTER 3 RESOURCE-DEPENDENT KEYWORD MODELING

3.1 Introduction

In section 1.1.1 and 2.1, it is known that acoustic models are used for estimating the probability of utterance O given the *term* sequence W , i.e., $p(O|W)$. In ideal case, each *term*, e.g., word, in the system has its own acoustic model describing its pronunciation details. However, having individual AM for each term is usually impractical. One of the major reasons is the lack of training data for parameter estimation. In conventional Gaussian Mixture Model (GMM) based ASR systems, *parameter sharing* is therefore utilized to alleviate this lack-of-training-data problem. Instead of creating term-level AMs, subword-level AMs, e.g., phoneme models, are constructed in the systems. Words like "to", "too", and "two" which have the same phoneme sequence can therefore share the same acoustic model parameters so that there is no need to collect training data for each of the words. Further, for context-dependent phone models, decision-tree-clustering techniques [98] are usually employed to allow parameter sharing among a set of phone states. Hence, even with small amount of training data, AMs with reasonable performance can still be created.

However, parameter sharing also means parameters of some terms or phones are tied together. In other words, there would be no way to distinguish those terms acoustically using the AMs. This could cause performance degradation for ASR or KWS especially when poor LMs are used.

3.2 The "*Share & Split*" training process for keyword modeling

To alleviate this poor acoustic modeling caused by parameter sharing, in this chapter a "*share & split*" training process is proposed for KWS systems. The training process consists of two stages:

1. ***Share stage***: where acoustic models are trained in the conventional way. Namely, the traditional acoustic-modeling process which creates subword-level AMs with decision-tree state tying is utilized to obtain acoustic model parameters, Λ .

2. **Split stage:** where the individual AM of each keyword, λ_q^* , in the system is created by the following process:

For a keyword, q , with speech samples, \mathbf{x}_q , of the keyword, its AM parameters, λ_q , is initialized with Λ to be λ_{q_init} . Then a MAP adaptation [99] is employed to fine tune the parameters –

$$\begin{aligned}\lambda_q^* &= \arg \max_{\lambda_q} g(\lambda_q | \mathbf{x}_q, \lambda_{q_init}, \boldsymbol{\alpha}) \\ &= \arg \max_{\lambda_q} f(\mathbf{x}_q | \lambda_q) h(\lambda_q | \lambda_{q_init}, \boldsymbol{\alpha})\end{aligned}\quad (6)$$

where g is the posterior probability density function (p.d.f) of λ_q given \mathbf{x}_q and λ_{q_init} , f is the p.d.f of \mathbf{x}_q , h is the prior p.d.f of λ_q , and λ_{q_init} and $\boldsymbol{\alpha}$ are the vectors of hyperparameters of the prior density. Since $\boldsymbol{\alpha}$ does not play any role in the formula derived in this research, it will be ignored in later equations for simplicity.

Note that if the keyword, q , has no available training data, \mathbf{x}_q , Eq. (6) reduces to

$$\lambda_q^* = \arg \max_{\lambda_q} g(\lambda_q | \lambda_{q_init}).\quad (7)$$

The solution to Eq. (7) is the mode of the prior density. In other words, λ_q^* would be equal to λ_{q_init} , namely the original model parameters trained in the share stage.

Note that the *share & split* process can also be applied to ASR systems to create whole-word models for each word in the vocabulary if the vocabulary size of the ASR systems is small. The *share & split* process has been shown effective for LVCSR tasks [100]. The word error rate can be improved about 4~13% absolute when no language models were used.

3.3 Hierarchical model optimization with *split* process

Eq. (6) optimizes acoustic models for each keyword at whole-word level to address the parameter sharing issue in the convention training process. However, in many cases, keywords in a KWS system do not have training samples, \mathbf{x}_q . Acoustic models of those keywords thus cannot benefit from the *split* process. To alleviate this problem, a hierarchical *split* process is proposed [101]. In the hierarchical modeling, the *split* process is first employed at phone level to break the phone-state tying and obtain better parameter

estimation for each phone model. Then the whole-word level *split* process is utilized on top of the phone-level optimized parameters for further model optimization. The following paragraphs explain the modeling process in detail.

For a phone (usually triphone), p , with speech sample, \mathbf{x}_p , its acoustic model parameters, $\boldsymbol{\theta}_p$, initialized with $\boldsymbol{\Lambda}$ are assumed to be $\boldsymbol{\theta}_{p_Init}$. The phone-level *split* process is:

$$\boldsymbol{\theta}_p^* = \arg \max_{\boldsymbol{\theta}_p} g(\boldsymbol{\theta}_p | \mathbf{x}_p, \boldsymbol{\theta}_{p_Init}) = \arg \max_{\boldsymbol{\theta}_p} f(\mathbf{x}_p | \boldsymbol{\theta}_p) h(\boldsymbol{\theta}_p | \boldsymbol{\theta}_{p_Init}), \quad (8)$$

where $\boldsymbol{\theta}_p^*$ is the optimized model parameters for the phone, p , given the speech sample, \mathbf{x}_p . When there is no speech data for the phone, $\boldsymbol{\theta}_p^*$ is simply equal to $\boldsymbol{\theta}_{p_Init}$.

Suppose $\{\boldsymbol{\theta}_p^*\}$ represents phone-level optimized model parameters for all the phones presenting in the keyword of interest, q , the whole-word level optimization for q is then conducted by a MAP adaptation with $\boldsymbol{\lambda}'_{q_Init} = \{\boldsymbol{\theta}_p^*\}$ as the hyperparameters of the prior density for the keyword model, $\boldsymbol{\lambda}_q$. And the estimation can be expressed as:

$$\begin{aligned} \boldsymbol{\lambda}_q^{**} &= \arg \max_{\boldsymbol{\lambda}_q} g(\boldsymbol{\lambda}_q | \mathbf{x}_q, \boldsymbol{\lambda}'_{q_Init}) = \arg \max_{\boldsymbol{\lambda}_q} g(\boldsymbol{\lambda}_q | \mathbf{x}_q, \{\boldsymbol{\theta}_p^*\}) \\ &= \arg \max_{\boldsymbol{\lambda}_q} f(\mathbf{x}_q | \boldsymbol{\lambda}_q) h(\boldsymbol{\lambda}_q | \{\boldsymbol{\theta}_p^*\}) \end{aligned} \quad (9)$$

An advantage of using Eq. (9) to replace Eq. (6) for whole-word level optimization is that with better initial values, keyword model parameters, $\boldsymbol{\lambda}_q$, can be better estimated. Also when a keyword has no corresponding training data, Eq. (9) can be reduced to

$$\boldsymbol{\lambda}_q^{**} = \arg \max_{\boldsymbol{\lambda}_q} h(\boldsymbol{\lambda}_q | \boldsymbol{\lambda}'_{q_Init}) = \arg \max_{\boldsymbol{\lambda}_q} h(\boldsymbol{\lambda}_q | \{\boldsymbol{\theta}_p^*\}), \quad (10)$$

and the keyword model would be the phone-level optimized $\boldsymbol{\lambda}'_{q_Init}$ instead of the original $\boldsymbol{\lambda}_{q_Init}$. Therefore, even for keywords with no training samples, the *split* process can still provide gains for them.

3.4 DNN-based keyword verification

For keywords with plenty of training samples (usually short keywords [102]), more complex and discriminative models can be used in addition to the GMM-HMM models for further performance improvement [40]. In this research, keyword-specific deep neural

network (DNN) verifiers are proposed to rescore putative hits detected by the KWS system.

Usually, input features for NN-based keyword verifiers are confidence-score outputs of the KWS systems [38] or acoustic features of the putative segments [39]. In the proposed DNN verifier, the input features are state-alignment-warped MFCC (mel-frequency cepstral coefficient) [103] and Government Phonology (GP)[104, 105] articulatory features of the putative segment. The GP features, which contain 11 attributes representing different spectral properties of speech sound, are used as an additional articulatory knowledge source to further improve the verification performance. Outputs of the DNN model are posterior probabilities of given putative hits being true hits or false alarms. The proposed keyword verification framework is illustrated in Figure 4.

Note that for each keyword, a DNN verifier is trained only on the putative hits generated by performing KWS on the training data. Therefore, the training time for the proposed verifiers is significantly less than keyword-independent verifiers which use the whole training set for training. The three major contributions of the proposed DNN-based keyword verification (i.e., the utilization of DNN, the input feature generation for NN models, and the use of GP features) are described in detail in section 3.4.1 to 3.4.3, respectively.

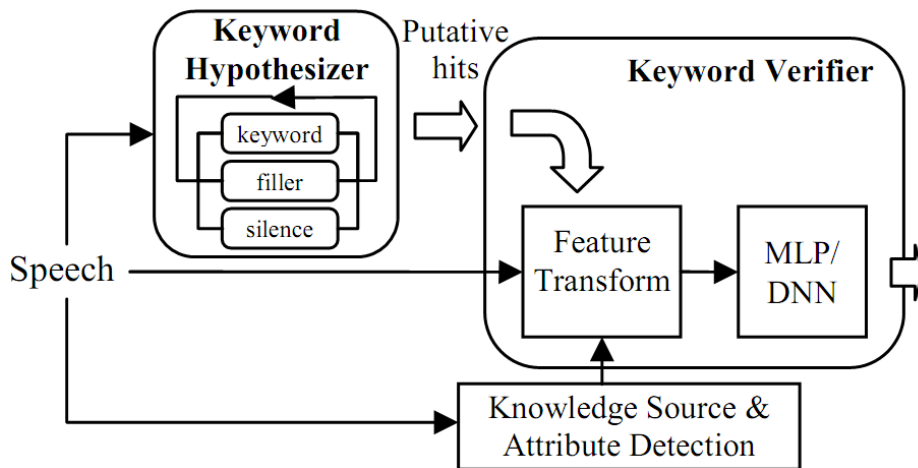


Figure 4. Illustration of the KWS framework with an additional keyword verification stage for keywords with plenty of training samples.

3.4.1 Deep neural network

In conventional neural network based keyword verification, the neural networks are usually multi-layer perceptron (MLP) models with a single hidden layer [39]. A major reason for the selection of single-hidden-layer MLPs is that the MLPs with more than one hidden layers often easily over-fit the training data and have poor performance on test data if the training data amount is not large enough. Experimental results also show that MLPs with single hidden layer usually outperform MLPs with more than one hidden layers [39].

Note that in the MLP training process, parameters are randomly initialized. In a recent study [106], it has been shown that the over-fitting problem in MLPs can be alleviated by carefully initializing the model to some better start point. The deep neural network (DNN) [106], which employs a restricted Boltzmann machine (RBM) pre-training process for parameter initialization, therefore, can be trained with more than one hidden layer without the over-fitting problem. It has been shown that the DNNs work very well in speech recognition tasks and significantly outperformed the GMM-based ASR systems [107]. Motivated by the success of DNNs in ASR tasks, in this research, the DNNs are adopted for the realization of the NN-based keyword verifier [40].

3.4.2 Input features generation for NN models

In the proposed verification framework, for each keyword, the input feature of its NN-based verifier for a putative hit is the combination of all acoustic feature frames in the putative segment. However, due to the nature of human speech, the lengths of putative segments for a keyword are usually different each time. In other words, the numbers of acoustic feature frames are often different in different putative segments even for the same keyword. Since NN can only manage input features with fixed length, it is necessary to transform the acoustic feature frames in the putative segments into a fixed-length feature vector before feeding them into the NN verifier. In a previous study [39], this was done by dividing each putative segment into three equal duration sub-segments and representing each sub-segment by the averaged feature vector in it. The three averaged feature vectors of the sub-segments are then concatenated into a super feature vector to represent the putative hit. In the 3-equal-segment approach, it is assumed that

the pronunciation of a keyword can always be represented by the means of the three sub-segments. However, the assumption is usually not true in real tasks.

In this study, a HMM-state-alignment (HSA)[103] based feature transformation is proposed to be used for the NN-base verifier. In the HSA based transformation, for each keyword, the acoustic feature frames of a putative hit are first segmented by the state alignment of the keyword's GMM-HMM based model. The mean of feature frames in each state is then concatenated to form the input vector for the NN model.

3.4.3 Government phonology features

In reference [108], it has been shown that the performance of KWS systems could be significantly improved by integrating additional acoustic-phonetic knowledge sources for KWS result pruning and rescoring. On the one hand, the machine learning extracted articulatory features are usually more robust to speaker/environment variations than the raw MFCC/PLP features. On the other hand, the phonetic/phonological features are derived from the human knowledge of speech sound structures, which could provides complementary information to the MFCC/PLP features. Motivated by the work in [108], in the proposed verification framework, in addition to the conventional MFCC features, a Government Phonology (GP) feature set [104] is exploited as a supplementary articulatory knowledge source to further enhance the performance of keyword verification.

The Government Phonology features are attributes derived by examining the spectral properties of speech sounds. In the GP feature set, sounds are divided into a collection of primes (attributes); and every phonological phenomenon can be represented by fusing the primes structurally. In this study, the King and Taylor's modified version [105] containing 11 attributes of the GP feature set is used (see Appendix B). The GP features are detected at the frame level by a recurrent neural network detector proposed in [109].

The GP features are integrated with the baseline MFCC features by feature concatenation. Namely, for each putative segment, both frame-level GP features and MFCC features are first transformed into fixed-length feature vectors using the HSA transformation described in section 3.4.2. The two feature vectors are then concatenated into a super feature vector and used as input for the NN verifier.

3.5 Experiments

3.5.1 Experimental setup

All the experiments were conducted on the TIMIT corpus [110] divided into three parts: a training set (3296 utterances, 2.79 hours), a development set (400 utterances, 0.34 hours), and a test set (1344 utterances, 1.14 hours). The training and development sets are subsets of the standard TIMIT training set, while the test set is the standard TIMIT test set. Dialect utterances (SA1 and SA2) were not used in the experiment. The acoustic features used in the experiments are 12 MFCCs plus energy and their first and second time derivatives. The CMU/MIT phone set, which contains 39 phonemes, was used. Triphone HMMs with decision-tree clustering (*share* process) were used for acoustic modeling in an initial ASR system. The AM of the initial system was then used for parameter initialization of keyword models in the *split* process.

The proposed method was tested on keyword-filler based KWS systems. Thirty words in the TIMIT vocabulary were selected as keywords in the KWS experiments. The selected keywords were divided into three groups: (i) words having no speech data in the TIMIT training corpus (14 words), (ii) words having some training tokens in the training set (7 words), and (iii) words having plenty of training samples in the training data (9 words). An empirical threshold of 50 samples is used to determine whether a word belong to Group 2 or 3. Table 2 lists the chosen keywords in the three groups. For each keyword, a KWS system was built for detection.

A keyword occurrence is considered correctly detected, or *hit*, if the mid-point of the detected reference falls within the time boundaries of the hypothesis. To evaluate the performance of the overall system, the figure-of-merit (FOM) [95], which is an upper-bound estimate of keyword spotting accuracy averaged over 1 to 10 false alarms per hour, was used. An average FOM over all keywords was used as the overall performance measure.

Table 2. Keywords used in this study. Keywords are divided into three groups according to the count of training instances: 1 – no training instance, 2 – some training instances (<50), 3 – plenty of training instances (>=50).

Group 1	overalls, potatoes, greg, tooth, shore, products, silly, prestige, avoid, popular, pretty, expense
Group 2	before, after, people, these, always, without, money
Group 3	was, with, his, this, from, not, but, every, often

There were two baseline keyword-filler based KWS systems in the experiments. The difference between the two systems is that the keyword model in the second baseline system was word-level optimized. Filler models used in the two baseline systems had 9 states and 78 Gaussian mixture components per state, and the whole TIMIT training set was used for filler model training. Confidence scores of detected keywords were evaluated as log likelihood ratios (LLR) [111] between the keywords and the filler models on the detected keyword segments.

For keyword verifiers, the average depth of the DNN models is three hidden layers. Each hidden layer contains 512 hidden nodes. An unsupervised pre-training process [106] was adopted before regular backpropagation training to make sure the DNN models were initialized at a good starting point.

3.5.2 Parameter sharing analysis

To illustrate the degree of parameter sharing in conventional ASR and KWS systems, a parameter sharing analysis was conducted on the initial ASR system. Parameter-sharing factors (PSF) at word- and phone-levels are defined as follows: (1) *Word-level PSF* – the average number of words sharing the same triphone. For example, triphone /ay-n+d/ is shared by the words "blind" and "finds". This factor can be computed using the TIMIT dictionary; and (2) *Phone-level PSF* – the average number of triphones sharing the same HMM state. This number can be calculated by examining the baseline triphone acoustic models, which are HMMs trained with conventional decision-tree clustering.

Table 3 shows that, for the 6,237 words in the TIMIT vocabulary, a triphone is shared by 5.59 words on the average. And in the baseline triphone acoustic model, which contains 35,468 triphones and 397 Gaussian mixture states, each state is on average shared by 268 triphones¹⁰. In other words, though these 268 triphones were supposed to model different pronunciations, the acoustic model lost the ability to differentiate these sounds due to parameter sharing. Note that even when expanding the baseline model to 3,000 states (which is a number used by most LVCSR systems) by adjusting the decision-tree clustering threshold, each state is still shared by about 35 triphones. This phone-level PSF is still considered relatively high.

Table 3. Word- and phone-level parameter sharing factors in the TIMIT corpus.

Level	Word	Phone
Parameter sharing factor	5.59	268.02

3.5.3 Experimental results for hierarchical keyword modeling

Table 4 lists the average FOMs of the baselines and the proposed KWS systems. The first three columns show the average FOMs of the keywords in Groups 1, 2, and 3, respectively. The last column is the average FOM for all 30 keywords. Note that since all the words in Group 3 were short words consisting of four phones or less, the FOM of the keywords in Group 3 is much lower when compared with the performance of the keywords in Groups 1 and 2.

From the results of the two baseline systems shown in the first and second rows, it can be found that word-level *split* process is indeed helpful for improving KWS performance. The overall FOM was increased from 38% to 45%. However, despite the significant FOM improvement for the keywords in Groups 2 and 3, the word-level optimization is not helpful for the keywords in Group 1 due to the lack of training data.

The third to fifth rows show the FOMs of the KWS systems using the proposed hierarchical modeling method on different optimization levels. Keyword modeling started

¹⁰ This number is calculated by $35468/(397/3)$ since each phone has three states.

with phone-level optimization (shown in the third row); word-level optimization was then applied to the phone-level optimized models (the fourth row); if keywords had sufficient training data, DNN models of the keywords were further built and used for verification (the fifth row). If keywords did not have enough training data for optimization on the current level, the models constructed on the previous level were kept; the performance is then shown in gray text in Table 4.

The third row shows that after utilizing phone-level optimization, the overall FOM can be improved from 38% to 46%, which is slightly better than the 45% of the second baseline system. This is because, in addition to the improvements of the KWS systems of the keywords in Group 2 and 3, the FOM performance of the KWS systems of the Group 1 keywords was also improved by phone-level optimization. The success of phone-level optimization also suggests a direction for training data collection of keywords whose speech data are not easy to obtain. In other words, for keywords having very few speech samples, data of other words sharing triphones with the keywords can be collected and used for phone-level optimization.

Table 4. Performance comparison of the baseline and the proposed KWS systems at different optimization levels. Numbers in gray are the performance using keyword models at the lower optimization level since the systems are lacking training data and cannot be optimized on the current level.

KWS Systems	FOM (%)			
	Group 1	Group 2	Group 3	Overall
Baseline 1	60.96	29.05	9.79	38.16
Baseline 2 – Word-level optimization	(60.96)	44.83	21.39	45.33
Phone-level optimization	66.51	42.08	17.79	46.19
Phone- + Word-level optimization	(66.51)	49.06	21.87	49.05
DNN-based Keyword Verification	(66.51)	(49.06)	42.78	55.32

When it comes to word-level optimization, KWS for the Group 1 keywords could not be improved due to the lack of training data, while KWS accuracies for the keywords in

Groups 2 and 3 can be further improved. The overall FOM was 49% after word-level optimization. If keyword-specific DNN models are further built for the Group 3 keywords, the FOM performance of the Group 3 keywords could be further improved to 42%. Finally, the overall FOM was 55% averaged over all 30 keywords.

In summary, using phone-level optimization is beneficial to all the KWS systems. For the keywords in Group 2, doing word-level optimization after phone-level optimization is significantly better than performing word-level optimization directly. This shows that phone-level optimization provides a better prior for modeling the Group 2 keywords. On the other hand, for the keywords in Group 3, additional phone-level optimization did not help word-level keyword modeling much. This might be because the Group 3 keywords had plenty of training data so that model priors had no significant effect on the final parameter values. However, when using the data of the Group 3 keywords to build keyword DNN models for verification, the KWS performance of Group 3 keywords can be further improved significantly. The proposed resource-dependent hierarchical keyword modeling approach thus provides a training process making the best use of the limited training data for each kind of keywords in the KWS systems.

3.5.4 Experimental results for keyword verification

Table 4 only shows the performance of the best DNN-based keyword verifier. To further study the contribution of each technique (i.e., HSA feature transformation, GP features, and DNNs) to the system performance improvement, in the third experiment the FOMs of the Group 3 keywords with different NN verifier configurations were evaluated. The results of the KWS systems are summarized in Table 5. In Table 5, the first row shows the FOM of the proposed phone- + word-level optimized system without verification on the Group 3 keywords. The FOM of the system is 22%.

The second to sixth rows in Table 5 show the system FOM achieved by imposing different second-stage keyword verifiers. Unlike what's reported in [39], the FOM decreased slightly after adding the keyword verifier with an equal-segment feature transform [39], from 22% down to 19%. However, with the proposed HMM-based feature transform, the FOM improved from 22% to 27% as shown in the third row of Table 5. After adding the GP features to the keyword verifier as the additional knowledge

source, the overall system performance was further improved to 38%. So far, the NNs used for the verifiers are MLPs with only one hidden layer. The fifth and the last rows in Table 5 display the FOMs when using multiple hidden layers. The number of the hidden layers was decided by the model performance on the development set. Without pre-training, the FOM of the verifiers with more than one hidden layers dropped slightly to 37.7%, which verified the finding in [39]. However, by employing pre-training, the system with DNN-based keyword verifiers achieved the best FOM of 43%.

Table 5. Comparison of the KWS systems with different settings. $n=1$ means NN with a single hidden layer; $n \geq 1$ means NN with one or more than one hidden layers, while the number n is selected empirically based on the performance on the development set.

KWS Systems	FOM (%)	Runtime (sec)	Real Time Factor
Phone- + Word-level optimization (without keyword verification)	21.87	83.6	0.02
KV MLP $n=1$ (3-eq-seg transform) [MFCC]	19.01	83.6 + 0.012	0.02
KV MLP $n=1$ (HSA transform) [MFCC]	27.23	83.6 + 0.012	0.02
KV MLP $n=1$ (HSA transform) [MFCC+GP]	38.08	83.6 + 0.013	0.02
KV MLP $n \geq 1$ (HSA transform) [MFCC+GP]	37.68	83.6 + 0.037	0.02
KV DNN $n \geq 1$ (HSA transform) [MFCC+GP]	42.78	83.6 + 0.037	0.02

Table 5 also compares runtime of the systems. The average runtime of the proposed KWS systems detecting the Group 3 keywords on TIMIT test data is 83.6 seconds without keyword verification. This corresponds to a real time factor (RTF) of 0.02 on the 1.14 hours TIMIT test data. The second-stage NN-based keyword verification only took less than 0.04 seconds to rescore the putative hits. The verification did not increase much delay to the KWS systems.

Figure 5 shows the ROC curves of the Group 3 keywords before and after applying the DNN-based keyword verification to the proposed KWS systems. It is clear that the proposed DNN-based keyword verification significantly improved the keyword hypothesizer and provided much higher true positive rate in all system operating points.

In summary, using the keyword verifier with HSA feature transform provided 5% of the FOM improvement. The FOM can be further improved by adding the GP knowledge source, which brought another 11% of the FOM improvement. Finally, by using the DNN techniques, another 5% of FOM improvement can be achieved.

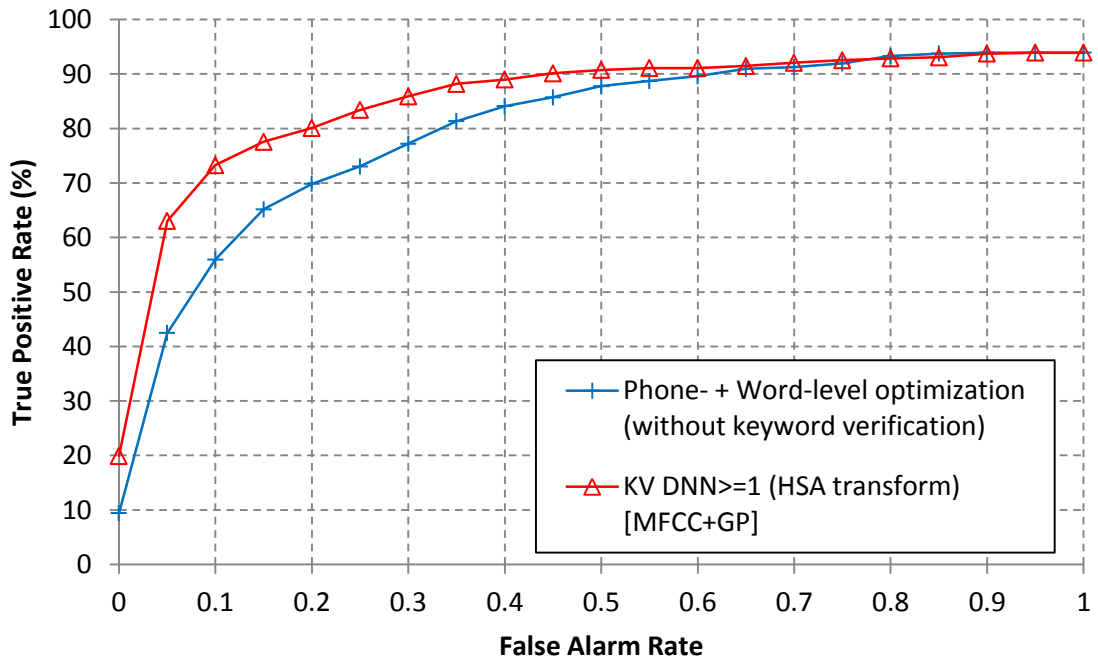


Figure 5. ROC curves of the Group 3 keywords for the phone- + word-level optimized keyword hypothesizers and the DNN-based keyword verifiers (rescore the putative hits generated by the hypothesizers).

3.6 Conclusion

In this chapter, a resource-dependent hierarchical keyword modeling framework is proposed for keyword-filler based KWS systems. The framework provides three model optimization levels for keywords with different amount of training data. For keywords with no training samples, the keyword models can be benefited from the phone-level

optimization in the *share & split* keyword modeling process. For keywords with some training samples, the word-level optimization can be conducted after the phone-level optimization to further improve the keyword models. When keywords having plenty of training samples, the proposed DNN-based keyword verifiers can be trained to further enhance the KWS performance. The framework thus provides a data-efficient modeling process exploiting all the in-domain training data when building keyword models. Detailed analysis of parameter sharing in conventional keyword models and investigation of constructing effective NN-based keyword verifiers are also provided. Experimental results show that the proposed approach significantly improves the KWS performance of keyword-filler based KWS systems, even when keywords have no training data.

CHAPTER 4 KEYWORD-AWARE LANGUAGE MODELING

4.1 Introduction

Despite the fact that keyword-filler based KWS has a longer history, most of the state-of-the-art KWS systems today are using the LVCSR-based KWS framework. With more detailed n -gram LM decoding grammars, LVCSR-based KWS usually provides better KWS performance when sufficient training data are available for n -gram LM training. It also provides better keyword flexibility for systems to detect new keywords without reprocessing the speech signals. However, a high-performance n -gram LM typically requires a significant amount of text training data [31, 44] to reduce the mismatch between training and test conditions as mentioned in section 1.1.2, turning n -gram LMs into a major performance bottleneck for LVCSR-based KWS in applications where only limited training resources are available.

In this chapter, a "keyword prior underestimation problem" of n -gram LM, which is caused by domain mismatches between LM training data and test keywords, will be revealed. The underestimation problem leads LVCSR-based KWS systems to a high missed detection rate for multi-word keywords. To alleviate this prior underestimation problem, a keyword-aware language modeling framework is proposed. The proposed framework reduces the mismatch between training and test conditions of n -gram LM in KWS systems by exploiting keyword information. Experimental results show that the proposed framework provides very significant performance improvement for LVCSR-based KWS system. A 50% relative ATWV performance improvement is observed in both Babel Vietnamese and Tamil limited language pack (LLP) tasks. The new systems also maintain flexibilities for detecting new keywords as in the conventional LVCSR-based KWS systems.

The chapter starts with a comparison of keyword prior estimations employed in conventional keyword-filler based KWS and LVCSR-based KWS in section 4.2. The prior underestimation problem of n -gram LM for multi-word keywords is then explained in section 4.3. In section 4.4, a keyword-aware language modeling framework, which integrates the keyword-filler loop grammar into n -gram LM grammar to alleviate the

underestimation problem, is introduced. Realizations of the proposed language modeling framework are presented in section 4.5 and 4.6. Section 4.7 shows experimental results.

4.2 Spoken Keyword Search Problem

Spoken keyword search can be considered as an application of the automatic speech recognition (ASR) technology that focuses on the recognition of keywords¹¹. Given a speech utterance O and a text-based query q , a KWS system detects the query q in the utterance by finding the best *term* sequence, W^* , corresponding to the utterance O as follows:

$$W^* = \arg \max_w P(W | O). \quad (11)$$

If the query, q , does exist in the utterance, then we expect $W^* = h \cdot q \cdot f$, where h and f are *term* sequences (which we do not really care) preceding and following the query in the utterance, and " \cdot " is a concatenation operator. Otherwise a missed detection error occurs. Note that usually miss errors are considered more serious than false alarms in KWS since the latter can still be removed with a further utterance/keyword-verification stage [34, 38, 40, 101].

With Bayes' rule, Eq. (11) can be rewritten as

$$W^* = \arg \max_w p(O | W)P(W), \quad (12)$$

where $p(O|W)$ is the probability of the utterance, O , given the hypothesized *term* sequence, W ; $P(W)$ is the prior probability for the hypothesized *term* sequence. In general, the probability $p(O|W)$ can be computed with acoustic models, and $P(W)$ is modeled by system language models. The *term* sequence W^* in Eq. (12) can then be searched with Viterbi algorithm with beam search to alleviate the computational burden caused by the large search space. Note in many applications, instead of using only the 1-best result, W^* , lattices or N -best sentences with confidence scores can also be generated for keyword

¹¹ This is from the perspective of ASR-based KWS, please see sectiond 1.1 and 2.1 for more details and other perspectives of the KWS problem.

detection [22-24, 43]. Thus for an utterance containing a query, q , it is a key to make sure the hypothesized *term* sequences, $W=h \cdot q \cdot f$, containing the query have probabilities high enough to stay in the search beam and be preserved in the final lattices or N -best sentences. More precisely, the probabilities of $P(q|h)$ estimated by language models should be sufficiently high to allow the query-containing search path to be retained in the beam width when processing the speech segment of the query in the utterance. Otherwise, the query would be missed.

The two conventional KWS groups, i.e., keyword-filler based and LVCSR-based KWS frameworks, utilize a similar acoustic modeling approach, but they are very different in the definition of *terms* and the estimation of the prior probability, $P(W)$. The differences in their language modeling approaches lead to their contrastive performance characteristics as explained in the following sections.

4.2.1 Keyword-filler based KWS

In a standard keyword-filler based KWS system, the *terms* are defined as a set of keywords and fillers (representing all non-keywords). The probability of each *term* in the utterance is usually assumed to be context independent in the standard keyword-filler loop grammar (shown in Figure 3(a)), namely $P(q|h)=P(q)$. And it is often assumed that $P(q)$ is a uniform distribution over all *terms* and thus equal to $1/N$, where N is the number of *terms* in the system. For most keyword-filler based KWS systems, N is a number smaller than 100 [35, 40, 43, 101]. Since the prior probabilities for most keywords are less than 10^{-4} in practical settings¹², by assuming $P(q)=1/N \geq 1/100 \gg 10^{-4}$, the estimation of $P(q|h)=P(q)$ in standard keyword-filler based KWS is usually sufficient to preserve the keyword in the search path in most cases. As a result, the systems usually achieve a high detection rate despite the over-estimated priors sometimes creating a great number of false alarms as well.

¹² For example, Figure 7 shows the average keyword prior probabilities in the IARPA Babel Vietnamese data are in the range of 5×10^{-5} to 5×10^{-6} .

4.2.2 LVCSR-based KWS

In LVCSR-based KWS, n -gram is used for the evaluation of $P(q|h)$. Given an L -word query, $q=(w_1, w_2, \dots, w_L)$, the conditional probability of q given h is evaluated as

$$P(q|h) = P(w_1 w_2 \dots w_L | h) \cong \prod_{i=1}^L P_{n\text{-gram}}(w_i | h_i), \quad (13)$$

where $P_{n\text{-gram}}(\cdot)$ is the probability estimated by the system n -gram LM, and h_i is the history of w_i in the query q dictated by the order of the n -gram LM. This prior estimation helps LVCSR-based KWS achieve better detection accuracy than keyword-filler based KWS when sufficient LM training data is available [43].

4.3 The Prior Underestimation Problem for Multi-word Keywords

Empirically, using n -gram LMs for keyword prior probability estimation works fine for KWS tasks when all the systems keywords are single-word keywords. However, in real world applications, it is often user queries are multi-word keywords, e.g. "volunteer firefighter", "needle in a haystack", "deep neural networks", and "hidden Markov models". Further, most of these multi-word keywords are meaningful phrases and can be seen as semantic units just like single-word keywords. Therefore, they usually have the same occurrence frequency as the single-word keywords in speech utterances. However, the prior probabilities of these multi-word keywords are usually underestimated by the n -gram LMs. This underestimation makes the multi-word keywords being easily pruned away from the search beam in the decoding process. LVCSR-based KWS systems therefore could have very high miss rate for the multi-word keywords.

There are two major reasons for the n -gram LMs having the prior underestimation problem for multi-word keywords. First, as shown in Eq. (13), the conditional keyword prior of the L -word query q given the history h , $P(q|h)$, is evaluated as a product of the L n -gram probabilities in the query q . As long as there are some n -gram probabilities in the product being underestimated because of domain mismatch, the overall prior probability $P(q|h)$ could be underestimated. Since usually the amount of LM training text is insufficient to cover all keyword-related domains, it is often that the LMs get into this domain mismatch situation and generate extremely low probability estimations for n -

grams in the keywords. The problem is more pronounced for multi-word keywords with a large L because more n -gram probabilities are involved in the multiplications.

The second reason is related to the back-off smoothing technique commonly used in n -gram LMs. Though the smoothing technique successfully salvages the probability estimation for single-word keywords, it is the major cause for the prior underestimation problem of multi-word keywords when the keywords are unseen in the LM training data. For example, suppose we are estimating the prior probability of a keyword “volunteer firefighter” using a bigram LM (as shown in Figure 6), but the phrase “volunteer firefighter” is absent from the LM training data. Because the bigram LM has never seen the word “firefighter” coming after the word “volunteer” in the training phase, the bigram probability of $P(\text{firefighter} | \text{volunteer})$ would be backed-off to the unigram probability of the word “firefighter” multiplies the back-off factor $\alpha(\text{volunteer})$. Now, it is clear that the bigram LM assumes the events of “volunteer” and “firefighter” are two independent events. However, since we know that the keyword “volunteer firefighter” is a semantic unit, the word events of these two words should be highly correlated. And the n -gram LM would therefore seriously underestimate the prior probability of the keyword “volunteer firefighter” because of the “*incorrect independent event assumption*” made by the n -gram LM with back-off smoothing.

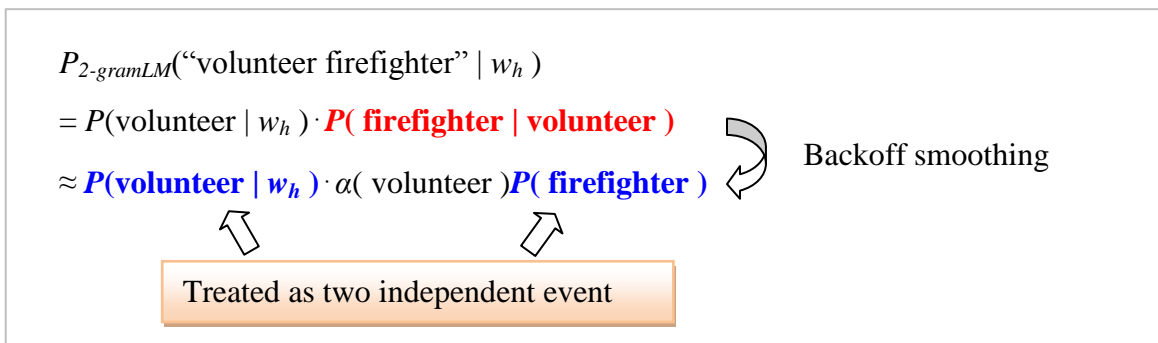


Figure 6. An example of the “*incorrect independent event assumption*” made by an n -gram LM ($n=2$ in this example) with backoff smoothing when the multi-word keyword is unseen in the LM training data. The incorrect assumption causes serious prior underestimation for multi-word keywords.

Figure 7 shows how serious the prior underestimation is for the multi-word keywords’ probabilities estimated with n -gram LMs. The figure compares the keyword probabilities

counted directly from the transcriptions of the Babel Vietnamese *evalpart1* evaluation data (ground-truth) and the probabilities estimated by the n -gram LM ($n=3$) in the Vietnamese LLP task. Obviously, the ground-truth probabilities of keywords with length from 1 to 6 words are about the same at $e^{-10} \approx 4.5 \times 10^{-5}$. However, the prior probabilities estimated by n -gram LM is a linear function of the keyword lengths; when keyword lengths are larger than 2, the estimated probabilities begin seriously underestimated.

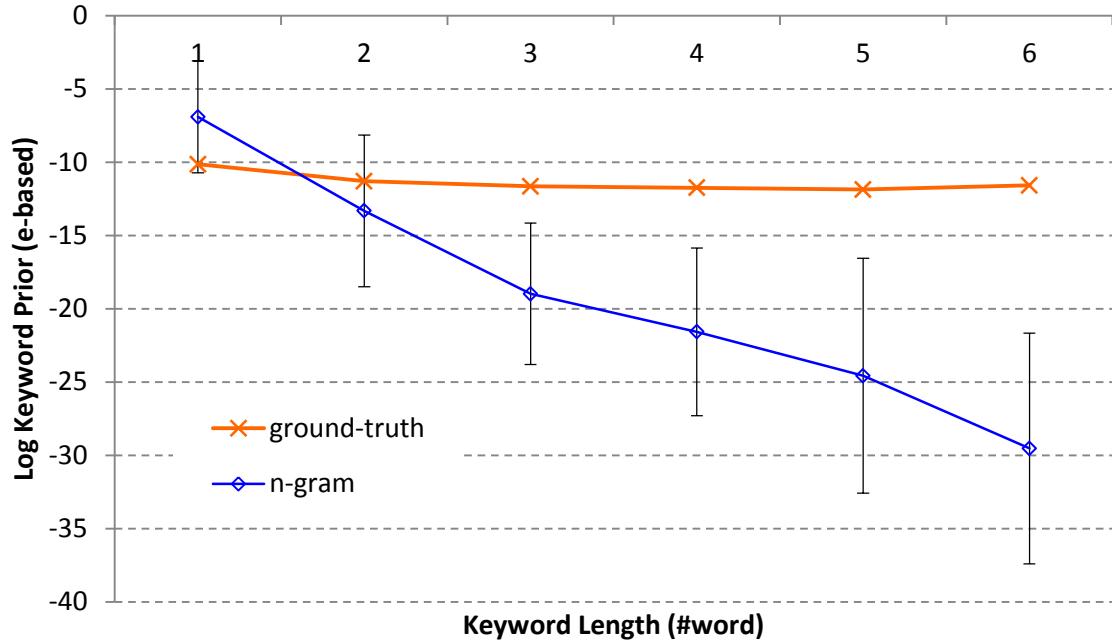


Figure 7. Keyword probabilities counted directly from the transcriptions of the Babel Vietnamese *evalpart1* evaluation data (ground-truth) and the probabilities estimated by the n -gram LM ($n=3$) in the Vietnamese LLP task. The n -gram LM seriously underestimated the probabilities of multi-word keywords.

4.4 Keyword-Aware Language Modeling

When the system n -gram LMs are trained with limited or topic-mismatched data, LVCSR-based KWS suffers from the abovementioned prior underestimation problem leading to a high miss rate for the multi-word keywords in KWS. The problem can be alleviated by making more efficient use of keyword information available to the LVCSR-based KWS systems. To do so, a keyword-aware language modeling framework is proposed in this research to integrate the prior estimation in keyword-filler based KWS into the LVCSR-based KWS framework for an accurate evaluation of the keyword priors.

As in LVCSR-based KWS, the proposed keyword-aware KWS framework also utilizes an underlying LVCSR system but with keyword priors computed by:

$$P_{KW-aware}(q|h) = \max\{P_{n-gram}(q|h), \kappa\}, \quad (14)$$

where κ is a parameter for query q to control the minimum keyword prior value allowed in the system. Note that if κ is set to 0, Eq. (14) would become Eq. (13), which is the prior estimation in LVCSR-based KWS. When setting κ to $1/N$ for an N -keyword task, Eq. (14) becomes the prior used in the keyword-filler based KWS since in most cases $1/N$ is larger than $P_{n-gram}(q)$. The two conventional KWS frameworks therefore can be seen as special cases of the proposed framework. By tuning the parameter κ for each query in the system, we are able to adjust the sensitivity of a system to the keywords of interest even when the n -gram LM of the system is not well trained.

The proposed keyword-aware framework also preserves the keyword flexibility because of the underlying LVCSR system. New keywords can be searched in the transcribed documents of the proposed system without reprocessing the speech signal. Note that in the keyword-aware LM only the prior probabilities of the preselected keywords are modified, while the rest of the n -gram probabilities in the original LM remain the same. The transcribed document of the proposed system is therefore the same as the original LVCSR-based KWS system for regular *terms* in the system vocabulary. As a result, detection accuracies of the new keywords, whose prior probabilities are not modified, would be similar to the original LVCSR-based KWS.

The concept of keyword-aware language modeling can be realized by integrating the decoding grammars of the two conventional KWS frameworks. By adding the keyword paths used in the keyword-filler grammar (Figure 8(a)) to the n -gram LM based grammar used in the LVCSR-based systems (Figure 8(b)), we obtain a *keyword-aware (KW-aware) grammar* (Figure 8(c)). In this research, both exact realization and approximate approach of the KW-aware grammar, which suit LVCSR systems with different architectures, are proposed. The proposed keyword-aware language modeling framework can therefore be easily implemented in any state-of-the-art LVCSR-based KWS systems.

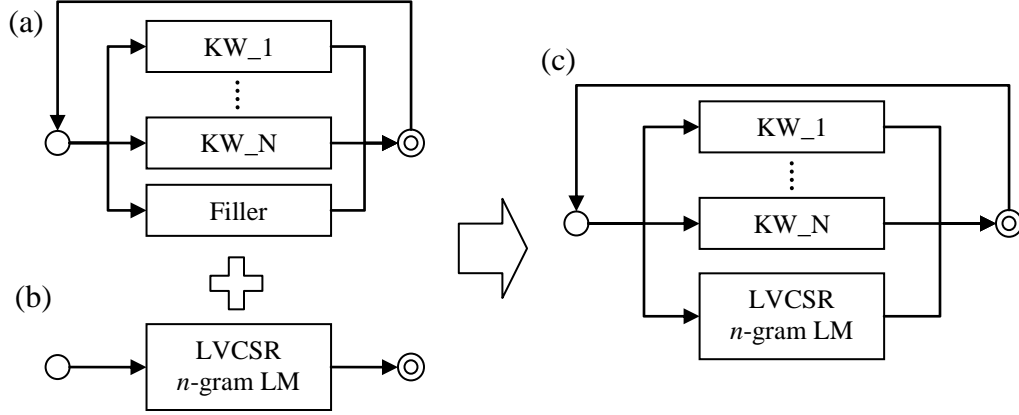


Figure 8. Illustrations of (a) the grammar of classic keyword-filler based KWS, (b) the n -gram LM based grammar used by LVCSR-based KWS, and (c) the proposed keyword-aware grammar, which combines the grammars used in the two KWS frameworks

4.5 Exact Realization of the Keyword-Aware Grammar

The keyword-aware grammar can be realized in a weighted finite-state transducer (WFST) based LVCSR system [112] by directly inserting additional keyword paths to the n -gram based grammar WFSAs [113]. Since word sequences of keywords can be present in both paths for the language model and keywords, extra care is required to make sure the resulting grammar-level WFST is deterministic and can be minimized.

In this section, a brief introduction of the WFSAs representation of n -gram LMs [112, 113] is first given. Then we will show how a deterministic KW-aware grammar WFSAs can be realized by modifying the n -gram LM WFSAs. For WFSAs formulations, annotations in [113] are adopted.

4.5.1 Preliminary

Definition 1. A system $(\mathbb{K}, \oplus, \otimes, \bar{0}, \bar{1})$ is a semiring [114] if: $(\mathbb{K}, \oplus, \bar{0})$ is a commutative monoid with identity element $\bar{0}$; $(\mathbb{K}, \otimes, \bar{1})$ is a monoid with identity element $\bar{1}$; \otimes distributes over \oplus ; and $\bar{0}$ is an annihilator for \otimes : for all $a \in \mathbb{K}$, $a \otimes \bar{0} = \bar{0} \otimes a = \bar{0}$.

In this dissertation, the log semiring $\mathcal{L} = (\mathbb{R} \cup \{\infty\}, \oplus_{\log}, +, \infty, 0)$ is used [112]. Note the log semiring is an isomorphism of the probability semiring $(\mathbb{R}_+, +, \times, 0, 1)$ via a log morphism with, for all $a, b \in \mathbb{R} \cup \{\infty\}$:

$$a \oplus_{\log} b = -\log(\exp(-a) + \exp(-b))$$

and we follow the convention that $\exp(-\infty)=0$ and $-\log(0)=\infty$.

Definition 2 A weighted finite-state automaton A over a semiring \mathbb{K} is an 7-tuple $A=(\Sigma, Q, I, F, E, \lambda, \rho)$ where: Σ is the finite alphabet of the automaton; Q is a finite set of states; $I \subseteq Q$ the set of initial states; $F \subseteq Q$ the set of final state; $E \subseteq Q \times (\Sigma \cup \{\infty\}) \times \mathbb{K} \times Q$ a finite set of transitions; $\lambda: I \rightarrow \mathbb{K}$ the initial weight function; and $\rho: F \rightarrow \mathbb{K}$ the final weight function mapping F to \mathbb{K} .

Given a transition $e \in E$, we denote its label $l[e]$, its origin or previous state $p[e]$ and its destination state or next state $n[e]$, its weight $w[e]$, namely $e=(p[e], l[e], w[e], n[e])$. Given a state $q \in Q$, we denote $E[q]$ the set of transitions leaving q .

A path $\pi=e_1 \dots e_L$ is an element of E^* with consecutive transitions: $n[e_{i-1}] = p[e_i]$, $i=2, \dots, L$. We extend n and p to paths by setting $n[\pi] = n[e_L]$ and $p[\pi] = p[e_1]$. The labeling function l and the weight function w can also be extended to paths by defining the label of a path as a concatenation of the labels of its constituent transitions, and the weight of the path as the \otimes -product of the weights of its constituent transitions: $l[\pi]=l[e_1] \dots l[e_L]$, $w[\pi]=w[e_1] \otimes \dots \otimes w[e_L]$. The path π can therefore be represented by $(p[\pi], l[\pi], w[\pi], n[\pi])$. We also define $states[\pi]$ and $transitions[\pi]$ being the set of states and transitions on the path π .

4.5.2 Representation of n -gram LMs with WFSA

In a WFSA representation of an n -gram LM over the log semiring, each state in the WFSA represents an n -gram conditioning history h_i , e.g. $w_{i-2}w_{i-1}$. Each transitions leaving the state represent a word w_i with a weight $-\log(P(w_i|h_i))$ or a backoff transition to a lower-order conditioning history state [113]. A string accepted by the WFSA has a single path through the automaton, and the weight of the string is the sum of the transition weights in that path in a form of negative log probability.

Given a finite set of state, Q , in an n -gram WFSA and a string $k=w_1 \dots w_L$, We denote $hist_of[k, Q]$ as the state in Q encoding the conditioning history that matches the end of the string k with the highest order.

4.5.3 Realization of KW-aware grammar with WFSAs

Suppose the set of keywords can be categorized into c classes, and K_i ($i=1\sim c$) is a list of keywords in class i with the list size $|K_i|$ and the constant prior κ_i for the class, given an n -gram LM WFSAs, $A=(\Sigma, Q, I, F, E, \lambda, \rho)$, a KW-aware grammar WFSAs, A' , can be realized by the pseudo code presented in Figure 9. The algorithm consists of four steps: (i) add disambiguation symbols to the alphabet of WFSAs, (ii) add keyword initial states, (iii) add keyword paths, and (iv) normalization to make the final KW-aware WFSAs stochastic. Note that, in the KW-aware grammar WFSAs, disambiguation symbols ($\#k_1, \dots, \#k_c$) are utilized on any transition from states in the n -gram WFSAs to the keyword initial states (line 7). The resulting WFSAs is therefore deterministic and can be optimized offline. In this dissertation, all keywords are assumed in the same class, and a single keyword initial state and κ are used.

```

Create KW-aware grammar WFSAs ( $A, K_{1\sim c}, \kappa_{1\sim c}$ )
1   $A' \leftarrow A$ 
2   $\Sigma' \leftarrow \Sigma \cup \{\#k_1, \dots, \#k_c\}$  // 1. Add disambiguation symbols
3  for  $i$  in 1 to  $c$  do:
4     $q_{ki} \leftarrow (K\_Init_i)$ 
5     $Q' \leftarrow Q' \cup \{q_{ki}\}$  // 2. Add keyword initial states
6    for  $q \in Q$  do:
7       $E' \leftarrow E' \cup \{(q, \#k_i, -\log(|K_i| \cdot \kappa_i), q_{ki})\}$ 
8    for  $k \in K_i$  do: // 3. Add keyword paths
9       $\pi \leftarrow (q_{ki}, k, \log(|K_i|), hist\_of[k, Q])$ 
10      $Q' \leftarrow Q' \cup states(\pi)$ 
11      $E' \leftarrow E' \cup transitions(\pi)$ 
12 for  $q' \in Q'$  do: // 4. Normalization
13    $norm \leftarrow \bigoplus_{e' \in E[q']} w[e']$ 
14   for  $e' \in E[q']$  do:
15      $e' \leftarrow (q', l[e'], w[e'] - norm, n[e'])$ 
16 return  $A'$ 

```

Figure 9. Pseudo code for the KW-aware grammar WFSAs realization

4.6 Approximation of the Keyword-Aware Grammar using n -gram LM

For LVCSR-based KWS systems not built with WFST, the effect of the keyword-aware grammar can be approximated by adjusting the probability of keywords in the n -gram

language models in the systems. Three approximations of the grammar using n -gram LMs are presented in section 4.6.1, 4.6.2, and 4.6.3 respectively.

4.6.1 Keyword-boosted language model

The most straight-forward way to boost the probability of the word sequences of keywords in a language model is adding the keywords to the training text of the language model. Given the training data for the language model and a list with N target keywords, we append each keyword to the training text k times (Figure 10). The resulting training text for the language model will be the original training transcriptions with additional $N \cdot k$ lines of keywords. The parameter k , which indicates the number of times a keyword repeat in the training text, is a parameter to be tuned. We call this a keyword-boosted LM (KW-boosted LM); [115] has explored similar methods and showed it help improve system performance on Cantonese KWS tasks.

Language models trained by this keyword-appended text will have a higher probability for the word sequences of keywords and thus are more sensitive to the predefined keywords even when the original training text contains very little information about them.

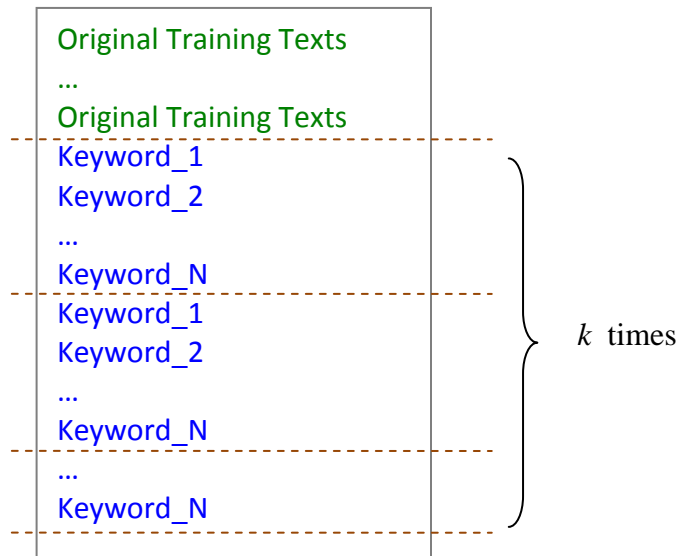


Figure 10. An illustration of the keyword-boosted LM training data.

4.6.2 Keyword language model interpolation

The KW-boosted LM approach adjusts the probabilities of keyword paths to the other paths in the original language model by setting the repetition number k of the keywords in the training text. However, since k can be any positive integer, such an infinite range of possibilities makes it difficult to optimize system performance. The approach also makes the utilization of keyword information much less efficient because the keyword data would be backoff-smoothed with the original training data during the model training process. To alleviate the problems, instead of appending keywords to the original LM training text, we train a keyword language model using keyword text alone and then perform a linear interpolation with the original language model using Eq. (15) (Figure 11). This approximate approach is named keyword language model (KWLM) interpolation.

$$P_{INT_LM}(w|h) = \alpha \cdot P_{KWLM}(w|h) + (1 - \alpha)P_{LM}(w|h) \quad (15)$$

In Eq. (15), the $P_{INT_LM}(w|h)$ is the interpolated probability between the keyword LM and the original LM for the n -gram (h, w) , where h is the history and w is the current word. Note that in the proposed KWLM interpolation, the parameter α , which tunes the weight of keyword LM to the original LM in the final LM, is in a manageable range of $[0,1]$ instead of the open range $[0,\infty)$. In addition, it makes linguistic sense to keep the two text lists separate as they are from intrinsically different sources. Integrating the two text lists via an interpolation weight makes the solution more elegant than the previous approach.

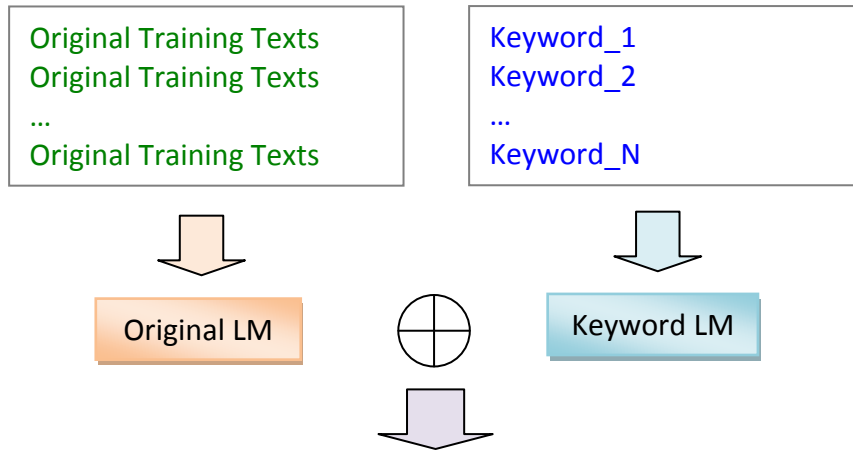


Figure 11. The construction of keyword language model interpolated LM. The final LM is an interpolated LM of the original LM and the keyword LM.

4.6.3 Context-simulated keyword language models

In the keyword language model training text, each keyword is treated as an individual sentence as shown in Figure 11. This makes the keyword language model overemphasize the probability of the keyword appearing at the beginning and the end of a sentence. To remove this bias, in the context-simulated keyword language model training text we put context terms before and after each keyword to simulate the situation where keywords are embedded in real sentences. The context terms can be selected as bigrams or trigrams with high probabilities in the original language model. Note that instead of using real context terms for a keyword, simulated context terms are used here because in most cases context information of the keyword is not available. Figure 12 illustrates the training text for CS-KWLM. Once the context-simulated keyword language model is trained, we can use Eq. (15) to obtain another interpolated language model, which approximates the proposed keyword-aware grammar for KWS.

Simulated-Context-1	Keyword_1	Simulated-Context-1
Simulated-Context-1	Keyword_1	Simulated-Context-2
	...	
Simulated-Context-2	Keyword_1	Simulated-Context-1
Simulated-Context-2	Keyword_1	Simulated-Context-2
	...	
Simulated-Context-M	Keyword_1	Simulated-Context-M
Simulated-Context-1	Keyword_2	Simulated-Context-1
	...	
Simulated-Context-M	Keyword_2	Simulated-Context-M
	...	
Simulated-Context-1	Keyword_N	Simulated-Context-1
	...	
Simulated-Context-M	Keyword_N	Simulated-Context-M

Figure 12. An illustration of the training text for context-simulated keyword language models (CS-KWLM). Note that instead of using real context terms for a keyword, simulated context terms are used here because in most case context information of the keyword is not available (especially in the domain mismatch conditions). The simulated context terms in this research are selected as top M bigrams in the original LM training data. The value of M is chosen to allow the text file size stay in a manageable scale (e.g., less than 2 Gb).

4.7 Experiments

4.7.1 Experimental setup

Experiments were conducted on the IARPA Babel OpenKWS13 (Vietnamese) [92] and OpenKWS14 (Tamil) [93] limited language pack (LLP) tasks. In both tasks only 10-hour transcribed audio were used for system training. The data are conversational speech between two parties over a telephone channel, which can be landline, cellphone, or phones embedded in vehicles, with the sampling rate set at 8 kHz. For system tuning, a 2-hour subset of the task development set (denoted as *dev2h* in this study) was used to speed up the system development process.

For both OpenKWS13 and OpenKWS14 systems, the 15-hour evaluation part 1 data (released as *evalpart1* by NIST) were used for testing. The evaluation keyword lists contain 4,065 and 5,576 phrases with out-of-vocabulary words not appearing in the training set for the two tasks respectively. The performance of keyword search was evaluated by measures including the number of missing keywords, Figure of Merit (FOM), and the actual term weighted value (ATWV) [46], which is a measure that takes

both detection miss and false alarm errors into account. A system with perfect detection performance would have ATWV of 1. Note that the IARPA Babel program set ATWV=0.3 as the benchmark for KWS system performance.

All keyword search systems were LVCSR-based with hybrid DNN-HMM acoustic models built with the Kaldi toolkit [116]. Readers can easily reproduce all baseline results presented in this paper by running the Babel recipe provided in the Kaldi toolkit. The DNNs were trained with sMBR sequential training [117]. The acoustic features were bottleneck features appended with fMLLR features, while the bottleneck features were built on top of a concatenation of PLP, fundamental frequency (F0) features, and for the Vietnamese systems fundamental frequency variation (FFV) features were used in the concatenation as well. A grapheme-to-phoneme (G2P) approach [83] is used to estimate the pronunciation for OOV words appearing in the evaluation keywords. The estimated pronunciations were then merged into the original lexicon provided by IARPA to form the system lexicon. The baseline language models are 3-gram LMs trained with the 10-hour training transcriptions only.

For each task, all the KWS systems shared the same acoustic model and lexicon. However, they are different in the decoding grammar. The baseline system is a system using the original trigram LM and is denoted as “ n -gram baseline”. The second system is a system with the exact realization of the proposed keyword-aware grammar and is marked as “KW-aware grammar”. Three approximate systems using the three approximate realizations (i.e., keyword-boosted LM, keyword language model interpolated LM, and context-simulated keyword language model interpolated LM) are denoted as “KW-boosted LM”, “KWLM Interpolation”, and “CS-KWLM Interpolation” respectively. All the system parameters κ , k , and α are tuned with the *dev2h* data for each task.

4.7.2 LM training data analysis for the Vietnamese LLP task

The Vietnamese LLP baseline language model is a trigram LM trained with the transcriptions of the 10-hour training text. Since the amount of the training data was very limited, lots of keywords and key phrases were unseen to the language model and therefore they resulted in very low estimated probabilities in the decoding phase. Table 6

shows how serious the problem is. In the first row of Table 6, there were 3,275 out of the 4,065 keywords unseen in the training text, namely n -grams used by these terms ended up with low probabilities in the baseline language model. Moreover, there were 619 keywords consisting of out-of-vocabulary words, which means that the baseline language model will give these terms nearly zero in back-off probability and make them easily pruned away during decoding. Therefore, it is not surprising that a substantial number of keywords will be missed if the baseline language model was used for decoding. This is why we need the keyword-aware language models to alleviate the problem.

Table 6. Numbers of keywords unseen in the training data and keywords containing OOV words among the given list of 4,065 keywords in the Vietnamese LLP task.

	#keywords	percentage in the keyword list
keywords unseen in training data	3275	80.6 %
keywords containing OOV words	619	15.2 %

4.7.3 Performance of approximate approaches

The first experiment compares the three approximate approaches proposed in this chapter. Table 7 shows the performance of the three approximate systems on the *dev2h* data in the Vietnamese LLP task. Note that the Babel OpenKWS13 Vietnamese data is relatively difficult when compared to most of the commonly used datasets. Despite using the state-of-the-art LVCSR techniques, the Kaldi baseline system still had a very high word error rate (WER) and could only achieve 0.2265 of ATWV (first row in Table 7). For the KW-boosted LM system, though being a very simple approximate approach for the KW-aware grammar, the method brought a 26% relative gain on the ATWV already at $k=5$. The slight WER improvement over the baseline system is due to the additional n -gram information provided by the extra appended keyword text in the LM training data. For KWLM and CS-KWLM systems, after tuning the best α in Eq. (15) to be 0.6 for both systems the ATWVs improved to 0.3431 and 0.3546, respectively.

Table 7. WER and ATWV comparison of Vietnamese LLP systems with different language models on the Vietnamese *dev2h* data.

Vietnamese LLP Systems [<i>dev2h</i>]		WER (%)	ATWV
<i>n</i> -gram baseline		62.5	0.2265
KW-aware LM	KW-boosted LM ($k=5$)	62.3	0.2853
	KWLM Interpolation ($\alpha=0.6$)	64.2	0.3431
	CS-KWLM Interpolation ($\alpha=0.6$)	63.5	0.3546

Table 8 shows the experiment results on the *evalpart1* data. A very similar trend of system performance on the *dev2h* data is observed. The ATWV of the Kaldi baseline was only 0.2093, which is still far below the IARPA Babel program's minimal requirement. The KW-boosted LM significantly reduced this performance gap and reached the ATWV of 0.2715. By adopting KWLM and CS-KWLM interpolation methods, the systems successfully achieved the goal of the program. For the CS-KWLM system, which had the best ATWV performance, the overall ATWV improvement over the baseline system is 0.1194 absolute and more than 50% relative. Note that optimizing system ATWV over the evaluation keywords using the proposed methods does not hurt WER performance of the underlying LVCSR systems significantly. In other words, the lattices generated by the proposed systems still have similarities for non-keyword terms to the lattices generated by the baseline system. Therefore, even when adding new keywords which are not in the current list for evaluation, in the worst case, the proposed system would have a similar performance to the baseline system for those new keywords.

Table 8. WER and ATWV performance of Vietnamese LLP systems with different language models on the Vietnamese *evalpart1* data.

Vietnamese LLP Systems [<i>evalpart1</i>]		WER (%)	ATWV
<i>n</i> -gram baseline		65.0	0.2093
KW-aware LM	KW-boosted LM ($k=5$)	65.1	0.2715
	KWLM Interpolation ($\alpha=0.6$)	66.7	0.3186
	CS-KWLM Interpolation ($\alpha=0.6$)	66.0	0.3287

4.7.4 Comparison of KWLM and CS-KWLM interpolation

The major difference between KWLM and CS-KWLM is the introduction of the simulated context information derived from the original LM. In Figure 13, the ATWVs of the two systems with different α on the *dev2h* data were compared. For α smaller than 0.6, the CS-KWLM system outperformed the KWLM system by more than 0.02 ATWV consistently. This demonstrated that the context information provides the CS-KWLM interpolated LM a better connectivity between the keyword LM and the original LM. In other words, it makes the CS-KWLM approach better represents the keyword-aware grammars.

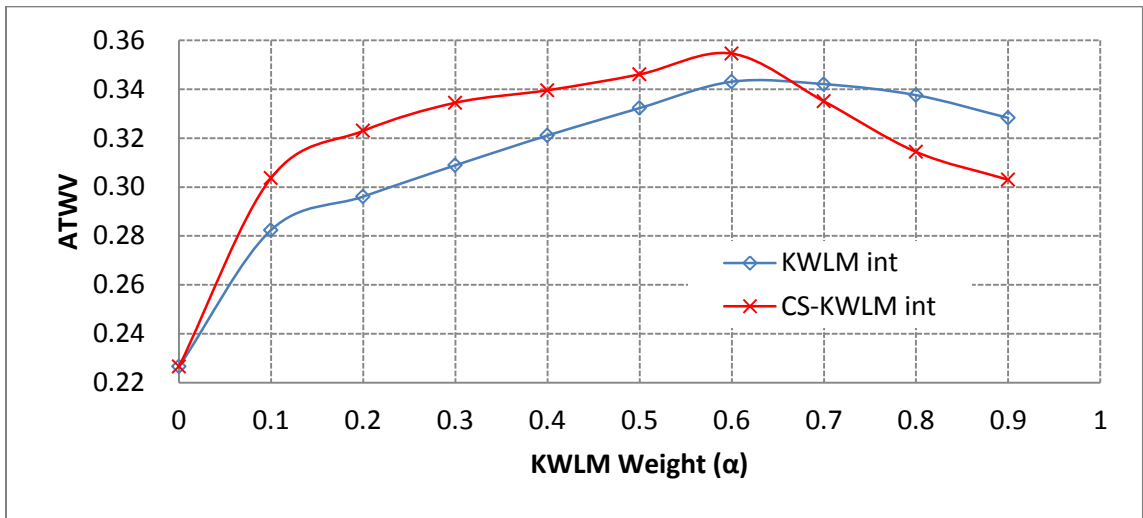


Figure 13. ATWV on *dev2h* with different keyword LM weights α for both KWLM and CS-KWLM interpolation methods in the Vietnamese LLP task.

Both systems reached the highest ATWV value when $\alpha=0.6$. The ATWV of the CS-KWLM system started dropping fast when α gets larger than 0.6 because of the increased false alarms. However, as long as α is tuned with a representative development data, the risk of such increase in false alarms is small since the optimal α is quite consistent as observed in Table 7 and Table 8.

4.7.5 Comparison of the exact and approximate realizations

In section 4.7.3, it has been shown that the CS-KWLM interpolation is the best approximate approach for the keyword-aware grammar. Thus, in the following experiments, the approximate approach refers to the CS-KWLM interpolation. In this

section, comparisons between the approximate approach and the exact realization of the proposed keyword-aware language modeling framework are conducted.

4.7.5.1 Grammar WFSAs comparison

Table 9 compares the grammar WFSAs used in the baseline, the KW-aware grammar, and the CS-KWLM interpolation systems in the Vietnamese LLP task. Carrying additional keyword information, both KW-aware grammar and CS-KWLM Interpolation systems had larger grammar WFSAs than the baseline in terms of arc number, state number, and file size. However, as the size of the CS-KWLM based grammar WFA was 10 times larger than the baseline due to the great amount of additional keyword n -gram states, the size of the KW-aware grammar remained in a similar scale of the original n -gram WFA. It is clear that the exact realization provides more compact grammar WFA than the approximate approach.

Table 9. Comparison of the three grammar WFSAs in terms of arc number, state number, and file size.

Vietnamese grammars		# arcs	# states	File size
n -gram baseline		38,713	17,616	812 Kb
KW-aware framework	KW-aware grammar (global $\kappa=0.00005$)	66,913	24,215	1.3 Mb
	CS-KWLM Interpolation ($\alpha=0.6$)	381,461	165,063	7.8 Mb

4.7.5.2 Performance on the Vietnamese LLP task

Performance of the three systems (i.e., the baseline, the KW-aware grammar, and the CS-KWLM interpolation systems) on the Vietnamese *evalpart1* data is compared in Table 10. For KWS, the baseline system had 2,562 missing keywords and was with ATWV of 0.2098. By using the keyword-aware framework, both CS-KWLM and KW-aware grammar systems significantly reduced the number of missed keywords by 40%. The remarkable miss reductions also reflected on the improvement of ATWV and FOM of the systems. The ATWV of the KW-aware grammar system achieved 0.3224, which is about

a 53.7% relative improvement over the baseline. For the CS-KWLM system, the ATWV is even better and reached 0.3287. For the FOM performance, KW-aware grammar system is slightly better than the CS-KWLM system and attained 57.82%; the latter reached 55.26%. Both systems again obviously outperformed the baseline system, which had only 20.50% of FOM. The result shows that two realizations of the proposed language modeling framework provide similar results, and both outperform the baseline significantly, regardless of the performance measures.

Table 10. Performance of the n -gram baseline and the two realizations of the proposed keyword-aware framework on the Vietnamese LLP *evalpart1* data

Vietnamese LLP [<i>evalpart1</i>]		#Miss	FOM (%)	ATWV
n -gram baseline		2,562	20.40	0.2098
KW-aware framework	KW-aware grammar (global $\kappa=0.00005$)	1,589	57.82	0.3224
	CS-KWLM Interpolated LM ($\alpha=0.6$)	1,651	55.26	0.3287

4.7.5.3 Performance on the Tamil LLP task

A similar trend was found in the Tamil LLP task. In Table 11, the baseline system had 3,663 missed keywords and ATWV of 0.2128. Again, the KW-aware framework reduced about one third of the miss in the baseline. And a relative 46% ATWV improvement was also observed on the KW-aware grammar and CS-KWLM systems.

Table 11. Performance of the n -gram baseline and the two realizations of the proposed keyword-aware framework on the Tamil LLP *evalpart1* data

Tamil LLP [<i>evalpart1</i>]		# Miss	ATWV
n -gram baseline		3,663	0.2128
KW-aware framework	KW-aware grammar (global $\kappa=0.0000347$)	2,830	0.3102
	CS-KWLM Interpolation ($\alpha=0.3$)	2,689	0.3160

4.7.6 ATWV analysis for keywords of different lengths

Section 4.7.5 shows that the exact and the approximate systems have similar overall performance. However, it is more interesting to know if they have different characteristics for different types of keywords. To further study the differences between the exact realization and the approximate approach, ATWVs of the two systems on keywords of different lengths were compared. Figure 14 displays the ATWVs of keywords with different lengths for the n -gram baseline, CS-KWLM Interpolation and the KW-aware grammar systems in the Tamil LLP task. In general, a KWS system has better detection performance for longer keywords because more acoustic context information is available for the system to make correct decisions. However, because of the misses caused by the underestimated keyword priors, the ATWVs of the n -gram baseline system in Figure 14 only increased slowly with the increase of keyword lengths and dropped rapidly when keyword length $L > 3$. By alleviating the underestimation problem, both CS-KWLM interpolation and KW-aware grammar systems significantly outperformed the baseline system, especially for long keywords.

If we further compare the CS-KWLM interpolation and KW-aware grammar systems, it is clear that the two systems were different in their performance with the keyword lengths. For long keyword ($L > 2$), the KW-aware grammar significantly outperformed the CS-KWLM interpolation system. This is because the CS-KWLM interpolation approach not only boosted probabilities of keywords but also other word sequences with keyword n -grams presented. The boosting effect may therefore being reduced relatively especially for multi-word keywords because more n -grams are presented in the queries. The standalone keyword paths in KW-aware grammar to some extent alleviate this problem caused by n -gram sharing in the n -gram LM, thus it has better performance for long keywords. On the other hand, the CS-KWLM system has slightly better performance than KW-aware grammar for $L \leq 2$ because keyword priors in the KW-aware grammar system were restricted to a global κ while for the CS-KWLM system such restriction did not exist which allowed the prior estimation for each keyword being closer to its ground truth. Since keywords with $L \leq 2$ are the majority in the evaluation list, the overall ATWV of CS-KWLM system in Figure 14 is slightly better than KW-aware grammar system.

However, the result suggests that the two realizations are complementary and should be considered in different cases.

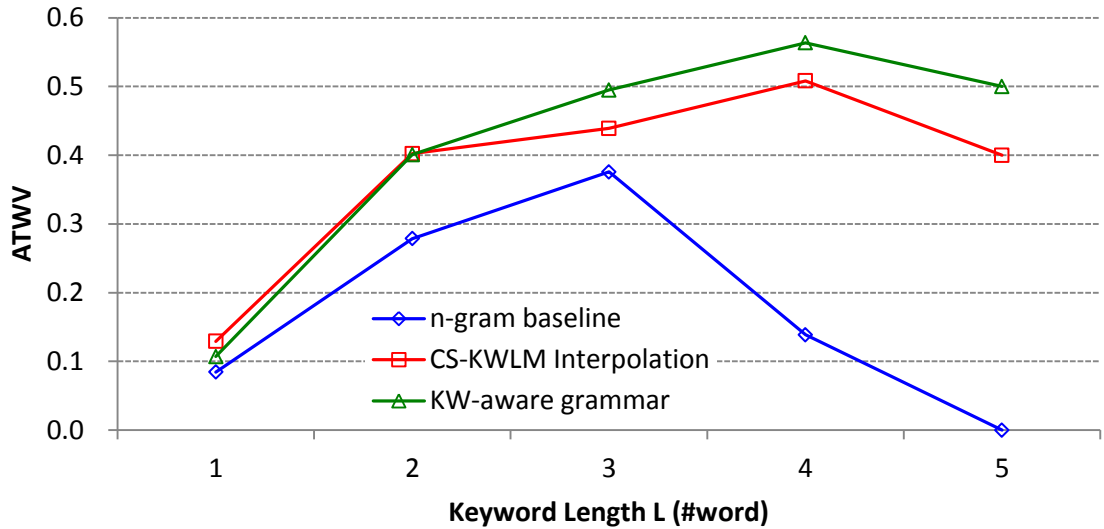


Figure 14. ATWV of keywords with different lengths for the three systems on *evalpart1* data in the Tamil LLP task. The ATWV drops at $L=5$ in all the systems are due to miss errors caused by underestimated keyword priors. Also, since there are only 3 keywords at $L=5$, the ATWV might not be representative for all the five-word keywords.

Figure 15 displays the ATWV curves for the n -gram baseline and the KW-aware systems in the Vietnamese LLP task. The performance difference between the CS-KWLM interpolation and KW-aware grammar system become much smaller in Vietnamese, however, both of them still significantly outperformed the n -gram baseline system. Similar to the results observed in Tamil LLP task, the ATWV curves for the KW-aware systems had a clear improvement over the baseline, and the improvement is especially larger for longer keywords. For example, the KW-aware systems successfully detected two out of the three five-word keyword, "đăng ký mùa hè xanh", in the evaluation data without any false alarm, while the n -gram baseline system missed all of them. The KW-aware systems showed a similar ATWV to the n -gram baseline on single-word keywords because priors of them were not as seriously underestimated because of LM smoothing [118].

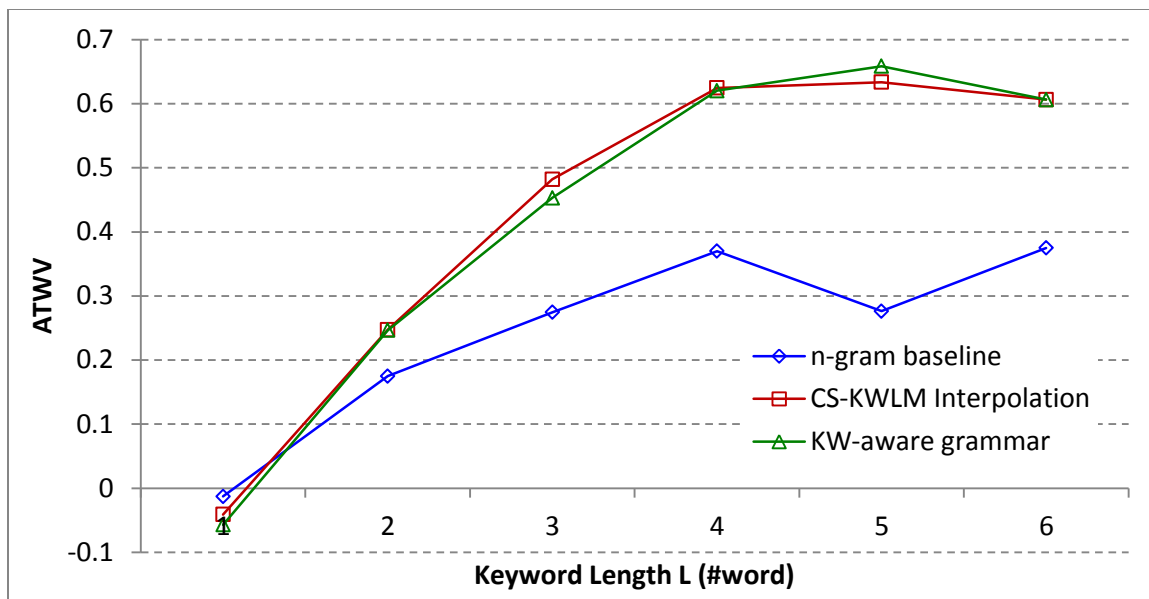


Figure 15. ATWV for keywords with different length, L , of the baseline and KW-aware systems on the *evalpart1* data in the Vietnamese LLP task.

4.7.7 ATWV analysis for IV and OOV keywords

In Table 12, ATWVs of in-vocabulary (IV) and out-of-vocabulary (OOV) keywords for the baseline and the CS-KWLM systems were compared. Note that for OOV keywords, which are keywords containing OOV words in them, the baseline had a very low ATWV because those keywords are represented with nearly zero probabilities in the language model, causing a high miss error rate. By using the CS-KWLM Interpolation method to alleviate this problem, ATWV for the OOV queries achieved 0.2343, which is a 154% relative improvement. For the IV queries, the CS-KWLM method also brought a relative ATWV improvement of 49%. Therefore, the proposed approach is effective for keywords in both categories, especially for OOV keywords.

Table 12. ATWV performance of all, in-vocabulary (IV), and out-of-vocabulary (OOV) keywords for the baseline LM and CS-KWLM Interpolation systems on the *evalpart1* data.

Vietnamese LLP [<i>evalpart1</i>]	all	IV	OOV
Baseline LM	0.2093	0.2338	0.0924 ¹³
CS-KWLM Interpolation($\alpha=0.6$)	0.3287	0.3485	0.2343

4.7.8 ATWV analysis for seen and unseen keywords

When dealing with topics not well-observed, data mismatch is assumed to be a major cause of prior probability underestimation in n -gram training. In this experiment, ATWVs of seen and unseen keywords in the LM training set in the Vietnamese LLP task were compared. The unseen keywords can be viewed as keywords whose topics were not covered by the training data. In other words, even IV keywords might still be unseen to the system LM. Because there were only 10-hour transcriptions available for LM training in the Vietnamese LLP task, 3,275 out of the 4,065 keywords were unseen to the baseline n -gram LM. In other words, more than three quarters of the evaluation keywords suffered the mismatch issue in the n -gram LM.

In Table 13, for both keyword groups the proposed KW-aware systems showed increased ATWVs in both cases. The improvement for the seen keywords showed that their priors might still be underestimated even for keywords already appearing in the LM training set. For unseen keywords, the improvement is especially significant – about a 0.15 absolute (from 0.2 to 0.35, 75% relative) ATWV increase over the baseline. The significant performance improvement in the unseen keywords is because most of the unseen keywords are long keywords, which ideally should have better detection results than short keywords because more acoustic cues are available to the systems to make the right decision as described in section 4.7.6. After alleviating the prior underestimation problem, it is clear that the ATWV of unseen keywords (usually long keywords) becomes

¹³ It was not zero because the baseline also used the G2P lexicon in Section 5 for a fair comparison with the KW-aware systems.

higher than the ATWV of seen keywords (usually short keywords). The trend is therefore closer to the ideal case.

Note that the significant performance improvement for unseen keyword also confirms the statement in section 4.3:

The n -gram LM could seriously underestimate the prior probability of the multi-word keywords which are unseen in the training data because of the “incorrect independent event assumption” made by the baseline n -gram LM with back-off smoothing.

With the proposed keyword-aware framework, it is obvious the underestimation problem was effectively diminished.

Table 13. ATWV for seen and unseen keywords in the Vietnamese LLP task.

Vietnamese LLP [<i>evalpart1</i>]	all	seen	unseen
n -gram baseline	0.2093	0.2350	0.1985
KW-aware grammar (global $\kappa=0.00005$)	0.3224	0.2567	0.3519
CS-KWLM Interpolation ($\alpha=0.6$)	0.3287	0.2648	0.3574

4.7.9 Comparison of estimated keyword priors

It is well known that training a good n -gram LM is difficult. When the domain of LM training data mismatches test conditions in KWS, n -gram LMs usually underestimate prior probabilities of multi-word keywords unseen in the LM training data. The underestimation can easily cause missed detections of the keywords. Figure 16 displays the average log priors of keywords with different lengths estimated by the n -gram baseline, CS-KWLM Interpolation, and KW-aware grammar systems on the Vietnamese *evalpart1* data. For each keyword appearing in the *evalpart1* data, the ground-truth of its prior probability was estimated by dividing its count with the total word count in the data set. In Figure 16, the ground-truths of keyword-prior probabilities in the *evalpart1* data remained in the range of -10 to -12 (in natural log) for all the keyword lengths.

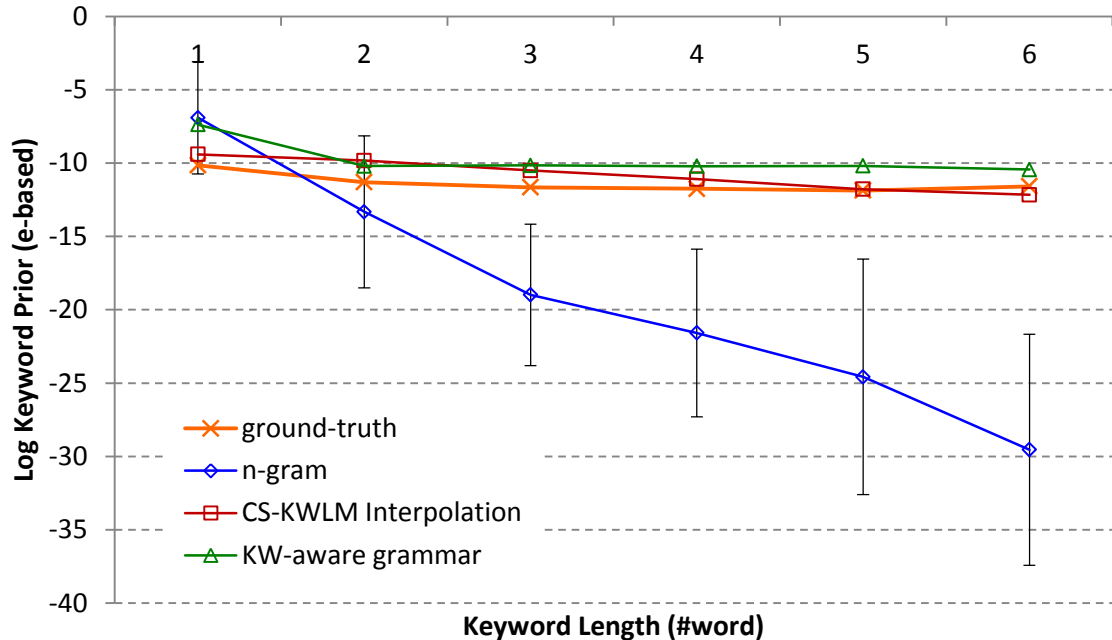


Figure 16. Prior estimations of keywords with different lengths on the Vietnamese *evalpart1* data.

The estimated keyword priors of the three systems were evaluated by searching the best keyword path in the system decoding grammar WFSAs for each keyword. The weight of the path was used as the estimated prior for the keyword in the systems. In Figure 16, though the estimated keyword priors of the *n*-gram baseline system were quite close to the real values for single-word keywords, the priors were seriously underestimated for longer keywords. Note the curve of the *n*-gram system monotonically decreased as the keyword length increased. For example, the system underestimated 6-word keywords' prior probabilities at the scale of 5×10^8 . The underestimation problem was alleviated by the proposed methods. By adding standalone paths or using LM interpolation to boost probabilities for each keyword, the prior estimations of the KW-aware and CS-KWLM Int systems were very close to the real priors regardless of the number of words in a keyword.

4.7.10 Performance of keyword-aware language model to new keywords

We know that with the knowledge about what keywords the system would be asked to detect, KW-aware systems outperformed the *n*-gram baseline system significantly.

However, it is interesting to know what would happen if we use a KW-aware system enhanced with a specific keyword list to detect another set of keywords. In section 4.4, it is claimed that the system performance for the new keyword set would be very close to the performance of the baseline system. The goal of this experiment is to verify the statement.

In the OpenKWS14 Tamil LLP task, in addition to the evaluation keyword list (denoted as Eval), there are three more keyword lists provided by BBN and IBM. Details of the keyword lists are presented in Table 14. Since for each of the additional keyword list, there were only a very small portion of the keywords overlapped with the evaluation keyword list, the lists can be seen as “another set of keywords” for the KW-aware system enhanced with the evaluation keyword list.

Table 15 shows the ATWVs of the baseline and CS-KWLM interpolation (enhanced with the Eval keyword list) systems on the four keyword lists in the Tamil LLP task. Since the CS-KWLM interpolation system was enhanced with the Eval keyword list, the system had significant ATWV performance improvement for the Eval keyword list (from 0.2128 to 0.3160). However, if we check the ATWVs of the CS-KWLM interpolation system for the other three keyword lists, the system performance is very close to the baseline. The slight improvement in the ATWV values is because of the overlapped keywords in the BBN/IBM keyword lists and the Eval keyword list.

The result confirms the statement made in section 4.4. It also shows one of the greatest advantages of the proposed keyword-aware language modeling framework. On the one hand, it allows us to significantly improve detection performance of a KWS system for keywords we know beforehand; on the other hand, for those keywords we do not know in advance, their detection performance would simply fall back to the performance of the n -gram baseline systems. Therefore, it is safe for us to apply the framework to any pre-known keywords without the worries about degrading the system performance for future keywords.

Table 14. The four keyword lists available in the OpenKWS14 Tamil LLP task.

Keyword list	BBN2	IBM931	IBM932	Eval
# keywords	1,290	1,770	2,039	5,576
# OOV keywords	710	605	556	0
# overlapped KW with Eval KWs	154	194	206	5,576

Table 15. ATWV performance of the n -gram baseline and the CS-KWLM interpolation system (enhanced with the Eval keyword list) for the four keyword lists on the Tamil *evalpart1* data

Tamil LLP [<i>evalpart1</i>]	BBN2	IBM931	IBM932	Eval
n -gram baseline	0.1473	0.1023	0.1361	0.2128
CS-KWLM Interpolation for Eval keywords ($\alpha=0.3$)	0.1567	0.1091	0.1465	0.3160

4.7.11 OpenKWS13 Vietnamese full language pack task

The last experiment verifies if the proposed language modeling approach works even when more system training data are available. Table 16 shows the system performance of on the *evalpart1* data in the Vietnamese full language pack (FLP) task. Note that in the Babel Vietnamese FLP task there are 78-hour transcribed audio data, which is about eight times larger than the data amount in the LLP task, for KWS system construction. With more training data for both acoustic and language modeling, the baseline system achieved the program goal with an ATWV of 0.4578. However, the performance could be further improved substantially (20% relative) by adopting the CS-KWLM interpolation method. This result shows that the underestimation problem does not go away by simply increasing the amount LM training data, and the proposed keyword-aware language modeling is an effective solution providing significant performance enhancement irrespective to the amount of system training resources.

Table 16. ATWV performance of the n -gram baseline and CS-KWLM interpolation systems on *evalpart1* data in the Vietnamese FLP task.

Vietnamese FLP [<i>evalpart1</i>]	ATWV
n -gram baseline	0.4578
CS-KWLM Interpolation ($\alpha=0.7$)	0.5486

4.8 Conclusion

In this chapter, a keyword-aware language modeling framework is proposed to alleviate the "prior underestimation problem" for multi-word keywords in LVCSR-based KWS systems. By integrating the decoding grammars used in conventional keyword-filler based KWS into the n -gram LM grammars used in LVCSR-based KWS, the proposed keyword-aware grammar provides better prior estimations for keywords. The keyword flexibility of LVCSR-based KWS is also preserved in the new grammar. The proposed grammar can be either exactly realized with WFSA or approximated by enhanced n -gram LMs. Experimental results on the Babel Vietnamese and Tamil LLP tasks and the Vietnamese FLP task show that the proposed keyword-aware framework is effective in alleviating the prior underestimation problem of n -gram LMs and significantly reduced the missed detection errors in the baseline LVCSR-based KWS systems. The significant performance improvement of the proposed keyword-aware framework over the n -gram baseline has also been shown to be consistent across languages and tasks with different amount of training data.

CHAPTER 5 SYSTEM OPTIMIZATION OBJECTIVES

5.1 Introduction

In addition to the data-efficient training procedures for acoustic and language modeling, utilizing KWS-performance related training objectives is also a possible direction to improve KWS performance. Usually, objectives of AM and LM training are based on the maximum likelihood (ML) criterion which is not directly related to the KWS performance. If AMs and LMs could be optimized with KWS performance-metric-related objective functions, training data can be more effectively exploited for achieving a better KWS performance. Discriminative training (DT), which is a technique allowing training processes to optimize desired objective functions, can therefore be employed to enhance KWS.

In this chapter, the effect of using discriminative training objectives for both acoustic and language models in KWS systems is studied. The chapter starts with a keyword-aware discriminative acoustic modeling for DNN AMs in Section 5.2. It suggests that the DNN AMs could have a better performance on keyword detection by revising the conventional DT objective functions with information about keywords. Section 5.3 discusses the idea of using discriminative language modeling approaches to enhance the keyword-aware LMs. Experimental results on the Babel Vietnamese limited language pack task for both discriminative acoustic and language modeling are presented in Section 5.4. Section 5.5 gives the conclusion.

5.2 Keyword-Aware Discriminative Acoustic Modeling

Discriminative acoustic modeling has been shown to be efficacious in LVCSR tasks [6-11]. Many discriminative training objectives for AMs, e.g., MCE (Minimum Classification Error [6]), MMI (Maximum Mutual Information [7, 8]), bMMI (boosted Maximum Mutual Information [9]), MPE (Minimum Phone Error [8, 10]), MWE (Minimum Word Error [8]), sMBR (state-level Minimum Bayes Risk [11]), have been proposed for LVCSR tasks. Since these objectives were designed for ASR tasks, most of them are focusing on word error rate (WER) reduction. However, in KWS tasks, systems

only care about whether preselected keywords are detected or not. For words unrelated to the target keywords, their recognition results are not as critical. To address this objective difference in KWS tasks, in [70], a non-uniform MCE training method is proposed for conventional GMM-HMM systems by putting more weight on the error cost function of keyword data. Inspired by the work in [70], in this dissertation a keyword-boosted sMBR training objective [71], which is a modified objective function of sMBR training, for DNN AMs in KWS systems is proposed. The reason of selecting DNN AMs with sMBR training is because sMBR training has been shown to outperform other discriminative training criteria for deep neural network (DNN) AMs [117].

In Sections 5.2.1 and 5.2.2, the sMBR training objective and the way to use the objective function to train DNN AMs will first be reviewed. Section 5.2.3 then presents the proposed keyword-boosted sMBR objective. Experimental results for keyword-boosted sMBR acoustic modeling are shown in Section 5.4.2.

5.2.1 State-level minimum Bayes risk (sMBR) training objective

The objective function of sMBR training is designed to minimize the expected state-level errors over the whole set of training utterances [117, 119]:

$$F_{sMBR}(\Lambda) = \sum_{u=1}^U \frac{\sum_W p(O_u | S_W)^\kappa P(W) A(W, W_u)}{\sum_{W'} p(O_u | S_{W'})^\kappa P(W')}, \quad (16)$$

where Λ represents acoustic model parameters, κ is an acoustic scaling factor, $O_u = \{\mathbf{o}_{u1}, \dots, \mathbf{o}_{uT_u}\}$ is the observation sequence of a training utterance u , S_W and $S_{W'}$ are the HMM state sequences of the hypothesized word sequence W and W' , respectively, W_u is the reference word sequence for the sentence u , and $A(W, W_u)$ is the raw state accuracy which could be expressed as

$$A(W, W_u) = \sum_{t=1}^{T_u} \text{StateAcc}(s(t), s_u(t)). \quad (17)$$

In Eq. (17), T_u is the number of frames in the utterance u , $s(t)$ and $s_u(t)$ are corresponding HMM states of word sequences W and W_u at time t , respectively. The $\text{StateAcc}(s, s_u)$ is defined as follows:

$$\text{StateAcc}(s, s_u) = \begin{cases} 1 & \text{if correct state, namely } s = s_u \\ 0 & \text{if incorrect state, namely } s \neq s_u \end{cases}. \quad (18)$$

5.2.2 Training DNN-HMM hybrid systems with sMBR criterion

DNN-HMM hybrid systems use DNNs in replace of GMMs to provide log-likelihood for the HMM states in an LVCSR system. For an observation out corresponding to time t in utterance u , a pseudo log-likelihood of state s provided by DNN AM is

$$\log p(\mathbf{o}_{ut} | s) = \log y_{ut}(s) - \log P(s) + C, \quad ,$$

where $P(s)$ is the prior probability of state s calculated from the training data, C is a constant referring to $P(\mathbf{o}_{ut})$ for the utterance, \mathbf{o}_{ut} ; and $y_{ut}(s)$ is the posterior probability for state s given observation out estimated by DNN and is defined as

$$y_{ut}(s) \equiv P(s | \mathbf{o}_{ut}) = \frac{\exp\{a_{ut}(s)\}}{\sum_{s'} \exp\{a_{ut}(s')\}}, \quad (19)$$

where $a_{ut}(s)$ is the activation at the output layer corresponding to state s .

The DNNs can be trained with sMBR criterion using a gradient decent approach [11]. By differentiating Eq. (16) with respect to $\log p(\mathbf{o}_{ut}|r)$, we get

$$\frac{\partial F_{sMBR}}{\partial \log p(\mathbf{o}_{ut} | r)} = \kappa \cdot \gamma_{ut}(r) [\bar{A}_u(s(t)=r) - \bar{A}_u] = \kappa \cdot \gamma_{ut}^{sMBR}(r), \quad (20)$$

where $\bar{A}_u(s(t)=r)$ is the average state accuracy of all paths in the lattice of sentence u passing through state r at time t , \bar{A}_u is the average state accuracy of all paths in the lattice, and $\gamma_{ut}^{sMBR}(r)$ is a factor as defined in the MPE training method [8]. Using Eq. (20), the gradient for activation function $a_{ut}(s)$ can be derived as [11, 117]:

$$\begin{aligned} \frac{\partial F_{sMBR}}{\partial a_{ut}(s)} &= \sum_r \frac{\partial F_{sMBR}}{\partial \log p(\mathbf{o}_{ut} | r)} \frac{\partial \log p(\mathbf{o}_{ut} | r)}{\partial a_{ut}(s)} \\ &= \frac{\partial F_{sMBR}}{\partial \log p(\mathbf{o}_{ut} | s)} = \kappa \cdot \gamma_{ut}^{sMBR}(s) \end{aligned} \quad (21)$$

The gradient then can be used to update the parameters in the whole network using back-propagation procedures.

5.2.3 Keyword-boosted sMBR training objective

In Eq. (17) and (18), the raw state accuracy computation considers all states as equally important disregarding their source words. However, in a KWS system, accuracies of keyword terms are more important than the rest of the terms in the vocabulary. To adapt the conventional sMBR objective function to the KWS tasks, Eq. (18) is modified with a boosting weight β for keyword terms in the system:

$$\text{StateAcc}(s(t), s_u(t)) = \begin{cases} \beta & \text{if } s(t) = s_u(t) \text{ and } t \in \{\text{KW Seg}\} \\ 1 & \text{if } s(t) = s_u(t) \text{ and } t \notin \{\text{KW Seg}\}, \\ 0 & \text{if } s(t) \neq s_u(t) \end{cases}, \quad (22)$$

where $\{\text{KW Seg}\}$ is a set of time segments for keyword words in the reference sentences. Note the resulting keyword-boosted sMBR has the same computational complexity as the conventional sMBR training since the only difference between them is the weight distribution in the objective function.

5.3 Discriminative Keyword-Aware Language Modeling

Despite the success of KW-aware LM, the global parameter α used in the CS-KWLM interpolation is not yet optimized for all keywords. Experimental results show that the best global α is still too large for some keywords (and causes many false alarms, especially for 1-word keywords), while for some other keywords the value is instead too small to reduce the miss rate. Discriminative language modeling (DLM) [12-16] is a possible solution to alleviate the problem. By fine-tuning the n -gram probabilities provided by the CS-KWLM interpolated LMs using discriminative objective functions, keyword probabilities are no longer bounded by a single parameter. The overall system performance therefore has a chance to be further improved.

A goal of this chapter is to investigate the effect of applying DLM to CS-KWLM interpolation systems in KWS tasks. Note that though DLM is popular in LVCSR tasks, there are few studies reporting the performance of the DLM approaches in KWS tasks. This chapter thus provides a preliminary study of the DLM technique for KWS. Potential issues of DLM in KWS are also revealed.

There are many DLM approaches [12-16] proposed for ASR tasks in the past decades. One of the most cited DLM methods is the discriminative global linear model (GLM) framework [14], which is a rescoring framework for ASR systems. In this dissertation, the framework is therefore adopted for the investigation. Sections 5.3.1 and 5.3.2 briefly review the discriminative GLM framework and the perceptron algorithm for model training (more details about the GLM-based DLM in ASR tasks can be found in [14, 120]). Section 5.3.3 discusses some potential issues of applying DLM to KWS tasks. Experimental results for discriminative keyword-aware language modeling are presented in Section 5.4.3

5.3.1 Global linear models

The global linear model (GLM) considers a task of learning a mapping from inputs $O \in \mathcal{O}$ to outputs $W \in \mathcal{W}$. By considering the following components:

- (1) A function **GEN**() which generate a finite set of candidates $\mathbf{GEN}(O) \subseteq \mathcal{W}$ for each possible input O ,
- (2) A function Φ mapping each $(O, W) \in \mathcal{O} \times \mathcal{W}$ to a **feature vector** $\Phi(O, W) \in \mathbb{R}^D$, and
- (3) A parameter vector $\mathbf{w} \in \mathbb{R}^D$,

the linear model defines the mapping through

$$W^* = \arg \max_{W \in \mathbf{GEN}(O)} \Phi(O, W) \cdot \mathbf{w} \quad , \quad (23)$$

where $\Phi(O, W) \cdot \mathbf{w}$ is the inner product $\sum_d w_d \Phi_d(O, W)$. The objective of the linear model training is to estimate the parameter vector \mathbf{w} using the training data. In the testing phase, the hypothesis W^* that maximizes Eq. (23) is selected as the model output.

The linear model has been used in natural language processing problems, such as part-of-speech tagging and base noun phrase chunking [120]. It can also be used for discriminative language modeling in ASR tasks [14]. In the discriminative language modeling setting, \mathcal{O} is the set of all possible acoustic inputs, $O \in \mathcal{O}$ is the acoustic observation of an input utterance to the ASR system, \mathcal{W} is the set of all possible word

sequences, Σ^* , for some vocabulary Σ , and $\mathbf{GEN}(O) \subseteq \mathcal{W}$ is a set of hypothesized word sequences, represented by a word lattice or N-best sentences, generated by the baseline ASR decoder for utterance O .

For the feature vector, $\Phi(O, W)$, in DLM for ASR tasks, the first feature component, $\Phi_0(O, W)$, is usually set to the log-probability of (O, W) in the lattice produced by the baseline decoder¹⁴. This setting allows the feature vector to include the contributions from the acoustic and the original language model. The parameter w_0 is used to control the weight of the log-probability score, $\Phi_0(O, W)$, in baseline lattice when rescored using the linear model. The remaining features are restricted to be functions over the transcription W alone and they track all 1~ n -grams in W , for example:

$$\Phi_1(O, W) = \text{Number of time "the the of" is seen in } W.$$

Note that since the total number of 1 to n -grams could be very large, practically the feature vector only considers the 1 to n -grams in the oracle and best paths in the lattice and ignores most of the rest 1 to n -grams.

5.3.2 The perceptron algorithm

The perceptron algorithm is a training algorithm for the linear model proposed by Collins in [120]. The algorithm is shown in Figure 17. In the iterative training process, for each training example (O_i, W_i) , the best-scoring hypothesis Z_i is generated with current GLM parameters, \mathbf{w} . If Z_i is different from the reference W_i , then the parameters, \mathbf{w} , would be updated by: i) increasing the weights in \mathbf{w} that correspond to features in W_i with the count of the features, and ii) decreasing the weights that correspond to features in Z_i with the count of the features.

¹⁴ In general, each component, $\Phi_d(O, W)$, in the feature vector can be any function of the acoustic input O and the candidate transcription W . By designing the feature function, we can realize DLM using global linear model.

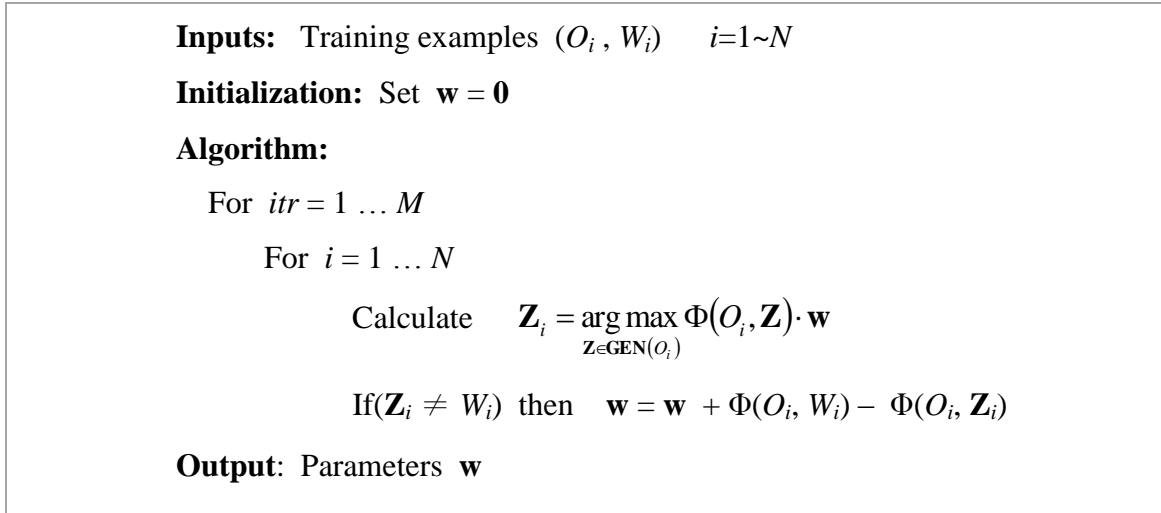


Figure 17. A general form of the perceptron algorithm. The value M , which is the number of iteration over the training data, is chosen by the validation set.

Since the linear model can be implemented using weighted finite-state automata (WFSA) by following the procedure presented in [14], the perceptron algorithm can be modified into the form shown in Figure 18. In Figure 18, L_i is the WFSA representation of the i^{th} training lattice generated by the baseline decoder. D is the n -gram LM WFSA representation¹⁵ of the linear model parameter \mathbf{w} . The rescoring process is done by scaling the original lattice weight with w_0 first and then composing the weight-scaled lattice $w_0 L_i$ with discriminative n -gram LM WFSA, D . In Figure 18, **BestPath**(A) is a process returning the best scoring path in the WFSA A , and **MinErr**(A, r) is a function returning the path with minimal word error when using string r as the reference.

¹⁵ Rigorously speaking, D is not really an “ n -gram LM” WFSA because the arc weights in D are scaled summations of n -gram counts (correspond to features in \mathbf{w}) instead of normalized log probabilities.

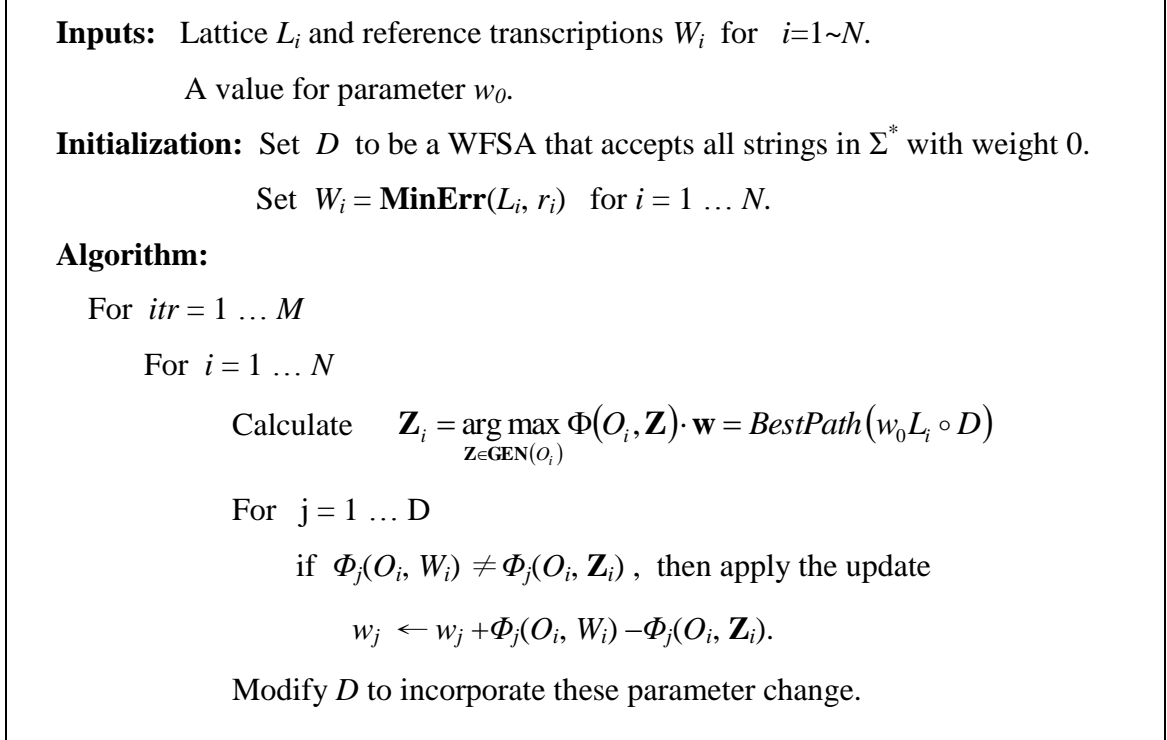


Figure 18. The perceptron algorithm implemented with WFSA. L_i is the WFSA representation of the i^{th} lattice, and D is the n -gram LM WFSA representation of the linear model parameter \mathbf{w} .

In the test phase, the Eq. (23) is still used and is equal to searching the best path in the rescored lattice, as shown in Eq. (24).

$$W^* = \arg \max_{W \in \text{GEN}(O)} \Phi(O, W) \cdot \mathbf{w} = \text{BestPath}(w_0 L \circ D) \quad (24)$$

5.3.3 Issues in the conventional DLM approaches

In [13], several issues regarding the DLM approaches for ASR tasks were discussed. The first problem is the selection of n -gram probability targets for updating when the n -grams do not exist in the original LM. Another issue is whether certain n -gram probabilities should be adjusted if they are involved in but not the cause to the recognition errors (for example, if a sequence "A B C" is misrecognized into "A D C", DLM should adjust the probabilities of $P("B" | "A")$ and $P("D" | "A")$ since they are the direct causes for the error. But it is unclear whether we should adjust the probabilities of $P("C" | "B")$ and $P("C" | "D")$ or not.) The same issues apply to the DLM used in the KWS tasks as well.

However, in KWS tasks there are even more potential problems that need to be considered when adopting the conventional DLM approaches.

A severe problem is that in most of the conventional DLM approaches, the adjusted “ n -gram probabilities” are no longer probabilities. To simplify the training process, most of the DLM methods in ASR tasks use gradient descent without the “sum-to-one” constraint when updating the n -gram probabilities in LMs [12-16]. For each n -gram, the learning step for its probability is a scaled value of its occurrence count in the training data (the reference word sequence and the best path in the lattice). The “ n -gram probabilities” in the updated discriminative language models thus become some sort of “score” rather than probabilities. The setting works fine for ASR tasks, when MAP decoding is used, because we only care about the path with the best score in the lattice. However, when it is necessary to evaluate the posterior probabilities of each word arcs in the lattice, problems emerge. The un-normalized DLM could lead to incorrect posterior probability estimation. Since posterior probability estimation in lattices is important for LVCSR-based KWS systems, using conventional DLM approaches in KWS tasks could lead the systems to provide incorrect posterior probabilities for keywords. The problem is especially critical for the global linear model used in this dissertation because the learning step for an n -gram probability is a sum of the occurrence counts for $1\sim n$ -gram in it [14], which makes the rescored n -gram “scores” even more unbalanced for n -grams with large n .

5.4 Experiments

5.4.1 Experimental setup

Experiments were conducted on the Vietnamese limited language pack (LLP) provided by the IARPA Babel program used in the NIST OpenKWS13 Evaluation [91]. The training set consists of 10 hours of transcribed audio. The audio data is conversational speech between two parties over a telephone channel, which can be landlines, cell phones, or phones embedded in vehicles, with the sampling rate set at 8 kHz. The NIST-provided development set consists of 10 hours of conversational telephone speech;

however, to speed up the tuning process, only a 2-hour subset of the data (denoted as *dev2h*) was used in the experiments.

The 15-hour evaluation part 1 data (denoted as *evalpart1*) was used for testing. Two keyword lists released by NIST for the OpenKWS13 Evaluation were used in the experiments. The evaluation keyword list containing 4,065 phrases, including out-of-vocabulary words not appearing in the training set, was used as the major keyword list for system development and performance evaluation; while the other development keyword list, consisting of 200 phrases, was used for investigating the effect of keyword list sizes to the proposed KW-boosted sMBR acoustic modeling approach. The performance of keyword search was measured by Actual Term Weighted Value (ATWV).

All keyword search systems evaluated were LVCSR-based with hybrid DNN-HMM acoustic models built with the Kaldi toolkit [116]. The baseline DNN AM was trained with conventional cross-entropy objective function. The acoustic features were concatenated features of bottleneck and fMLLR features. The input for the bottleneck DNNs and fMLLR transform is a concatenation of PLP, fundamental frequency (F0), and fundamental frequency variation (FFV) features used in [121]. Keyword-specific threshold normalization [63] was used in all the KWS systems, and the common threshold after normalization is chosen to be 0.5.

For pronunciation dictionaries, in addition to the original LLP lexicon provided by IARPA, a grapheme-to-phoneme (G2P) tool [83] was also used to estimate the pronunciations of out-of-vocabulary (OOV) words in the evaluation keywords. The estimated pronunciations were then merged into the original LLP lexicon to form a G2P lexicon. Since the G2P lexicon is built with keyword information, the lexicon can be seen as a keyword-aware lexicon. The two lexicons (the original lexicon and the G2P lexicon) were then used in two test configurations in the experiments.

Two language models were used in the experiments. The first one is a basic trigram language model, trained by the 10-hour training text and denoted as original LM (*orig LM*) here. The second one is a keyword-aware trigram language model (*CS-KWLM Int*) [122] presented in Chapter 4, which strengthens the probabilities of n -grams used by

keywords in the system to increase the keyword detection rate while keeping the word error rate performance of the underlying LVCSR system nearly intact.

5.4.2 Keyword-aware discriminative acoustic modeling

5.4.2.1 System performance

We first compared the proposed KW-boosted sMBR system with the cross-entropy trained DNN baseline and sMBR trained DNN systems in three different lexicon-LM configurations, namely: (i) original lexicon+LM (denoted as *orig Lex+LM*), (ii) G2P lexicon+LM (denoted as *G2P Lex+LM*), and (iii) G2P lexicon + CS-KWLM interpolated LM (denoted as *G2P Lex + CS-KWLM Int*) corresponding to the results listed in the first, second, and third columns in Table 17 and Table 18. The system ATWVs and WERs were evaluated on the *evalpart1* data with the 4,065 evaluation keywords. Note that the DNN acoustic models of all the KWS systems were trained with the original lexicon and LM in the training phase and then were tested with the three test configurations. System parameters and training iterations were selected with the *dev2h* set.

Table 17 lists the KWS ATWV. One major difference between the first and second configurations, shown in the first and second columns of Table 17, is the system lexicon and the vocabulary size. For the original lexicon, there are 632 OOV words distributed in the 4,065 keywords, and 800 out of the 4,065 keywords cannot be detected because of the lack of complete pronunciations. The G2P lexicon provided the estimated pronunciations of these OOV words and the sMBR trained DNN system has a slight ATWV improvement from 0.2069 to 0.2093. Since the KW-boosted sMBR system put more weight on acoustic modeling for the keywords, it achieved the best ATWV at 0.2104 in the *orig Lex+LM* test configuration. After switching to the G2P lexicon, the KW-boosted sMBR system again achieved the best ATWV among the three systems at 0.2220.

All three KWS systems were significantly improved after switching to the *G2P Lex + CS-KWLM Int* test configuration due to the fact that CS-KWLM provided additional keyword information for detection. Comparing the ATWV improvement from switching the original LM with CS-KWLM interpolated LM and the improvement from replacing the sMBR trained DNN AM with the KW-boosted sMBR trained AM, it is clear that the keyword information is much more helpful for language modeling than for acoustic

modeling. Nevertheless, the effects of keyword-aware language modeling and acoustic modeling are additive. In the *G2P Lex + CS-KWLM Int* test configuration, the KW-boosted sMBR system still outperforms the other two systems and achieves 0.3396 in ATWV. In all the three test configurations, the KW-boosted sMBR system has a 1.7 ~ 6.1% relative ATWV improvement over the sMBR system.

Table 17. ATWV (4,065 keywords) of the Vietnamese LLP systems on the *evalpart1* test set with the three test configurations.

ATWV [<i>evalpart1</i>]	(i) <i>orig Lex + LM</i>	(ii) <i>G2P Lex+LM</i>	(iii) <i>G2P Lex + CS-KWLM Int</i>
Baseline	0.1895	0.1883	0.3044
sMBR	0.2069	0.2093	0.3287
KW-boosted sMBR	0.2104	0.2220	0.3396

In Table 18, the WER was reduced from the first row of baseline to the second row of sMBR. However, it was increased slightly from the second row to the third row since the KW-boosted sMBR training criterion put much less focus on words not used by the keywords.

Table 18. WER (4,065 keywords) of the Vietnamese LLP systems on the *evalpart1* test set with three test configurations.

WER [<i>evalpart1</i>]	(i) <i>orig Lex + LM</i>	(ii) <i>G2P Lex+LM</i>	(iii) <i>G2P Lex + CS-KWLM Int</i>
Baseline	66.6	66.6	67.0
sMBR	65.0	65.0	66.0
KW-boosted sMBR	66.1	65.9	66.3

5.4.2.2 The influence of KW-boosting weights

In KW-boosted sMBR, a boosting-weight parameter β is used to control the weight ratio between the accuracies of keywords and non-keywords as presented in Eq. (22). In the

following experiments, the influence of the KW-boosting weights on the system performance is investigated.

Six KW-boosted sMBR systems with $\beta= 2, 4, 5, 6, 8,$ and 10 were trained, and the ATWVs and WERs of the systems on *dev2h* with the *orig Lex+LM* test configuration are shown in Figure 19 and Figure 20, respectively. Since the conventional sMBR system can also be considered as a KW-boosted sMBR system with $\beta= 1$, its performance was also included in Figure 19 and Figure 20 for comparison. From the plots, it is clear that β plays an important role to the keyword detection accuracy. ATWV at $\beta = 1$ (sMBR system) is about 0.22, and it is improved as β increased till $\beta = 5$, in which the KW-boosted sMBR system achieved an ATWV of 0.255, which is about a 16% relative improvement over the sMBR system. The ATWV then decreased to 0.235 when $\beta = 8$ and slightly increased to 0.24 when $\beta = 10$. As shown in Figure 19, the relation between the system β and the ATWV performance is not simple, and thus a development set is required for tuning this parameter. On the other hand, from Figure 20 it is clear that the WER of KW-boosted sMBR system increases as β increases, as expected.

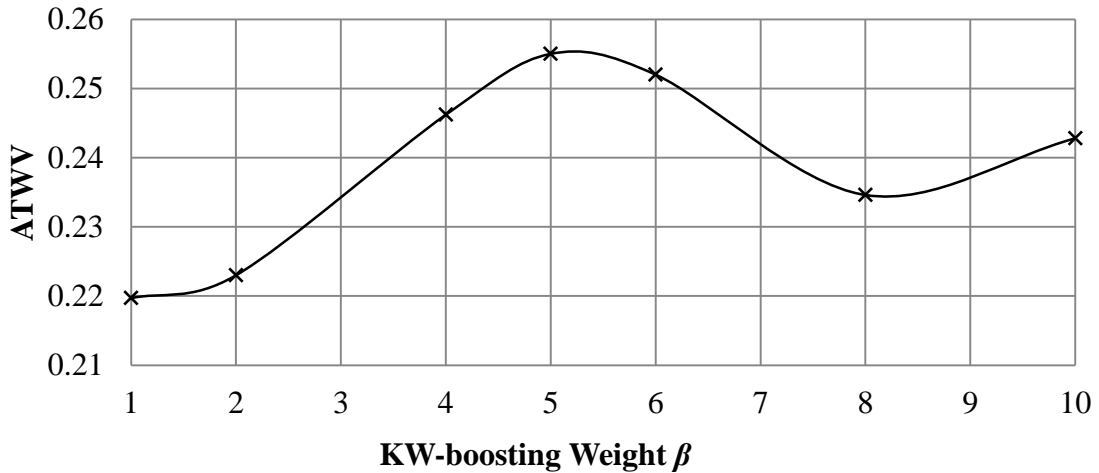


Figure 19. ATWV (4,065 keywords) of Vietnamese KW-boosted sMBR systems on *dev2h* with different keyword-boosting weight values of β .

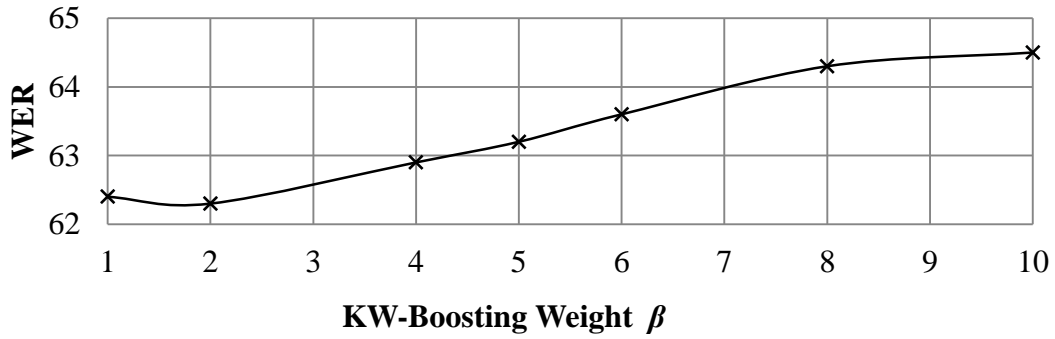


Figure 20. WER (4,065 keywords) of Vietnamese KW-boosted sMBR systems on *dev2h* with different keyword-boosting weight values of β .

5.4.2.3 The influence of keyword set size on KW-boosted sMBR training

In the previous experiments, the KWS systems were built with the 4065-keyword evaluation list. However, since the idea of KW-boosted sMBR is to put more weights on the accuracies of words been considered, it is interesting to know whether the percentage of the interested words in the system vocabulary affects the performance of the KW-boosted sMBR systems. In this following, KW-boosted systems with the 200-keyword development list were built. As in Section 5.4.2.2, the relations of ATWV and WER vs. β are illustrated in Figure 21 and Figure 22, respectively.

The 4065-keyword list used 2,155 words in the system's 3210-word vocabulary (namely about two thirds of words in the vocabulary), while the 200-keyword list covers 408 words (about an eighth) in the system vocabulary. Comparing Figure 19 and Figure 21, the first thing been noticed is that the β -ATWV curves are very different for these two keyword sets. For the 200-keyword set, unlike in the 4065-keyword systems, ATWV of the systems dropped when β increased from 1 to 2. In Figure 21, the ATWV of KW-boosted sMBR systems oscillated between 0.22 and 0.19 and is worse than the sMBR system till $\beta = 6$, where the KW-boosted sMBR system start getting better than the sMBR system. When $\beta = 8$ the proposed system achieved the best ATWV at 0.2124, however the number dropped again when $\beta = 10$. This observation shows that the smaller the keyword set size is, the less stable the system performance is as β changes. It also shows that, the optimal parameter for KW-boosted sMBR is different for different keyword sets,

and the best way of finding the optimal β is tuning it with a small development data. The ATWV of the *dev2h* tuned 200-keyword KW-boosted sMBR system (with $\beta = 8$) on *evalpart1* is 0.1453, which is again better than the 200-keyword sMBR system whose ATWV is at 0.1349.

For the system WER, similar trends were observed for both of the 4065-keyword and 200-keyword sets. When comparing Figure 20 and Figure 22, it is easy to see that WER of the 200-keyword systems increased faster with increasing β than that of the 4065-keyword system because non-keyword words, whose accuracies are less cared in KW-boosted sMBR training, occupy a much higher percentage in the 200-keyword systems.

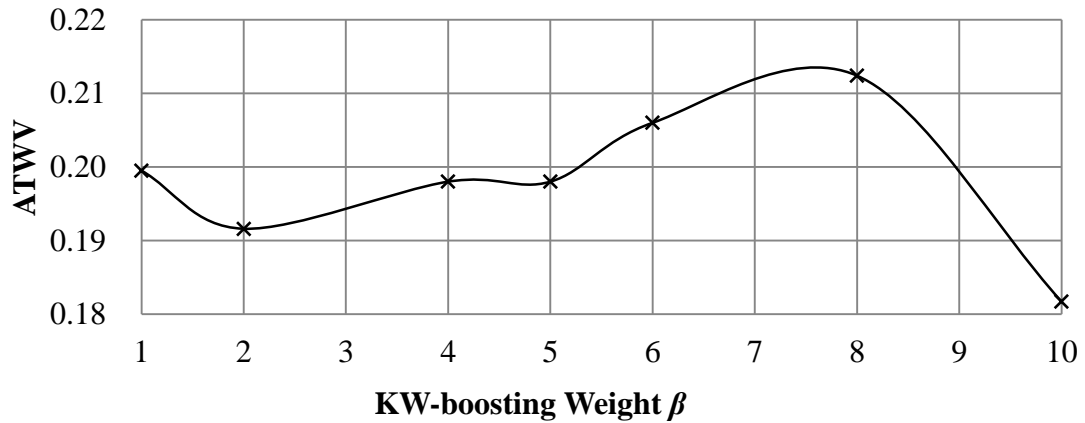


Figure 21. ATWV (200 keywords) of Vietnamese KW-boosted sMBR systems on *dev2h* with different keyword-boosting weight β .

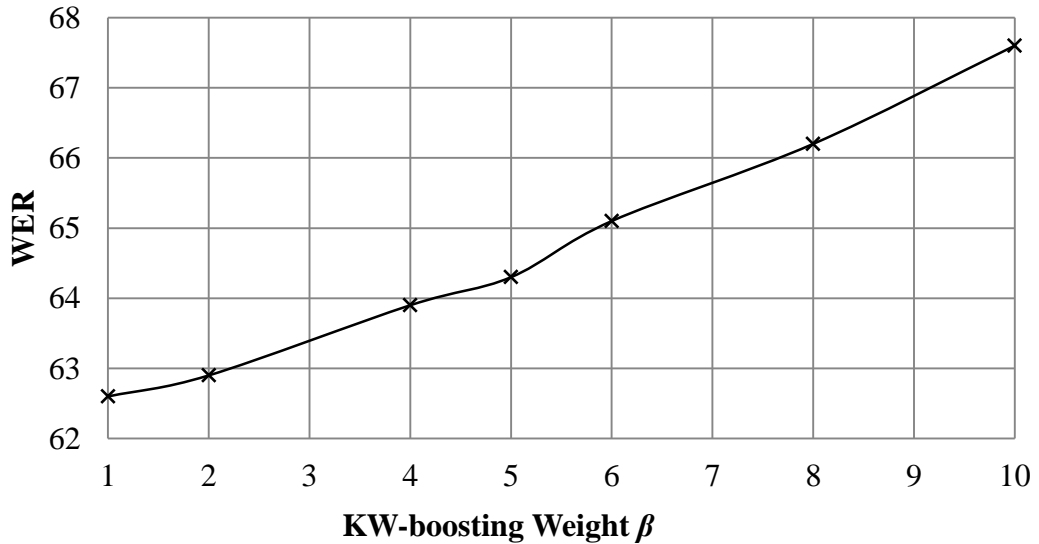


Figure 22. WER (200 keywords) of Vietnamese KW-boosted sMBR systems on *dev2h* with different keyword-boosting weight β

5.4.2.4 System 1-best output analysis

To realize why the KW-boosted sMBR criterion worked for keyword search tasks, the 1-best sentence outputs of both conventional sMBR and KW-boosted sMBR systems were further analyzed. By categorizing the system vocabulary into keyword-word (KWW) and non-keyword word (non-KWW) groups, the correction, substitution, deletion, and insertion rate regarding words in these two groups for the two systems were evaluated. The 4065-keyword evaluation list was used in the experiments.

Table 19 shows the performance of KWW and non-KWW for sMBR and KW-boosted sMBR systems under the *orig Lex+LM* test configuration on the *dev2h* data, where the ATWV of KW-boosted sMBR system (0.2564) outperformed the ATWV of the sMBR system (0.2101) by 20% relative. In Table 19, comparing the KWW performance of both systems, the KW-boosted sMBR system has higher correction and insertion rates, and is with much lower rate for deletion. While for the non-KWWs, the KW-boosted sMBR system has high deletion rate and is with lower rate for correction and insertion rate. In other words, the KW-boosted sMBR system tends to generate keyword words in the system output while reducing the output of non-keyword words. In Eq. (22), the training criterion with more weight on the reference KWW segments implies the criterion penalizes KWW misses more than insertions of KWWs. As a result, the KW-boosted sMBR trained systems are prone to generating keyword words with higher scores in the decoding lattices. Since ATWV is a measure with higher emphasis on misses than on false alarms [123, 124], this tendency provides KWS systems a better chance to get good performance result on ATWV. On the other hand, this also explains the non-trivial relations between the boosting factor β and ATWV for the KW-boosted sMBR system since with inappropriately estimated β , a KW-boosted sMBR system may generate too many keyword words and result in high false alarm rate which greatly degrades the ATWV performance. This suggests that a possible improvement for the current KW-boosted sMBR criterion is to take the keyword words appearing in lattice hypotheses into consideration as well so that the false alarm of KWWs can be also penalized with higher weight.

Similar results were observed in the *G2P Lex + CS-KWLM Int* test configuration (shown in Table 20). The KW-boosted sMBR system has higher correction and insertion rates for KWWs, while for non-KWWs the system has higher deletions.

Table 19. The 1-bset sentence analysis (correction, substitution, deletion, and insertion) for the *orig Lex+LM* test configuration on the Vietnamese *dev2h* data with 4,065 keywords. ATWV for sMBR and KW-boosted sMBR systems are 0.2101 and 0.2564.

	System	Corr(%)	Sub(%)	Del(%)	Ins(%)
KWW	sMBR	41.4	43.6	15.1	10.3
	KW-boosted sMBR	43.1	47.0	10.0	15.8
non-KWW	sMBR	40.4	31.5	28.0	17.0
	KW-boosted sMBR	35.4	25.8	38.8	8.4

Table 20. The 1-best sentence analysis for *G2P Lex+ CS-KWLM Int* test configuration on the Vietnamese *dev2h* data with 4,065 keywords. ATWV for sMBR and KW-boosted sMBR systems are 0.3546 and 0.3649 respectively.

	System	Corr(%)	Sub(%)	Del(%)	Ins(%)
KWW	sMBR	40.0	44.9	15.1	10.3
	KW-boosted sMBR	42.8	47.0	10.2	15.2
non-KWW	sMBR	38.5	30.5	31.0	14.5
	KW-boosted sMBR	34.6	26.5	38.9	7.6

5.4.3 Discriminative keyword-aware language modeling

5.4.3.1 Setup

The baseline system for the discriminative keyword-aware language model experiments is the system with sMBR trained DNN AM and the *G2P Lex + CS-KWLM Int* test

configuration. Training data for the GLM-based DLM were prepared by following the procedure reported in [14]:

The 10-hour Vietnamese LLP training data were first partitioned into 20 sets. For each set, a CS-KWLM system proposed in Chapter 4 was used to generate training lattices of the set. Since language models are prone to over-fitting more easily than acoustic models, the LM used in each CS-KWLM system was built by using only transcriptions from the other 19 sets. In other words, 20 different CS-KWLM interpolated LMs were used in the 20 CS-KWLM systems for training lattice generation. The generated training lattices of the 20 sets then merged to form the complete training data for DLM training. For acoustic models, all the 20 CS-KWLM systems shared the same DNN AM which was trained with the entire 10-hour training data with sMBR training criterion.

Note that the 20 CS-KWLM interpolated LMs mentioned above were only used in training lattice generation. In testing phase, the LM used in the CS-KWLM system was trained with the entire LLP transcriptions as usual.

5.4.3.2 ASR performance without MBR decoding

Since the DLM approach was originally designed for ASR tasks, in the first experiment the word error rate (WER) and sentence error rate (SER) of the baseline system and the DLM systems were compared. Table 21 and Table 22 show the experimental results of the systems using conventional MAP decoding, i.e., Eq. (4) and (24), on *dev2h* and *evalpart1* data. The WERs for the baseline CS-KWLM Interpolation system on the two data sets are 68.44% and 70.96% respectively. Note that the WERs here are much higher than the numbers reported for CS-KWLM Interpolation in section 4.7.3 (last rows in Table 7 and Table 8, where the WERs for the CS-KWLM Interpolation system are 63.5% for *dev2h* and 66.0% for *evalpart1*) because the decoding method in section 4.7.3 is discriminative minimum Bayes Risk (MBR) decoding. In Table 21 and Table 22, it is clear that the DLM approach is effective when conventional MAP decoding is applied. When using only 1-gram as the features in the global linear model, the system WERs reduced to 68.07% and 70.49% on *dev2h* and *evalpart1* data respectively. The system accuracies were further improved by including 2-grams as features in the global linear

model. However, including 3-gram features did not help the DLM system much due to the limited amount of training data. Overall, the DLM system achieve the best WER performance on *dev2h* and *evalpart1* data at 67.59% and 70.05% when using n -gram features with n being up to 3 and 2 respectively. The SER improvements for the DLM systems are much smaller comparing with the improvements on WER. However, the trend is still perceptible.

Table 21. Word Error Rate (WER) and Sentence Error Rate (SER) of DLM systems on the Vietnamese *dev2h* data set using conventional MAP decoding in the Vietnamese LLP task.

Vietnamese LLP [<i>dev2h</i>]	WER (%)	SER (%)
CS-KWLM Interpolation $\alpha=0.6$ (baseline)	68.44	96.74
1-gram DLM rescore	68.07	96.86
2-gram DLM rescore	67.61	96.56
3-gram DLM rescore	67.59	96.50

Table 22. Word Error Rate (WER) and Sentence Error Rate (SER) of DLM systems on the Vietnamese *evalpart1* data set using conventional MAP decoding in the Vietnamese LLP task.

Vietnamese LLP [<i>evalpart1</i>]	WER	SER
CS-KWLM Interpolation $\alpha = 0.6$ (baseline)	70.96	94.36
1-gram DLM rescore	70.49	94.26
2-gram DLM rescore	70.05	94.11
3-gram DLM rescore	70.07	94.14

5.4.3.3 Real system performance

The default ASR decoding method in this dissertation is minimum Bayes Risk (MBR) decoding instead of the conventional MAP approach. This section shows the real system performance for ASR and KWS. The MBR decoding is a discriminative decoding approach that exploits the posterior probabilities of word arcs in lattices to evaluate the expected WER for each path. The path with minimum expected WER is then selected as the decoding result for ASR output. It has been shown that with the discriminative decoding criterion, the MBR decoding usually achieves much lower WER than the conventional MAP based decoding [125]. Note that both the MBR decoding and KWS processes require estimation of the word-arc posterior probabilities in the lattices of test utterances.

Table 23 and Table 24 show the WER and ATWV of the baseline CS-KWLM Interpolation system and the DLM systems on *dev2h* and *evalpart1* data in the Vietnamese LLP task. It is clear that the WER performance of the CS-KWLM Interpolation system was significantly improved by using MBR decoding when comparing the results with the numbers shown in Table 21 and Table 22. In general, the MBR decoding provided about 5% absolute WER reduction over the conventional MAP decoding. However, the WER improvements of the DLM systems observed in conventional MAP decoding diminished after applying MBR decoding. On the *dev2h* data, though the 2-gram DLM system achieved 66.3% of WER, the improvement is very small when comparing it with the improvement observed in section 5.4.3.2. On the *evalpart1* data, all the DLM systems received worse WER compared with the baseline CS-KWLM interpolation system. The WERs of the 2-gram and 3-gram DLM systems even rose to about 81% because of a very high word deletion rate caused by incorrect word arc posterior estimation in the test lattice. The same problem was observed in ATWV as well. The DLM systems attained worse ATWVs because of the incorrect posterior estimation.

In conclusion, for DLMs with un-normalized n -gram probabilities, the effect of DLM could be diminished when discriminative decoding is applied. The DLM could also degrade the KWS performance dramatically when un-normalized probabilities are used.

The problem is especially critical for GLM-based DLM due to the feature setting of the framework.

Table 23. WER and ATWV of the baseline CS-KWLM interpolation and DLM systems on the Vietnamese *dev2h* data in the Vietnamese LLP task.

Vietnamese LLP [<i>dev2h</i>]	WER	ATWV
CS-KWLM Interpolation $\alpha = 0.6$ (baseline)	63.5	0.3546
1-gram DLM rescore	63.4	0.3582
2-gram DLM rescore	66.3	0.3503
3-gram DLM rescore	66.4	0.3408

Table 24. WER and ATWV of the baseline CS-KWLM interpolation and DLM systems on the Vietnamese *evalpart1* data in the Vietnamese LLP task.

Vietnamese LLP [<i>evalpart1</i>]	WER	ATWV
CS-KWLM Interpolation $\alpha = 0.6$ (baseline)	66.0	0.3287
1-gram DLM rescore	66.3	0.3241
2-gram DLM rescore	81.7	0.2568
3-gram DLM rescore	81.6	0.2859

5.5 Conclusion

In this chapter, the effects of using discriminative training objectives for both acoustic and language modeling in KWS systems is investigated. For acoustic modeling, a

modified objective function of sMBR training for DNN-HMM hybrid systems is proposed to enhance the performance of keyword search tasks. By putting more weight on acoustic modeling of keyword states, the KW-boosted sMBR system is capable of detecting more keywords while reducing false alarms. Experimental results show that the proposed KW-boosted sMBR trained systems outperform the baseline sMBR systems by a relative increase of 1.7 ~ 6.1% in ATWV in all test configurations. An analysis of the experimental results also shows that tuning the keyword-boosting weight β with different keyword sets is a key to KW-boosted sMBR training. For language modeling, the GLM-based DLM, which is one of the most popular DLM methods used in ASR tasks, was adopted to enhance the KW-aware LMs proposed in Chapter 4. Experimental results, however, show that the convention DLM approach does not work for KWS tasks due to the difficulty of correct posterior probability estimation in lattices.

The current KW-boosted objective function only focuses on keyword segments in the reference sentences. The KWS systems therefore tend to generate more keyword words. Though the setting significantly reduced keyword miss rates for system, it may also create unwanted false alarms. A possible extension of the proposed approach is to introduce the boosting weight to words used by keywords in hypothesized sentences as well so that both miss and false alarm rates of keywords can be minimized at the same time.

CHAPTER 6 DATA AUGMENTATION

6.1 Introduction

When the amount of in-domain training data is too small to support the estimation of distributions $p(O|W)$ and $P(W)$ in the test condition, the effect of utilizing *data-efficient training processes* and *modifying system optimization objectives* presented in the previous chapters on the system performance could become considerably smaller. The mismatches between training data estimated distributions $p_{\Lambda^*}(O|W)$ and $P_{\Gamma^*}(W)$ and the test distributions would still be large. Data augmentation (DA) [74-79], which utilizes out-of-domain data to provide prior knowledge about the potential test distributions in the modeling process, is an efficacious technique to enhance the system performance in this situation. Common data augmentation approaches for acoustic modeling include semi-supervised training (SST) [75, 77], acoustic data perturbation [74-76], and multilingual acoustic modeling [52, 53]. For language modeling, it is common to use web text data to assist model training.

In this chapter, two important and effective data augmentation techniques – *multilingual acoustic modeling* and *web text augmented language modeling* – used in ASR and KWS community were explored. Issues regarding the optimization of the DA approaches, including the selection of out-of-domain data sources and the design of training procedure, are studied. Integration of the DA approaches and the KW-aware language modeling framework proposed in Chapter 4 is also inspected. Section 6.2 proposes a multilingual acoustic modeling recipe for DNN AMs. In Section 6.3, the web text augmented language modeling procedure used in this dissertation is described. Experimental results and analysis are presented in Section 6.4.

6.2 Multilingual Acoustic Modeling for Deep Neural Networks

The multilingual acoustic modeling recipe proposed in this section is a modified version of the shared-hidden-layer multilingual DNN (SHL-MDNN, depicted in Figure 23) framework outlined in [19]. The recipe consists of two training stages:

1. **Multilingual training** stage: where an SHL-MDNN acoustic model is trained with target and foreign languages simultaneously, and
2. **Target language fine-tuning** stage: in which a DNN AM of the target language is initialized with the SHL-MDNN and trained with target language data to further enhance the system performance.

The major difference between the SHL-MDNN framework proposed in [19] and the training recipe proposed here is the *target language fine-tuning* stage, which does not exist in [19]. The two training stages are explained in the following sections. Experimental results for multilingual acoustic modeling are presented in section 6.4.3.

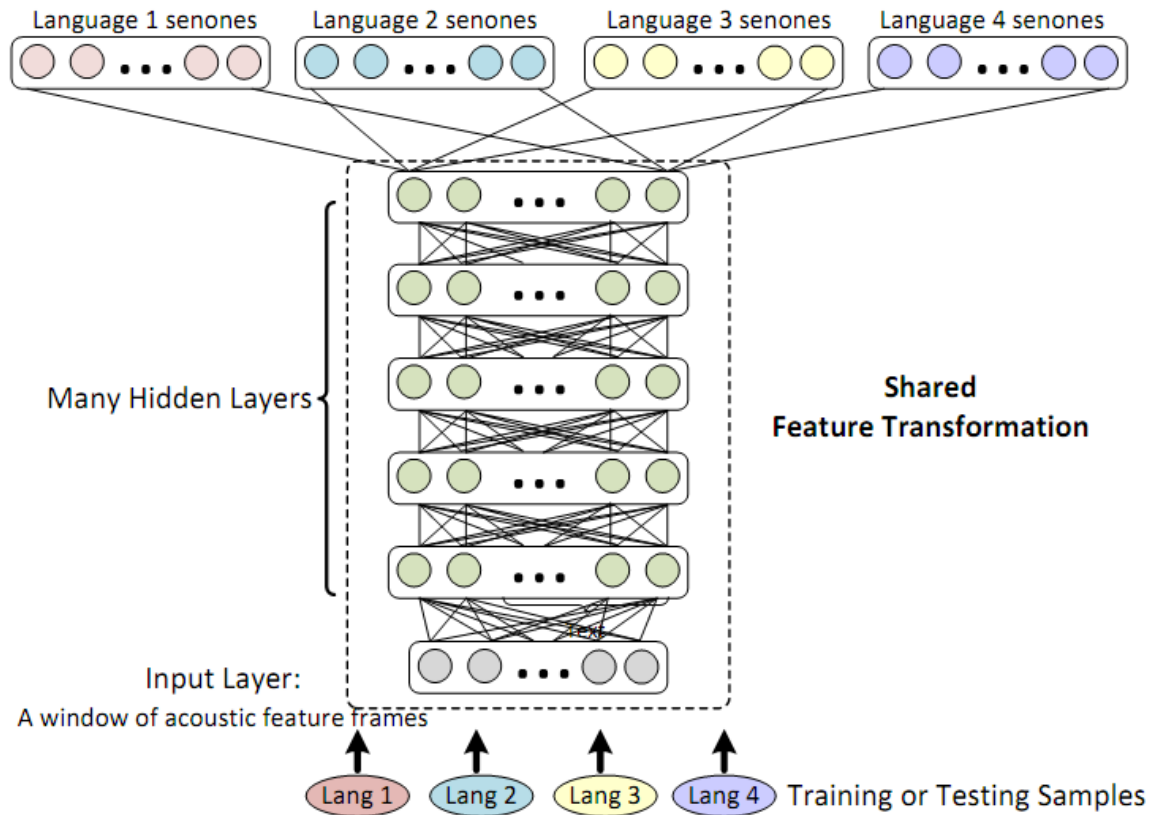


Figure 23. An illustration of the shared-hidden-layer multilingual DNN (SHL-MDNN) proposed in the reference [19] (the figure is adopted from the paper). In this dissertation, the SHL-MDNN framework is used in the multilingual acoustic modeling recipe.

6.2.1 Multilingual training for DNN acoustic models

In conventional GMM-HMM based LVCSR systems, the likelihoods of an HMM state (senone) are estimated by Gaussian mixture models (GMM)[2]. In hybrid DNN-HMM based LVCSR systems, instead of using GMMs, DNN models are used for the likelihood¹⁶ estimation [4]. The input of the DNN models are splices of acoustic features such as MFCC and PLP used in the conventional GMM-HMM based ASR systems. The output layers of the DNN models are softmax layers, where each output node in the layers generates the posterior probability of its corresponding senone given an input acoustic feature frame. The likelihood of a senone can then be evaluated by dividing the posterior probability with the senone's prior probability, which can be estimated from the training data, and used in HMM for decoding. Usually, for each LVCSR task, an individual DNN AM is used. However, since in DNN AMs the output softmax layer is the only layer closely related to the task target outputs, the function of the rest of the DNN AMs can be seen as for feature transformation (see Figure 23). If we can share the transformation across languages and tasks, we may use all the training data of the languages and tasks to estimate a more robust feature transformation and enhance system performance for all the languages and tasks.

The design of the shared-hidden-layer multilingual DNN (SHL-MDNN) is based on this idea. In the SHL-MDNN, the input and hidden layers are shared across languages while only the softmax output layers are language dependent. The training process is very similar to conventional DNN training. Given an acoustic feature input, the neural network first does forward propagation through all the hidden layers to generate softmax output at output layer. For each input, only the softmax output layer corresponding to the language/task of the input is activated. Errors of the softmax output are then evaluated using the training objective function and back-propagated through the whole DNN for parameter updates. Therefore, the hidden layers in a SHL-MDNN can be updated with all

¹⁶ Though outputs of DNNs are posterior probabilities of HMM-states, to adapt DNN AMs into conventional HMM-based decoding framework, people still generate "pseudo" likelihoods for the purpose by dividing the state posterior probabilities with their priors as shown in Section 5.2.2.

the training data; while the softmax layer would only be updated with the data of the corresponding language/task.

A key to the success of SHL-MDNN training is that training data of all the languages should be fully shuffled and interleaved when doing the training. This is especially important when mini-batch is used to avoid the problem that the DNNs over-fit to a specific language and fall into an unwanted local maximal region for parameters in the early stage of training. The implementation of the SHLD-MDNN in this dissertation consists of a pre-training process for all the hidden layers using the target language data. The softmax output layers for each language are then added after the pre-trained hidden layers as shown in Figure 23. The multilingual training process described above then is used for the SHL-MDNN training.

The Kaldi default DNN training configuration, which is empirically tuned on the English wall street journal (WSJ) corpus, is adopted in the SHL-MDNN training process and is described as follows: In the training process, the mini-batch size and the initial learning rate are set to 256 inputs and 0.008, respectively. To determine when to halve the learning rate or terminate the training process, a held-out validation set of the training data is used. When the relative cross-entropy improvement on the validation data is smaller than 0.01, the training process starts to halve the learning rate every training iteration. Finally the process would be terminated if the relative cross-entropy improvement on the validation data is smaller than 0.001.

6.2.2 Target language fine-tuning for DNN acoustic models

After the multilingual training stage, the output layers of the non-target languages in the SHL-MDNN are removed as shown in Figure 24. The resulting DNN is a DNN with only the output layer of the target language. In other words, it is a DNN AM for the target language and can be used for KWS immediately. However, since, in multilingual training, usually the amount of target language data is considerably smaller when compared with the amount of the foreign language data, the DNN is not optimized toward the target language. A fine-tuning stage, using only the target language data to optimize the DNN, is therefore necessary to enhance the DNN performance. Discriminative training can also be used in this stage to further improve the system performance.

The fine-tuning process consists of a cross-entropy training followed by a sMBR training using the target language training data. Since training data of the target language is very limited, to avoid over-fitting, only parameters at the last n layers in the DNN are updated, where n can be determined using development data. Empirically, $n = 3$ or 4 is a reasonable choice for a target language with 3 hour training data. In addition to restricting the number of layers being updated in the DNN model, since the DNN has already been well pre-trained, it is also observed that a smaller initial learning rate (comparing with the one used in the first training stage) for cross-entropy training in the fine-tuning stage is needed to guarantee the success of the model fine-tuning process.

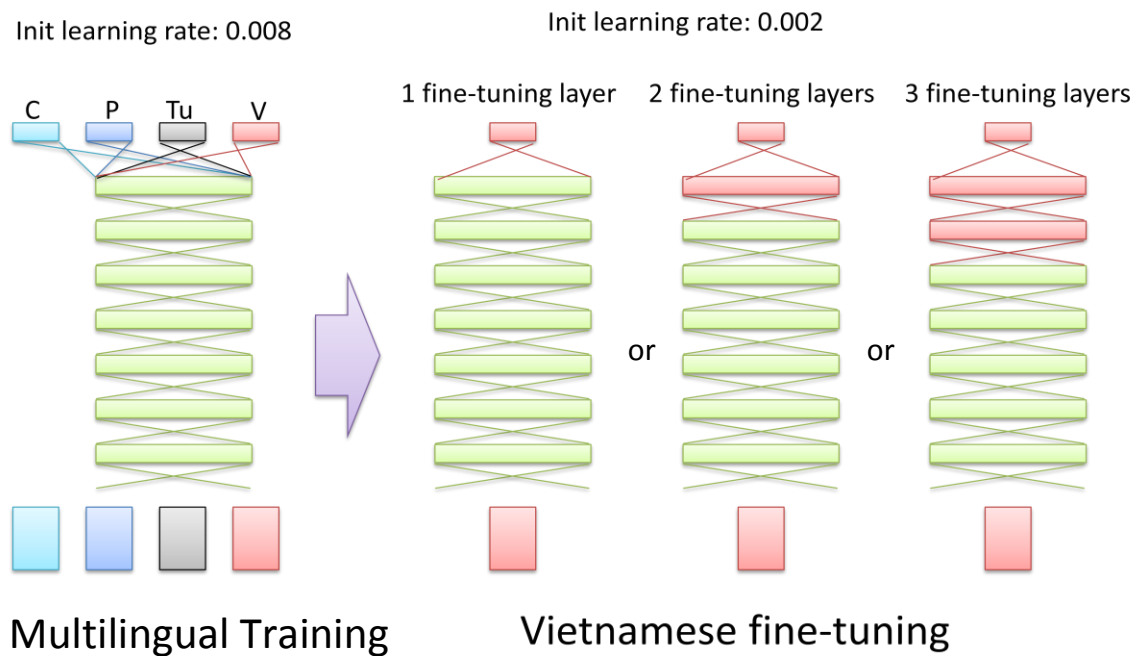


Figure 24. An illustration of the multilingual acoustic modeling recipe for Vietnamese DNN AMs. The target language (Vietnamese) fine-tuning is followed after the multilingual training stage. The initial learning rates for multilingual training and target language fine-tuning are empirically set to 0.008 and 0.002, respectively.

6.3 Web Text Augmented Language Modeling

Web text augmented language modeling¹⁷ is another common technique utilized by ASR and KWS systems when the amount of in-domain training text is limited [20, 21, 126, 127]. The additional information provided by the web text content usually benefits language modeling for covering potential topics in test conditions. To integrate the web data information into the language model, the most common approach is to build a standalone language model using the web text content and then combine the web LM with the original in-domain LM using Eq. (25) for linear interpolation [21, 126, 127]:

$$P_{INT_LM}(w|h) = \alpha_{Web} \cdot P_{Web}(w|h) + (1 - \alpha_{Web})P_{LM}(w|h), \quad (25)$$

where the α_{Web} is a tunable parameter controlling the weight of the web LM in the interpolation. Since the Internet is filled with a variety of content, selecting the right web data sources is the key to the maximization of the LM enhancement. A set of data selection approaches has been proposed in the past decades. In [127], the coverage of unseen n -grams in the test data is used to measure the web data quality. The reference [20] also proposed the use of perplexity to measure the similarity between the web text and the in-domain training data for data selection.

One of the goals of this chapter is to investigate web text augmented language modeling in KWS tasks. The IARPA OpenKWS13 Vietnamese limited language pack (LLP) and very limited language pack (VLLP) tasks are selected for the purpose. In the OpenKWS13 Vietnamese VLLP task, there are six web data sources collected with different approaches provided by NIST:

- **Wiktionary** – text contents dumped from the wiktionary website of the target language. The text content is a dictionary of the target language. The data provides high vocabulary coverage. However, most of the contents are

¹⁷ Ideally, web text contents should be used to enrich both system vocabularies (i.e., lexicons) and language models simultaneously for the best performance enhancement. However, since vocabulary enhancement is beyond the research scope of this dissertation, only language modeling would be focused on here.

incomplete sentences. The style of writing is also very different from telephone conversation, which is the task domain of the OpenKWS tasks.

- **Wikipedia** – text contents dumped from the Wikipedia website of the target language. The text content is an encyclopedia of the target language. The data provides great variety of topics and good coverage of vocabularies. However, the writing style is usually too formal and technical for the conversation training data.
- **Open Subtitle** – movie subtitles for the target language downloaded from the Open Subtitle website (<http://www.opensubtitles.org/>). The text contents are movie scripts of the target language. The data provides sentences of designed conversations, which are similar to the style of telephone speech to some extent.
- **TED talk transcriptions** – TED talk transcriptions in the target language crawled from the TED talk site (<https://www.ted.com>). The contents provide broad topic coverage. However, the style of the sentences is closer to monologue than conversation.
- **Blogspot** – text contents of Blogspot feeds containing target language data crawled from Blogspot (<http://xxxx.blogspot.com/xxx>). The data provides wide domain coverage. However, the writing style could be very different from telephone conversation.
- **Query results** – website contents retrieved by using the top 220 n -grams in the Vietnamese VLLP training transcriptions as queries for the Google search engine. The advantage of data collected in this way is that the collected text would contain popular n -grams in the in-domain training data. It is also one of the most popular data collecting approaches employed by many research groups because of the simplicity [20, 21]. The data is provided by BBN (Raytheon Company).

This chapter examines the performance enhancements of these six web data sources on the OpenKWS Vietnamese KWS tasks. The influences of the original in-domain training text amount and the combination of keyword-aware language modeling on the

KWS performance are also investigated. Experimental results are presented in Section 6.4.4.

6.4 Experiments

6.4.1 Experimental setup

Experiments were conducted on the IARPA Babel OpenKWS13 (Vietnamese) [92] and OpenKWS14 (Tamil) [93] very limited language pack (VLLP) tasks. Note that, unlike the LLP tasks used in the previous chapters, the VLLP tasks only has 3 hours of transcribed audio from the target language for system training. The OpenKWS13 Vietnamese LLP task, which has 10-hour training data, was also used for investigating the performance of data augmentation approaches when more in-domain training data are available. For parameter tuning, a 2-hour subset of the IARPA development set (denoted as *dev2h*) is used for each language.

Note that, in addition to the Vietnamese and Tamil VLLP data, the training data of six full language pack (FLP) tasks, which include Vietnamese (V), Tamil (Tm), and another four languages, were also provided by NIST in the OpenKWS competition. These FLP data can be used as augmenting data in the construction of the Vietnamese and Tamil VLLP KWS systems. Table 25 shows the details of the transcribed audio data used in the experiments. The rows of the three tasks focused in this experiment – Vietnamese LLP and VLLP, and Tamil VLLP – are highlighted in the table. The four additional languages are Cantonese (C), Pashto (P), Turkish (Tu), and Tagalog (Tg), and the amount of their audio data ranged from 70 to 129 hours. The Vietnamese FLP training set, which has 78-hour transcribe audio data, can be used for Tamil KWS system construction, but are not allowed to be used in Vietnamese LLP and VLLP system building. The same rule applies to the Tamil FLP data.

For all the Vietnamese and Tamil systems, the 15-hour evaluation part 1 data (released as *evalpart1* by NIST) were used for testing. The evaluation keyword lists contain 4,065 and 5,576 phrases with out-of-vocabulary words not appearing in the training set for the two languages respectively. Both Vietnamese LLP and VLLP systems

use the Vietnamese LLP lexicon provided by NIST. The vocabulary size of the Vietnamese LLP lexicon is 3,208 words. While for the Tamil VLLP system, the official Tamil FLP lexicon with 58,474 words is used.

Instead of using advanced BNF and fMLLR features as in Chapter 4 and 5, acoustic features used in the VLLP systems are 40 dimensional log filter bank (Fbank) features concatenated with 3 dimensional F0 features extracted using Kaldi tools. The reason for selecting these "raw" features is that the advanced features usually involve language dependent transformations, which lead to different feature spaces for different languages. This feature space mismatch among languages would degrade the performance of multilingual acoustic modeling. Further, experimental results showed that the amount of 3-hour training data in the VLLP tasks is not enough to train a good bottleneck DNN for feature extraction. The performance of VLLP systems with BNF+fMLLR feature is much worse than the performance of systems using the "raw" FBank+F0 feature.

Table 25. Training data sets used in the experiments. The FLP training data can be used for the multilingual acoustic modeling. The Vietnamese LLP/VLLP and Tamil VLLP data were used for pure Vietnamese/Tamil system building and the *target language fine-tuning* stage in the multilingual acoustic modeling recipe. Note that we reserved a tenth of the task data as validation set when doing DNN training.

Language	Babel LID	Data Set	Training (hr)	Validation (hr)	Total (hr)
Cantonese	101	FLP	126.62	2.13	128.75
Pashto	104	FLP	70.12		70.12
Turkish	105	FLP	69.55		69.55
Tagalog	106	FLP	75.54		75.54
Vietnamese	107	FLP	78.28		78.28
		LLP	9.75	1.17	10.92
		VLLP	2.70	0.33	3.03
Tamil	204	FLP	56.69		56.69
		VLLP	2.70	0.28	2.98

6.4.2 Analysis of the importance of AM and LM in VLLP tasks

The first experiment investigated the importance of acoustic and language models toward the KWS performance for systems with very-limited training resources. The results could help us determine the priority of acoustic and language model enhancement when building a KWS system. The ATWV and WER of Vietnamese systems with AMs and LMs trained with different amount of training data were compared. Two AMs, trained with LLP and VLLP data, and two LMs, trained with LLP and VLLP transcriptions, were considered in the experiment. Experimental results are shown in Table 26 and Table 27.

In Table 26, it is obvious that the LLP system with both LLP-data trained AM and LM retained the best ATWV performance at 0.2145. For the VLLP system, in which both AM and LM were trained with 3-hour VLLP data, the ATWV performance significantly dropped to 0.0572. The results show that the impact of training data amount is indeed very high to system performance. However, if the AM in the VLLP system being replaced with the LLP-data trained AM, the system ATWV increased remarkably from 0.0572 to 0.1883, which is very close to the LLP system's performance. On the other hand, switching the LM in the VLLP system with the LLP-trained LM did not provide much performance improvement.

Table 26. ATWVs of KWS systems with different combinations of acoustic and language models on Vietnamese *dev2h* set.

ATWV		Acoustic Model	
		LLP	VLLP
Language Model	LLP	0.2145	0.0583
	VLLP	0.1883	0.0572

Similar results were observed in WER as shown in Table 27. The LLP system has the lowest WER at 64.0%, and the VLLP system has the highest WER at 75.5%. By changing the AM in the VLLP system with the LLP-data trained AM, the system WER remarkably dropped to 65.5%, which is a 10% absolute WER reduction. The improvement is much less significant when the VLLP-data trained LM was replaced with the LLP-data trained LM in the VLLP system.

Table 27. WER of KWS systems with different combinations of acoustic and language models on Vietnamese *dev2h* set.

WER (%)		Acoustic Model	
		LLP	VLLP
Language Model	LLP	64.0	74.1
	VLLP	65.5	75.5

From Table 26 and Table 27, it is clear that AM is more important to system performance when only limited in-domain training data is available. Acoustic model enhancement thus has higher priority than LM in such condition.

6.4.3 Multilingual acoustic modeling

6.4.3.1 Setup

In the multilingual acoustic modeling experiments, the baseline monolingual DNN AMs is a DNN model trained purely with the target language data, namely the 10-hour and 3-hour data in the LLP and VLLP tasks respectively. The VLLP baseline DNN structure consists of 5 hidden layers, which is the best setting found empirically for the monolingual DNN in the VLLP tasks. The multilingual DNN structure, on the contrary,

consists of 7 hidden layers. The hidden layer dimension is 1,200 for both monolingual and multilingual DNN AMs. Both DNNs were pre-trained using target language data only.

As mentioned in the section 6.2, the multilingual training recipe contains two stages: (i) the *multilingual training* stage, and (ii) the *target language fine-tuning* stage. In the multilingual training stage, data of the 6 languages (including the target language) were used for multilingual training, and the cross-entropy training criterion is used at this stage. In the fine-tuning stage, only target language data is used for updating the parameters in the last n fine-tuning layers. The fine-tuning process is the standard sMBR training process for DNN AMs in this dissertation: The process started with cross-entropy training followed by a iteration of sMBR training to form a first sMBR trained DNN AM. The DNN AM was then used to align the training data for second sMBR training with 4 iterations.

Note that the training process of the baseline monolingual DNN AMs is almost the same as the fine-tuning process in the multilingual training recipe. However, there are three major differences between the two processes. First, in the baseline monolingual training, the initial DNN AMs were target language data pre-trained deep belief networks (DBN); whereas the initial models for the fine-tuning process were the DNNs trained with multilingual data. Second, the initial learning rate of cross-entropy training for the baseline DNNs was set to 0.008. However, the rate was set to a much smaller value at 0.002 for the multilingual DNN in the fine-tuning stage since the models were already well-trained with multilingual data. Third, in the baseline monolingual DNN training, the whole DNNs were updated; whereas the multilingual DNNs only updated their last n layers in the fine-tuning stage. Table 28 shows the details of the training data utilized in each stage of the multilingual training recipe for the three tasks.

Table 28. Details of the training data utilized in each stage of the multilingual acoustic modeling recipe for the three tasks. The validation data were used by the iterative cross-entropy training process to determine whether to halve the learning rate in the next iteration or terminate the training process. Language name abbreviations used here are: C – Cantonese, P – Pashto, Tu – Turkish, Tg – Tagalog, V – Vietnamese, Tm – Tamil

Task	Multilingual Training		Target Language Fine-Tuning	
	Training data (hours)	Validation (hours)	Training data (hours)	Validation (hours)
Vietnamese LLP	C+P+Tu+Tg+Tm (398.52) + V(9.75) = 408.27 hours	C (2.13)	V (9.75)	V (1.17)
Vietnamese VLLP	C+P+Tu+Tg+Tm (398.52) + V(2.70) = 401.22 hours	C (2.13)	V (2.70)	V (0.33)
Tamil VLLP	C+P+Tu+Tg+V (420.11) + Tm(2.70) = 422.81 hours	C (2.13)	Tm (2.70)	Tm (0.28)

6.4.3.2 Performance of the multilingual recipe

Figure 25 to Figure 30 show ATWV and WER of the baseline and multilingual systems at each training stage in the three tasks. The first two training steps (multi-train and fine-tuning) are conventional DNN cross-entropy training, and the rest four steps are discriminative sMBR training with target language data.

ATWVs and WERs of the Vietnamese LLP task, which has 10-hour Vietnamese transcribed audio data for system training, are shown in Figure 25 and Figure 26. In Figure 25, it is obvious that the ATWV of the multilingual-data trained DNN AM, though without target language fine-tuning, was already significantly better than the baseline monolingual DNN AM with cross-entropy training. After the multilingual AM being fine-tuned with the 10-hour Vietnamese data, the performance gap was further increased

from about 0.025 to more than 0.04. Both multilingual and monolingual systems benefit from discriminative sMBR training. However, with different number of fine-tuning layers in the multilingual DNN, the multilingual systems achieved different ATWV performance after the sMBR training. The best system is the multilingual system with 3 fine-tuning layers on the 4th iteration of 2nd sMBR training in the fine-tuning stage, which achieved 0.2285 of ATWV on the Vietnamese *dev2h* data. Note that it is important to set a reasonable number n for the n fine-tuning layers used in the multilingual AM. If the number is too small, e.g., $n = 1$ as proposed in [19], to allow the DNNs to absorb the information provided by the target language data, the performance of multilingual AMs could easily saturate during the sMBR training process as shown in Figure 25 for the 1 fine-tuning layer system. For the baseline monolingual system with the 10-hour Vietnamese training data, sMBR training significantly improved the system ATWV from 0.1483 to 0.2121. However, all the multilingual systems (except the one with 1 fine-tuning layer) still outperformed the baseline system in all the training steps.

For WER performance (shown in Figure 26), the advantage of using multilingual AMs is even more obvious. All the multilingual systems significantly outperformed the baseline monolingual system in all training steps. A consistent absolute WER reduction at about 2.5% ~ 3.5% is observed. However, the performance of the multilingual system with 1 fine-tuning layer saturated again during the sMBR training process.

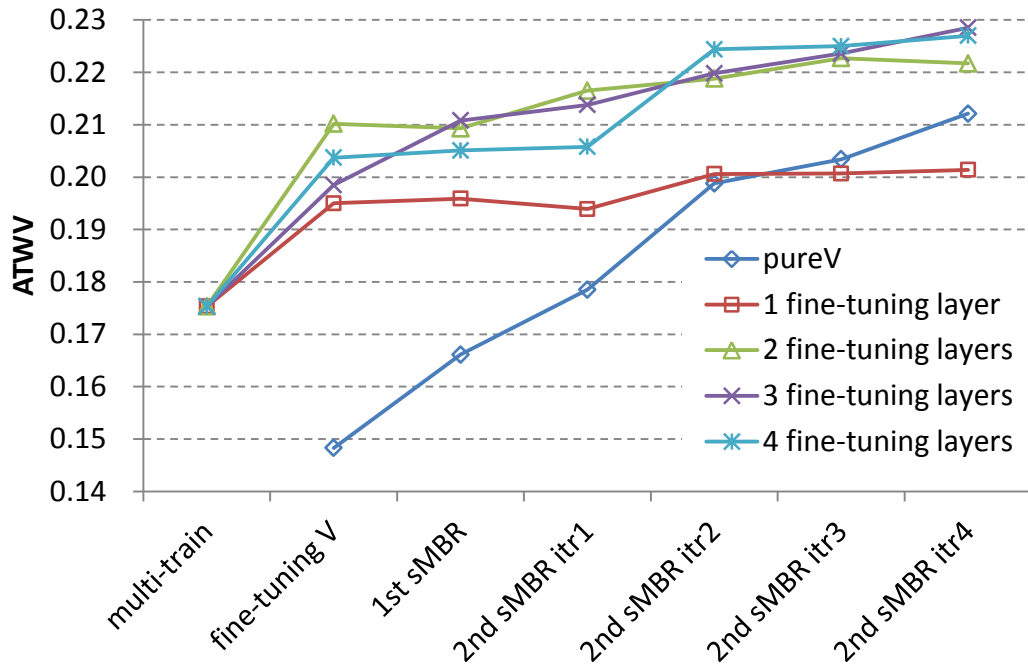


Figure 25. ATWV of multilingual acoustic modeling recipes with different number of fine-tuning layers on *dev2h* data for the Vietnamese LLP task.

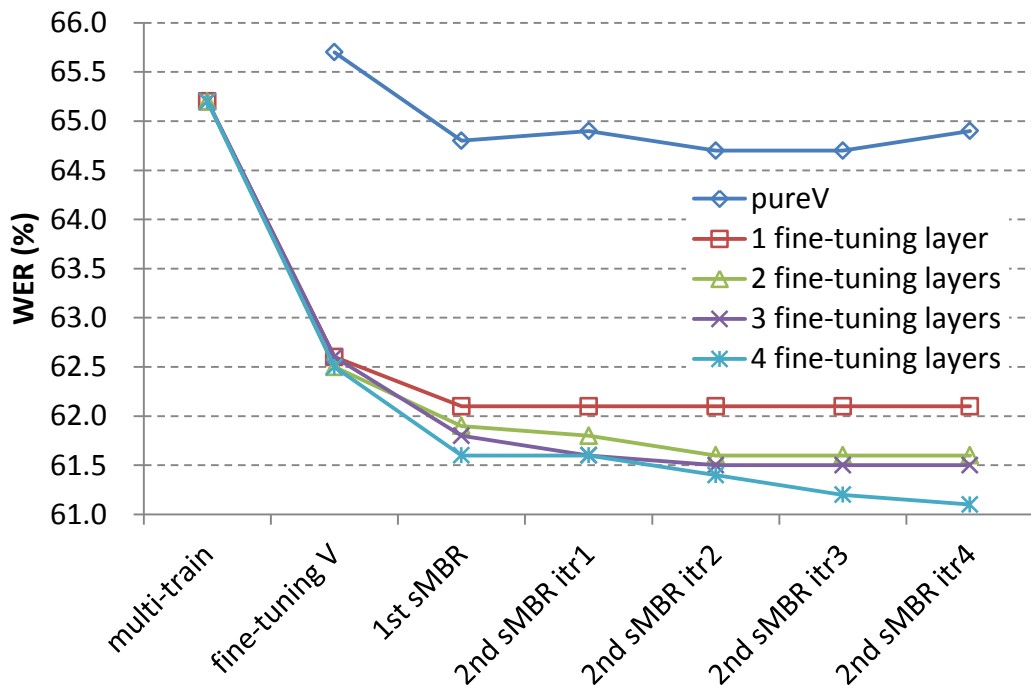


Figure 26. WER of multilingual acoustic modeling recipes with different number of fine-tuning layers on *dev2h* data for the Vietnamese LLP task.

When the amount of the target language training data was reduced to 3 hours in the VLLP tasks, the performance difference between the multilingual and monolingual systems became more prominent. Figure 27 shows the ATWV performance of systems in the Vietnamese VLLP task. With 3-hour Vietnamese training data, the best ATWV achieved by the baseline system is 0.0572 after sMBR training. This performance, however, is much worse than the ATWVs of multilingual systems in all training steps. Even for the multilingual AM without fine-tuning, the system attained an ATWV at 0.0879, which is about 53% relative improvement over the baseline system. The performance of multilingual system can be further improved with the fine-tuning stage. The best multilingual system is the system with 4 fine-tuning layers on the 3rd iteration of 2nd sMBR training, which achieved 0.1344 of ATWV. The relative ATWV improvement of switching from the best monolingual system to the best multilingual system is therefore 135% (from 0.0572 to 0.1344). The multilingual systems also provided significant WER performance improvement over the baseline system as shown in Figure 28. An about 5% absolute WER reduction is observed when switching from the best monolingual system to the best multilingual system.

Similar results can be observed in the Tamil VLLP task as shown in Figure 29 and Figure 30. In Figure 29, the ATWV of the multilingual systems were consistently better than the performance of the baseline Tamil system in all training steps. The best multilingual system is the system with 3 fine-tuning layers on the 4th iteration of 2nd sMBR training. The system achieved 0.1368 of ATWV and provided a 52% relative ATWV improvement over the best baseline system ATWV at 0.0899. For WER performance, the multilingual systems provided more than 3% absolute WER reductions over the baseline system in all test steps.

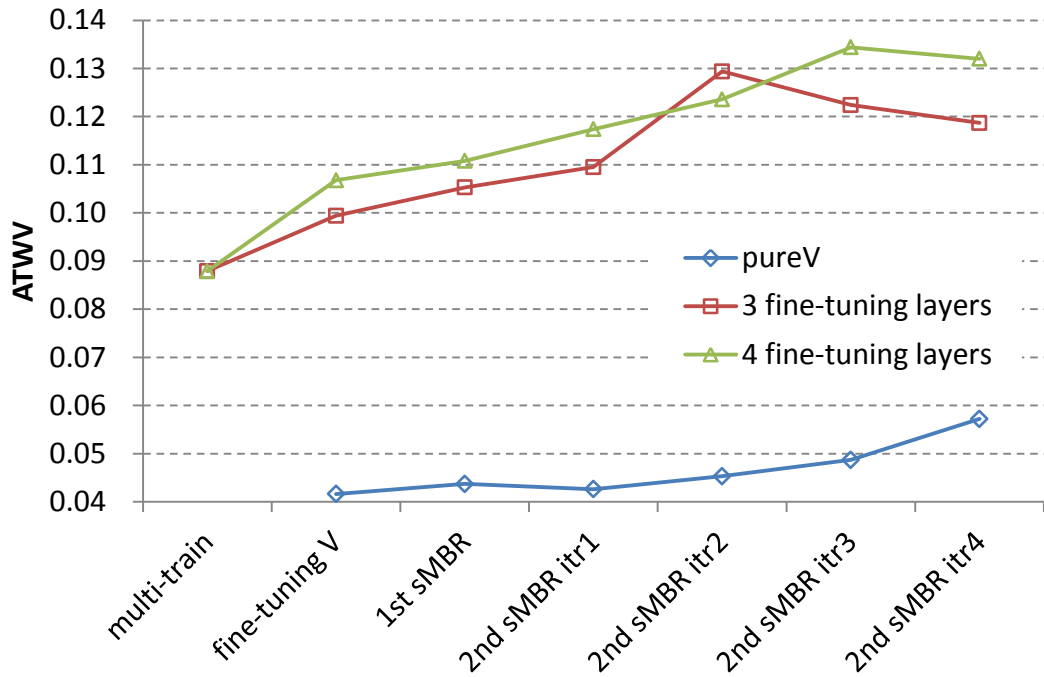


Figure 27. Vietnamese VLLP Systems' ATWV at different training stages in multilingual acoustic modeling recipes. Tested on the Vietnamese *dev2h* data.

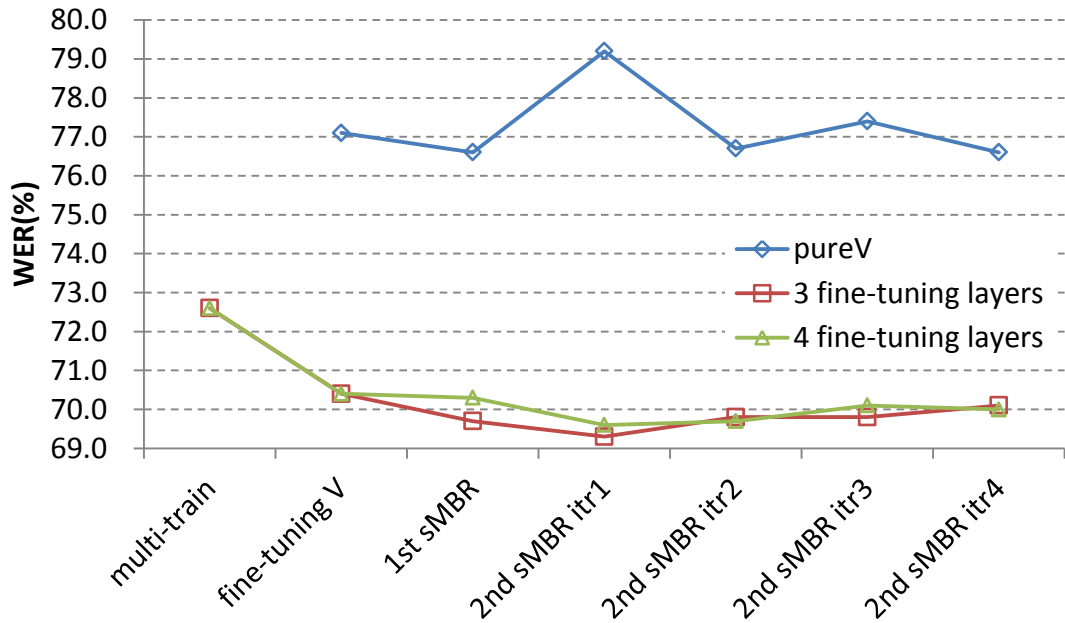


Figure 28. Vietnamese VLLP Systems' WER at different training stages in multilingual acoustic modeling recipes. Tested on the Vietnamese *dev2h* data.

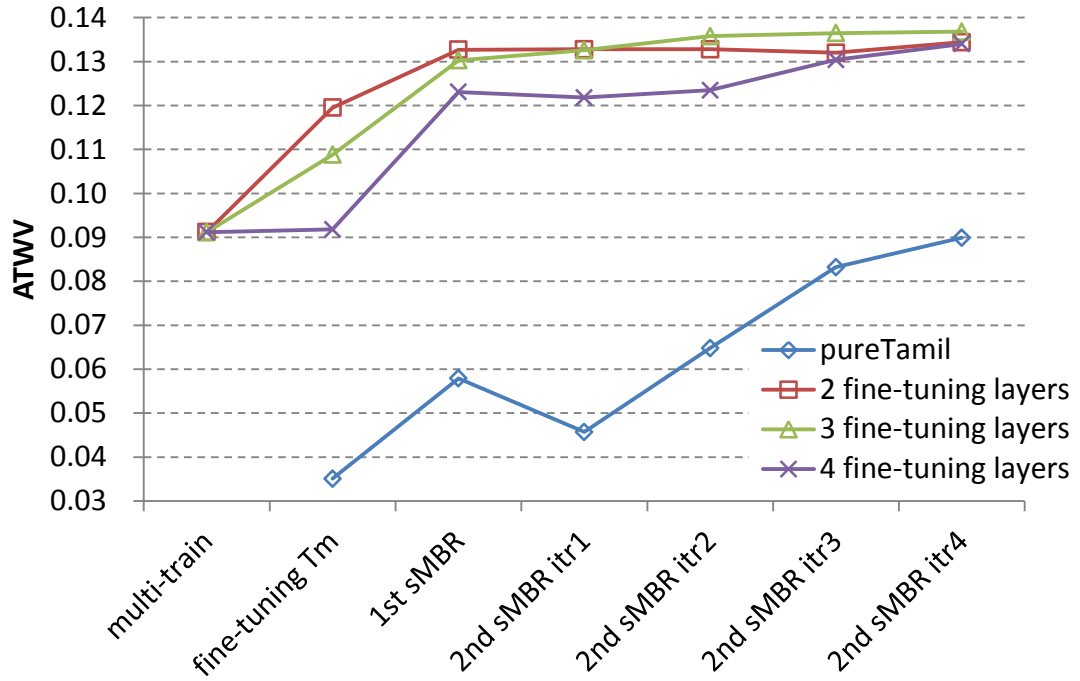


Figure 29. Tamil VLLP Systems' ATWV at different training stages in multilingual acoustic modeling recipes. Tested on Tamil *dev2h* data.

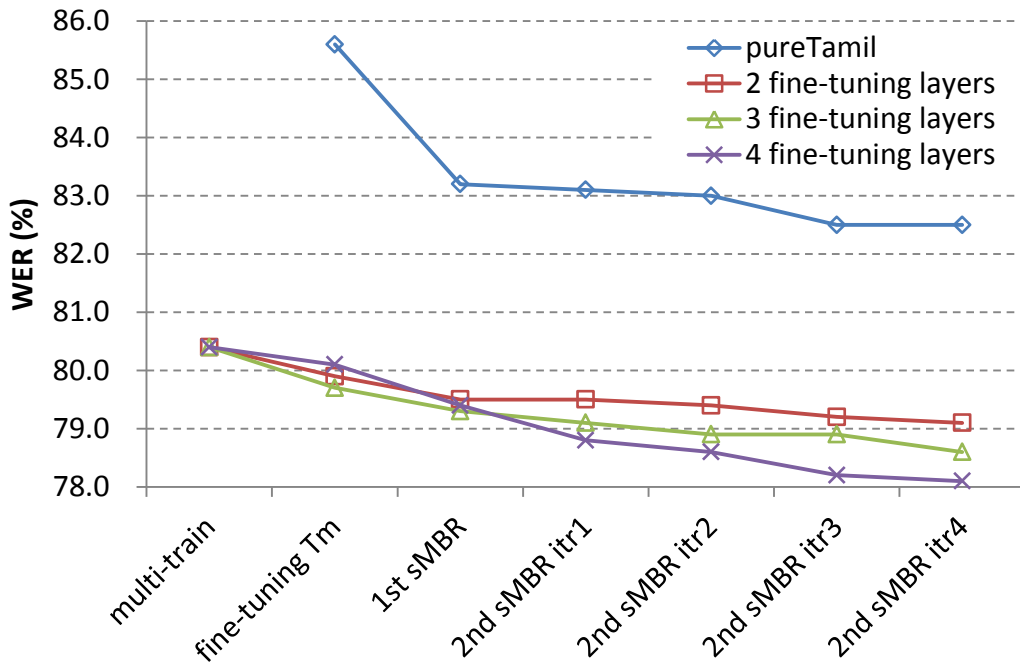


Figure 30. Tamil VLLP Systems' WER at different training stages in multilingual acoustic modeling recipes. Tested on Tamil *dev2h* data.

The ATWVs and WERs shown in Figure 25 to Figure 30 were evaluated on the Vietnamese and Tamil *dev2h* data sets. Table 29 shows the ATWVs of the multilingual and monolingual systems in the Vietnamese LLP and VLLP tasks on the *evalpart1* data. In both LLP and VLLP tasks, the multilingual systems outperformed the monolingual system significantly on the *evalpart1* data as well. The multilingual acoustic modeling process is especially effective and important to the VLLP task in which only very limited in-domain training resources are available for system building.

Table 29. The ATWVs of the monolingual and multilingual systems in the Vietnamese LLP and VLLP tasks on the *evalpart1* data. Multilingual acoustic modeling provided a significant performance improvement in the VLLP task.

ATWV [<i>evalpart1</i>]	Pure Vietnamese	Multilingual AM (System setup)
Vietnamese LLP	0.2006	0.2206 (C+P+Tu+Tg+Tm+V, 3 fine-tuning-layers, 2 nd sMBR itr4)
Vietnamese VLLP	0.0555	0.1253 (C+P+Tu+Tg+Tm+V, 4 fine-tuning layers, 2 nd sMBR itr3)

6.4.3.3 Influence of language distances in multilingual training

Despite the fact that there are thousands of languages in the world, many languages have close relationships with each other because of human population migration. In linguistics, these relationships among languages could be explained with the concept of *language family*. For example, both Cantonese and Mandarin Chinese are in the Sino-Tibetan language family, Vietnamese and Khmer (or Cambodian) are of the Austroasiatic language family, and Pashto and English are languages in the Indo-European language family. Languages in a same family have shorter distances to each other in terms of phonetic or syntactic structures.

In multilingual acoustic modeling, information on non-target languages is used to augment acoustic modeling of a target language. It is usually assumed that acoustic models of the target language can be better enhanced by selecting foreign languages which are similar to the target language for multilingual acoustic modeling. The goal of this experiment is to verify the assumption using the Vietnamese VLLP task.

In the experiment, five Vietnamese VLLP KWS systems with multilingual acoustic modeling were created. For each system, in addition to the 3-hour Vietnamese VLLP data, a 56-hour foreign language data set is used for the system AM to do multilingual training. Five assisting languages, i.e., Cantonese, Pashto, Turkish, Tagalog, and Tamil, provided by NIST in the Babel project are used here. The 56-hour data set of each assisting language is a subset of the FLP data set for the language. Table 30 shows details of the data sets used in the experiment. Language information including number of tones, spoken area, and language family are also presented. All the systems used a same multilingual acoustic modeling recipe to train system DNN AMs. The DNN structure contains 7 hidden layers and 1,200 node for each layer. In the multilingual training stage, a held out training set of the assisting language data was used as validation set to determine whether to halve the learning rate or terminate the training. In the fine-tuning stage, the 3-hour Vietnamese VLLP data was used to update the last 3 fine-tuning layers. System ATWVs and WERs are evaluated on the Vietnamese *dev2h* set.

Three distance measures are used to evaluate the distance between an assisting language to the target language, i.e., Vietnamese. The first measure is the language family distance between two languages. The distance was measured by checking whether the two languages belong to the same language family. Phonetic distance, which considers the similarity of sound units used in the two languages, is also used. The distance is measured as the number of overlapped IPA phones used in the two languages. The last measure is the acoustic distance between two languages. The distance provides the most fundamental distance measurement from the acoustic signal perspective and is evaluated as Kullback-Leiber 2 (KL2) distance between the acoustic feature distributions of the two languages.

In Table 30, it is clear that none of the assisting languages is in the same language family of Vietnamese. Table 31 shows the number of overlapped phones between Vietnamese and each assisting language. The smaller the number of the overlapped phones indicates the larger distance between the two languages. Both tonal and non-tonal (by removing tone IDs) Vietnamese phone sets are considered¹⁸, where tonal Vietnamese phone set has 236 phones and non-tonal phone set has 41 phones. The first row in Table 31 shows the number of overlapped tonal phones between Vietnamese and each assisting language. Since Cantonese is the only tonal language in the assisting languages, it has the most overlapped tonal phones with the Vietnamese. However, because the tone system in Cantonese is different from the system in Vietnamese, the tone ID in Cantonese is not necessary referring to the same tone in Vietnamese of the same ID. A more reasonable phonetic distance measure is to evaluate the overlap of the non-tonal phones. When considering non-tonal phone overlap, the Cantonese is the language having least overlapped phones with Vietnamese. There are only 14 non-tonal phones out of the 41 Vietnamese phones in the Cantonese phone set. On the other hand, Pashto and Tagalog have the most overlapping with Vietnamese phone set. In other words, from a phonetic perspective, Cantonese is the language having the largest distance with Vietnamese, while Pashto and Tagalog are the two assisting languages closest to Vietnamese.

To evaluate the acoustic distances between languages, for each language, a Gaussian mixture model (GMM) was trained with the training data of the language. For each pair of the GMMs, the KL2 distance between the two GMMs were used as the acoustic distance between the two corresponding languages. The acoustic features used in the GMMs are the FBank+F0 feature, which was the same feature used in multilingual DNN AMs. Table 32 shows the acoustic distances of all possible language pairs in the experiment. It is clear that Tagalog and Tamil are the two languages having the shortest acoustic distance with Vietnamese. Cantonese, on the other hand, is the language that having the longest acoustic distance with Vietnamese.

¹⁸ The lexicon used in the Vietnamese systems uses tonal phone set. The tonal phones are denoted by appending tone IDs, e.g. 1~6 for Vietnamese and 1~7 for Cantonese, to IPA phones.

Table 30. Details of the six languages data used in the experiment. Except for Vietnamese, which is the target language, the amount of training data for each of the 5 assisting languages was set to 56 hours for fair comparison. Information including the number of tones, the spoken area, and the language family of each language is also listed.

ID	Language (Babel ID)	Hours	#Tone	Spoken Area (Language Family)
C_56hr	Cantonese (101)	56	7	Gaungdong & Guangxi Provinces of China (Sino-Tibetan)
P_56hr	Pashto (104)	56	0	Afghanistan & North West Frontier Province of Pakistan (Indo-European)
Tu_56hr	Turkish (105)	56	0	Turkey (Turkie)
Tg_56hr	Tagalog (106)	56	0	Philippines (Austronesian)
Tm_56hr	Tamil (204)	56	0	India (Dravidian)
V_3hr	Vietnamese (107)	3	6	Vietnam (Austroasiatic)

Table 31. Number of overlapped phones for the 5 assisting languages with Vietnamese. Both tonal and non-tonal (by removing tone tags) Vietnamese phone sets are considered, where tonal Vietnamese phone set has 236 phones and non-tonal phone set has 41 phones. The one overlapped phone among Vietnamese tonal phone set with Pashto, Turkish, Tagalog, and Tamil is the phone /m/.

# Overlapped Phone with Vietnamese	Cantonese	Pashto	Turkish	Tagalog	Tamil
w/ Tone (236)	82	1	1	1	1
w/o Tone (41)	14	23	22	23	19

Table 32. Acoustic distances between pairs of the training languages. The acoustic distance is measured by evaluating the KL2 distances between the acoustic feature distributions of the languages. Both V_3hr and V_78 are for Vietnamese language. V_3hr used the 3-hour Vietnamese VLLP data to build the Vietnamese GMM for KL2 distance evaluation, while V_78hr used all the Vietnamese FLP data to build the language GMM.

KL2D	C_56hr	P_56hr	Tu_56hr	Tg_56hr	Tm_56hr	V_3hr	V_78hr
C_56hr	0	20.9	14.5	14.2	17.3	16.9	16.5
P_56hr	20.9	0	12.9	15.0	13.8	14.2	12.7
Tu_56hr	14.5	12.9	0	11.7	13.8	14.9	13.3
Tg_56hr	14.2	15.0	11.7	0	13.1	13.7	11.8
Tm_56hr	17.3	13.8	13.8	13.1	0	13.5	11.7
V_3hr	16.9	14.2	14.9	13.7	13.5	0	11.2
V_78hr	16.5	12.7	13.3	11.8	11.7	11.2	0

Figure 31 shows the WERs of the six Vietnamese VLLP systems on the Vietnamese *dev2h* data set. It is clear that the Vietnamese VLLP systems benefit from all the five assisting languages. However, different assisting languages did provide different scale of WER reduction. Systems assisted by Tamil and Tagalog, which are the two languages having the shortest acoustic distance to Vietnamese, achieved the largest WER reduction over the baseline VLLP system. On the other hand, the WER reduction of the Cantonese system is the smallest among the five languages. The observation is consistent with the acoustic distances of the five assisting languages to Vietnamese.

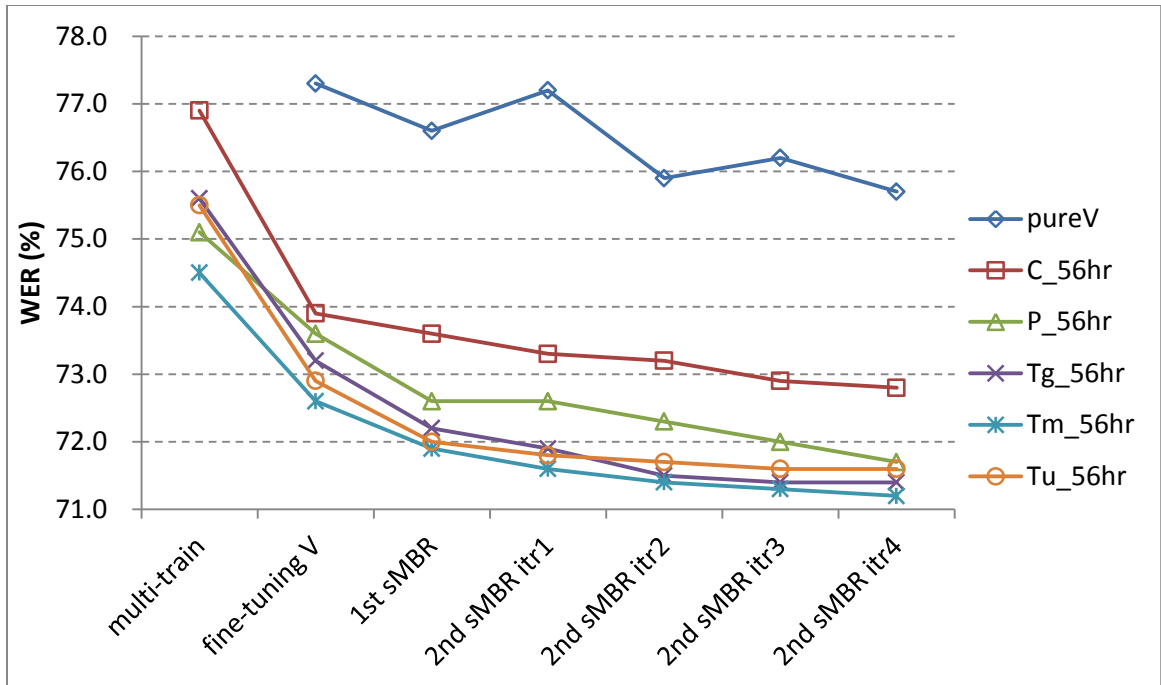


Figure 31. WER of the 5 bilingual trained Vietnamese VLLP systems at each training stage on the Vietnamese *dev2h* set. Tagalog and Tamil assisted systems are the two systems with the best WER performance.

For ATWVs (shown in Figure 32), the improvement trend of the five languages is slightly different from WERs. Though Cantonese has the largest acoustic distance with Vietnamese among the five languages, the Cantonese system achieved similar ATWVs of the Turkish system and outperformed the Pashto system. A plausible reason is that since ATWV only considers partial test data, i.e., the keywords, it is more keyword list dependent. It is possible that the words in the Vietnamese evaluation keyword list were the words that the Cantonese system has high detection accuracy. Therefore, the Cantonese system achieved a better ATWV than Pashto though it attained the worst WER among all.

In summary, acoustic distance is a best distance measure for selecting assisting language for a target language ASR and WER system. Experimental results show the measure provides good prediction for WER performance and reasonable prediction for ATWV performance after the multilingual acoustic modeling.

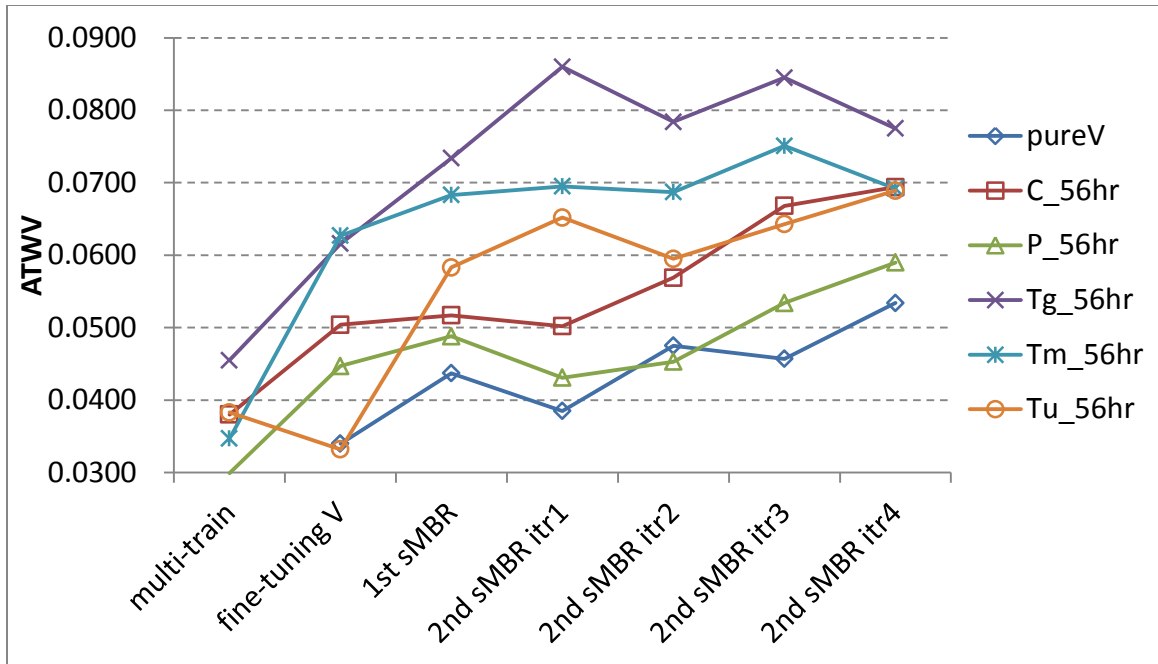


Figure 32. The ATWVs of the 5 bilingual trained Vietnamese VLLP systems at each training stage on the Vietnamese *dev2h* set. Tagalog and Tamil assisted systems are the two systems with the best ATWV performance.

6.4.3.4 Combine with keyword-aware language modeling

The VLLP systems with multilingual trained AMs can be further improved by using the keyword-aware LM. Table 33 shows the ATWV performance of Vietnamese VLLP systems with original 3-hour transcription trained 3-gram LM and the CS-KWLM Interpolated 3-gram LM. It is clear that the improvement of multilingual AM and KW-aware LM is additive. By adopting the multilingual AM, the Vietnamese VLLP achieved 126% relative ATWV improvement (from 0.0555 to 0.1253). A further 76% relative ATWV performance improvement (from 0.1253 to 0.2208) can be attained by using the KW-aware LM.

Table 33. ATWV performance of Vietnamese VLLP systems with multilingual AM and KW-aware LM on evalaprt1.

Acoustic Model (Model id)	Original LM	CS-KWLM Interpolation ($\alpha = 0.4$)
pure Vietnamese VLLP AM (pureV_2nd_smbr_it4)	0.0555	0.0927
Multilingual AM (CPTuTgTm_4ftl_2nd_smbr_it3)	0.1253	0.2208

6.4.4 Web text augmented language modeling

6.4.4.1 Setup

Experiments for web-text augmented language modeling were conducted on the Vietnamese LLP and VLLP tasks. The Vietnamese LLP baseline is the baseline used in Chapter 4 and 5, where the acoustic model is sMBR trained DNN AM with BNF+fMLLR features. The Vietnamese VLLP baseline is the multilingual Vietnamese VLLP system presented in section 6.4.3.2. Details of the acoustic models for the two baseline systems are shown in Table 34.

Table 34. Acoustic model configurations of the Vietnamese LLP and VLLP systems in the experiment.

Vietnamese Task	Acoustic Model	Description
LLP	Pure Vietnamese	10 hour Vietnamese training data trained DNN AM with sMBR training
VLLP	C+P+Tu+Tg+Tm+V Multilingual	402 hour Multilingual data trained DNN AM with fine-tuning shown in section 6.4.3.2 with sMBR training

Table 35 shows the details of the eight LM training text sources used in the experiments. The amounts of in-domain training data are very limited for both VLLP and LLP tasks, in which only 213 Kb and 718 Kb transcriptions are available respectively. For the web data, the Wikipedia and Query are the two sources with largest amount of training text. The file sizes of the Wikipedia and Query data are 601 Mb and 528 Mb, respectively. Data sizes of the rest of the web sources are much smaller at about 1/10 of the Query data amount. Among the six web sources, the Wiktionary data contains only 31,217 Vietnamese phrases and has the smallest file size of 351Kb.

Table 35. Details of the two in-domain and six web-text sources used in the experiments.

Text Source	Amount of Data		Data Description
	# Words	File Size	
VLLP	35,832	213 Kb	VLLP task 3 hour transcription (in-domain data)
LLP	117,573	718 Kb	LLP task 10 hour transcription (in-domain data)
Wiktionary	60,978	351 Kb	31,217 Vietnamese phrases listed presented in the Vietnamese Wiktionary
Wikipedia	100,643,820	601 Mb	Vietnamese Wikipedia content
Open Subtitle	8,910,907	48 Mb	OpenSubtitles 2012 and 2013 Vietnamese movie subtitles (1848 movies)
TED	3,374,096	20 Mb	Vietnamese TED talk transcriptions
Blogspot	8,771,565	52 Mb	Vietnamese Blog text
Query	91,647,282	528 Mb	Web text data retrieved by querying the web search engine using the top 220 n -grams from the Vietnamese VLLP transcripts

6.4.4.2 Keyword coverage analysis

The keyword coverage rates of the eight LM training sources used in the experiments are listed in Table 36. The first two rows show the keyword coverage rates for the LM

training text of the VLLP and LLP baseline systems. Note that though the LLP training data size is only about three times of the VLLP data, the ratio of the keyword coverage rates of LLP data to VLLP data is about ten. The significant coverage rate drop for the VLLP training text shows the severity of the information loss in the VLLP task. Thus, it is crucial to the VLLP system to utilize the out-of-domain web data to recover the lost information. For the six web data sources, all the sources provide better keyword coverage rates than the VLLP data. The Query data has the highest keyword coverage rate at 79%, which is very close to the coverage rate of the LLP training data. Interestingly, though the data size of Wikipedia is larger than the Query data, the keyword coverage rate of Wikipedia is smaller than the coverage rate of the Query data. This shows that the method employed in the Query data retrieval is a more efficient way to collect domain related training data for KWS tasks.

Table 36. Evaluation keyword coverage rates of the 8 training text sources used in the experiment.

Vietnamese Text source	# covered evaluation keywords (4065 in total)	Coverage (%)
VLLP task 3 hr transcription	332	8 %
LLP task 10 hr transcription	3265	80 %
Wiktionary	979	24 %
Wikipedia	2873	71 %
Open Subtitle	1877	46 %
TED	1524	37 %
Blog	2044	50 %
Query	3226	79 %

6.4.4.3 Language model perplexity analysis

Language model perplexity (PPL) is another common measure for evaluating the quality of web data. A web-data augmented LM with a lower perplexity on the test data is a

model better predicting word sequences in the test data. Perplexities of LMs on the evaluation keyword list and the evaluation transcriptions therefore can be indicators of how well the language models predict keyword or word sequences in the evaluation data for KWS and ASR tasks. Table 37 shows the perplexities of seven LMs in the LLP task on the evaluation keyword list and transcriptions. The interpolation weight, α_{web} , for each web-data augmented LM is tuned by selecting the α_{web} providing the best system ATWV on *dev2h* data. The first row in Table 37 shows the perplexity of the original LLP LM on the keyword list and evaluation utterances. It is clear that though the original LLP LM has very low perplexity on the evaluation utterance at 150, its perplexity on the keyword list is quite high at 611. The difference is because of the fact that most of the keywords are sequences of content words, which are less frequent than function words even in the in-domain training data. LMs thus often lack enough training samples to estimate correct n -gram probabilities in the keywords. The web-text can be used to alleviate this lack-of-training data problem for keyword sequences. In Table 37, it is obvious that all the perplexities of the web-text augmented LMs on the keyword list are significantly smaller than the perplexity of the original LLP LM. Wiktionary augmented LLP LM achieved the lowest PPL on keyword list at 166. However, when considering both KWS and ASR tasks, the Query augmented LMs seems to be the best one since it achieved low PPL on both keyword list and evaluation utterances.

Table 37. Language model perplexities on keyword list and evaluation sentences for the LMs used in the Vietnamese LLP experiments.

Language Models for LLP Task	PPL on keyword list	PPL on eval utts
Original LLP LM	611	150
Wiktionary augmented LLP LM ($\alpha_{web}=0.7$)	166	290
Wikipedia augmented LLP LM ($\alpha_{web}=0.6$)	240	200
OpenSubtitle augmented LLP LM ($\alpha_{web}=0.6$)	252	170
TED augmented LLP LM ($\alpha_{web}=0.4$)	335	160
Blog augmented LLP LM ($\alpha_{web}=0.2$)	313	140
Query augmented LLP LM ($\alpha_{web}=0.6$)	174	180

Table 38 shows the perplexities of the LMs in the VLLP task on the evaluation keyword list and transcriptions. Note that the interpolation weights, α_{web} , for the web LMs in the VLLP task are much smaller than the weights used in the LLP task. This is because the original VLLP LM was poorly trained due to the very small amount of in-domain training text. The inadequately trained VLLP LM made the interpolated LMs easily over-fit the web-data and deviate from the original task domain when large interpolation weight, α_{web} , is used.

From Table 35 we know that the lack-of-training-data problem is more serious in the VLLP task than in the LLP task. Fortunately, the problem can be alleviated by using web data. For the PPL performance, all the web-data augmented LMs achieved lower PPL than the original VLLP LM on both the keyword list and the evaluation utterances. The Query augmented LM again attained the best overall PPL performance among all the six web-data augmented LMs.

Table 38. Language model perplexities on keyword list and evaluation sentences for the LMs used in the Vietnamese VLLP experiments.

Language Model for VLLP Task			PPL on keyword list	PPL on eval utts
Original VLLP LM			684	202
Wiktionary	augmented VLLP LM	($\alpha_{web}=0.1$)	353	190
Wikipedia	augmented VLLP LM	($\alpha_{web}=0.2$)	312	183
OpenSubtitle	augmented VLLP LM	($\alpha_{web}=0.1$)	408	175
TED	augmented VLLP LM	($\alpha_{web}=0.2$)	380	181
Blog	augmented VLLP LM	($\alpha_{web}=0.3$)	300	182
Query	augmented VLLP LM	($\alpha_{web}=0.3$)	214	177

6.4.4.4 System performance

Table 39 and Table 40 show the ATWVs and WERs of the seven LLP systems on the Vietnamese *evalpart1* data. Consistent with the PPLs on the evaluation keyword list

observed in section 6.4.4.3, all the web-data augmented systems achieved higher ATWV than the baseline system at 0.2089. The OpenSubtitle and Query assisted LMs provided the best ATWVs at 0.2429 and 0.2422 respectively. However, comparing the LM PPLs in Table 37 and the system ATWVs in Table 39, it is clear that, though there are some correlations between the two performance measures, the correlation between PPL and ATWV is not very high. The Wiktionary assisted LM, which had the lowest PPL on keyword list, had the smallest ATWV improvement over the baseline system among the six web-data sources. While the OpenSubtitle augmented LM, which obtained the best ATWV, received considerably high PPL comparing to the other web-data augmented LMs. This shows that PPL can only be used as a rough guide for web data selection in KWS task. For more accurate data selection, a real KWS process must be involved.

For WER, all the web-data augmented LMs except Blog assisted LM had higher WER than the baseline. The result is predictable from Table 37.

Table 39. The ATWVs of the web text augmented LM systems on the *evalpart1* data in the Vietnamese LLP task.

ATWV	Wiktionary ($\alpha_{Web}=0.7$)	Wikipedia ($\alpha_{Web}=0.6$)	OpenSubtitle ($\alpha_{Web}=0.6$)	TED ($\alpha_{Web}=0.4$)	Blog ($\alpha_{Web}=0.2$)	Query ($\alpha_{Web}=0.6$)
Baseline	0.2089					
Web Text Int	0.2186	0.2363	0.2429	0.2365	0.2379	0.2422

Table 40. The WERs of web text augmented LM systems on *evalpart1* data in the Vietnamese LLP task. (LMWT=12)

WER (%)	Wiktionary ($\alpha_{Web}=0.7$)	Wikipedia ($\alpha_{Web}=0.6$)	OpenSubtitle ($\alpha_{Web}=0.6$)	TED ($\alpha_{Web}=0.4$)	Blog ($\alpha_{Web}=0.2$)	Query ($\alpha_{Web}=0.6$)
Baseline	64.7 %					
Web Text Int	68.4 %	66.8 %	65.8 %	65.5 %	64.7 %	65.7 %

Similar results are observed in the VLLP task as shown in Table 41 and Table 42. In Table 41, all the web-data augmented systems achieved better ATWVs against the

baseline system. The Wikipedia, Blog, and Query are the three web sources attained the best ATWVs at 0.1502, 0.1495, and 0.1477 respectively. From Table 39 and Table 41, it is also clear that the Query data provided the most stable ATWV performance enhancement in LLP and VLLP tasks among all the six web sources.

Table 42 shows the WER of the VLLP systems. All the web-data augmented LMs except Wiktionary assisted LM had better WER than the baseline system. The results are to some extent predictable from the LMs' PPL on the evaluation utterances as shown in Table 38.

Table 41. The ATWVs of web text augmented LM systems on *evalpart1* data in the Vietnamese VLLP task.

ATWV	Wiktionary ($\alpha_{Web}=0.1$)	Wikipedia ($\alpha_{Web}=0.2$)	OpenSubtitle ($\alpha_{Web}=0.1$)	TED ($\alpha_{Web}=0.2$)	Blog ($\alpha_{Web}=0.3$)	Query ($\alpha_{Web}=0.3$)
Baseline	0.1257					
Web Text Int	0.1369	0.1502	0.1423	0.1405	0.1495	0.1477

Table 42. The WERs of web text augmented LM systems on *evalpart1* data in the Vietnamese VLLP task.

WER (%)	Wiktionary ($\alpha_{Web}=0.1$)	Wikipedia ($\alpha_{Web}=0.2$)	OpenSubtitle ($\alpha_{Web}=0.1$)	TED ($\alpha_{Web}=0.2$)	Blog ($\alpha_{Web}=0.3$)	Query ($\alpha_{Web}=0.3$)
Baseline	71.4 %					
Web Text Int	71.5	71.2	71.1	71.2	71.4	70.9

6.4.4.5 Combine with keyword-aware language modeling

In Chapter 4, a keyword-aware language modeling framework is proposed. The proposed framework and web-text augmented language modeling are both techniques utilizing supplementary information in addition to the original LM training texts to enhance LM performance in KWS systems. The major difference between the two techniques is that the keyword-aware language modeling is making use of in-domain keyword information

for the enhancement; while the web-text augmented LM is using out-of-domain text data to improve the system LMs. Since the information sources exploited by these two techniques are independent, it is possible to combine the two techniques to further improve LMs in the KWS systems.

In this experiment, the combination of the two language modeling techniques is investigated. The performance of KWS systems with six different lexicon and LM configurations is also summarized in this section. The six configurations are:

- (a) "Orig Lex + LM", which is the setting of the baseline system using the original lexicons and LMs of the task,
- (b) "Orig Lex + WebText LM Int", where the system lexicon is the original task lexicon, but the system LMs are web-data augmented LMs,
- (c) "G2P Lex + Orig LM", in which OOV words used by the evaluation keywords are added to the system lexicon with G2P estimated pronunciations, while the LM is still trained with the original training text of the task with the expanded vocabulary,
- (d) "G2P Lex + WebText LM Int", where the G2P lexicon in configuration (c) is used, but the system LMs are web-text augmented LMs built with the expanded vocabulary,
- (e) "G2P Lex + CS-KWLM Int", in which the G2P lexicon in configuration (c) is used, and the system LM is the CS-KWLM interpolated LM proposed in Chapter 4.
- (f) "G2P Lex + WebText LM & CS-KWLM Int", where the G2P lexicon in configuration (c) is used, and the system LM is a keyword-aware LM built on top of the web-text augmented LM in configuration (d).

Note that the configuration (f) combines the two language modeling techniques, i.e., web-text augmented language modeling and keyword-aware language modeling, for LM construction. In this experiment, Wikipedia and Query are selected as the two data sources for the web-text augmented LMs. Table 43 and Table 44 shows the nine system WERs and ATWVs on the *evalpart1* data in the LLP and VLLP tasks respectively.

The first column in Table 43 shows the WER performance of the systems in the LLP task. Since the LM interpolation weights, i.e., α_{Web} and $\alpha_{CS-KWLM}$ in systems (b), (d), (e),

and (f), for web-text augmented LMs and keyword-aware LMs are tuned to optimize ATWV instead of WER performance, the LM-enhanced systems achieved better ATWVs with the price of higher WER. The systems with not enhanced-LMs, namely system (a) and (c), therefore had the lowest WER at 64.8%. The ATWV performance of the nine LLP systems are shown in the second column of Table 43. When no keyword information is available to the KWS systems, i.e., configurations (a) and (b), it is clear that the web-text augmented LMs could provide noteworthy ATWV improvement. The ATWV of the baseline system (a) at 0.2083 was improved to 0.2404 and 0.2442 for the Wikipedia and Query augmented systems respectively. Note that the performance difference between Wikipedia and Query systems in configuration (b) is quite small even though the keyword coverage rate for Query data is much higher than the rate of Wikipedia as shown in Table 36. This is because many keywords contain OOV words and cannot be handled by the KWS systems. The OOV words also make the correct probabilities of the keywords cannot be learned by LMs even when training samples of the keywords are provided by the web data. The problem can be alleviated by adding these OOV words to the system vocabulary, i.e., configurations (c) and (d). By expanding the system lexicon, both Wikipedia and Query data augmented systems received further ATWV improvements, i.e., from 0.2404 to 0.2671, and from 0.2442 to 0.2773, respectively. The performance difference between the Query and the Wikipedia system is also much larger which correctly reflects the keyword coverage rate of the data. When complete keyword information is available to the KWS systems, i.e., configurations (e) and (f), keyword-aware language modeling can be used to further provide remarkable ATWV improvements. The keyword-aware language modeling can be applied directly on the original LM, i.e., the LM in (c), or on the web-text augmented LMs, i.e., the LMs in (d). Interestingly, though the web-text augmented LMs are better than the original LM in terms of the ATWV performance, using the web-text augmented LMs for keyword-aware language modeling led to worse ATWVs than using the original LM in the LLP task. A possible reason is that, in the LLP task, the original LM is already good enough to provide complementary information to the CS-KWLM and form a good interpolated LM for KWS tasks. On the other hand, the web-text augmented LMs introduced additional

out-of-domain noises to the keyword-aware language modeling process and thus degraded the overall ATWV performance.

Table 43. The WERs and ATWVs of the Vietnamese LLP systems with different lexicon and language model configurations on *evalpart1* data.

Systems	WER (%)		ATWV	
	Wikipedia	Query	Wikipedia	Query
(a) Orig Lex + LM	64.8		0.2083	
(b) Orig Lex + WebText LM Int ($\alpha_{Web} = 0.6/0.6$)	65.7	65.4	0.2404	0.2442
(c) G2P Lex + Orig LM	64.8		0.2098	
(d) G2P Lex + WebText LM Int ($\alpha_{Web} = 0.6/0.6$)	65.6	65.4	0.2671	0.2773
(e) G2P Lex + CS-KWLM Int ($\alpha_{CS-KWLM} = 0.6$)	66.0		0.3287	
(f) G2P Lex + WebText LM & CS-KWLM Int ($\alpha_{Web} = 0.6/0.6, \alpha_{CS-KWLM} = 0.3$)	66.1	66.1	0.3197	0.3126

With much fewer in-domain training data for the VLLP system LMs, the trend of the VLLP system performance is slightly different from the LLP task. The first column of Table 44 shows the WER of the nine systems in the VLLP task. Unlike the LLP task, the systems with web-text augmented LMs, i.e., the systems with configuration (b) and (d), achieved better WER performance than the baseline systems (a) and (c). The Query data assisted LM attained a lowest WER at 70.9%. For ATWVs shown in the second column of Table 44, it is clear that web-data is helpful in all configurations even when keyword information is available to the KWS system. The best ATWV among all the nine systems is achieved by the system integrating the Query assisted LM with keyword-aware language modeling and is at 0.2254, which is slightly higher than the 0.2212 attained by the pure keyword-aware LM system in the VLLP task.

In summary, when there is no keyword information available to KWS systems, i.e., configuration (a) and (b), using web-text augmented LM is always helpful for ATWV performance improvement. The effect of web-data augmented language modeling could be further enhanced by enriching system lexicons with vocabulary used by potential evaluation keywords, i.e., configuration (c) and (d). However, when keyword information is available to KWS systems, i.e., configuration (e) and (f), the web-data become less helpful and can be ignored without much harm to the system performance when keyword-aware language modeling is applied.

Table 44. The WERs and ATWVs of the Vietnamese VLLP systems with different lexicon and language model configurations on *evalpart1* data.

Systems	WER (%)		ATWV	
	Wikipedia	Query	Wikipedia	Query
(a) Orig Lex + LM	71.4		0.1257	
(b) Orig Lex + WebText LM Int ($\alpha_{Web} = 0.2/0.3$)	71.2	70.9	0.1502	0.1477
(c) G2P Lex + Orig LM	71.3		0.1326	
(d) G2P Lex + WebText LM Int ($\alpha_{Web} = 0.2/0.3$)	71.2	70.9	0.1543	0.1675
(e) G2P Lex + CS-KWLM Int ($\alpha_{CS-KWLM} = 0.4$)	71.9		0.2212	
(f) G2P Lex + WebText LM & CS-KWLM Int ($\alpha_{Web} = 0.2/0.3, \alpha_{CS-KWLM} = 0.3$)	72.0	71.8	0.2231	0.2254

6.5 Conclusion

In this chapter, two important techniques utilizing out-of-domain data for acoustic and language model enhancement, i.e., *multilingual acoustic modeling* and *web text augmented language modeling*, in KWS systems are investigated. For multilingual acoustic modeling, a revised training recipe of the method presented in [19] is proposed

for DNN AMs. In the revised recipe, in addition to the output layer, parameters in the last N hidden layers of the DNN AMs can also be updated during AM fine-tuning with target language data. This relaxation allows the DNN AMs to learn the target language better and to achieve better KWS performance. For web text augmented language modeling, six different web sources are used for the study of data selection.

Experimental results on OpenKWS LLP and VLLP tasks show that acoustic model enhancement is more important than language model improvement when very few in-domain training data is available to KWS systems. The results also show that multilingual acoustic modeling is a promising approach for acoustic model enhancement. In both LLP and VLLP tasks, the multilingually trained AMs consistently provided significant ATWV improvements for the KWS systems. The improvement is especially prominent when the amount of training data for target language is extremely small. Further, the acoustic distance between languages can be a gauge indicating how well a foreign language can help acoustic modeling for a target language. Despite the fact that using data from all available foreign languages provides better AMs than using partial foreign data for multilingual training, this language distance can still help developers to decide what foreign languages should be considered in the early data collection phase of the KWS system building process. The effects of multilingual acoustic modeling and the keyword-aware language modeling proposed in Chapter 4 is also additive.

For language modeling, the Query data, which is retrieved by searching high frequency n -grams in the original LM training text on the Internet, has been shown to be the best web source for LM augmentation. The web text augmented language modeling provides significant performance improvement when there is no keyword information available to KWS systems. However, when keyword information is available, web data has very little help to the KWS performance in the keyword-aware language modeling framework.

CHAPTER 7 CONCLUSION

7.1 Summary

In this dissertation, a resource-dependent acoustic and language modeling framework is proposed and studied for KWS. Depending on the training resources available to a KWS system, three different strategies and research directions, i.e., the use of *data-efficient training process*, the utilization of better *system optimization objectives*, and the employment of *data augmentation*, are investigated. The research directions attempt to reduce the two major problems, i.e., the used of incorrect models and training/test mismatch, in today's KWS modeling framework and to enhance KWS performance. The first two strategies are useful when keyword information is available to the KWS systems. While the third strategy is suitable for systems with accessible out-of-domain data.

In each research direction, both acoustic and language model enhancement techniques are proposed and studied. Detailed analyses of techniques in the three research directions and their combination are also conducted to provide comprehensive knowledge of building KWS systems with good performance. In Chapter 3 and 4, resource-dependent keyword modeling and keyword-aware language modeling are proposed for acoustic and language models respectively in the *data-efficient training process* direction. The resource-dependent keyword modeling exploits a "*share & split*" process to optimize the GMM-HMM based AMs of the keywords toward the keywords' pronunciations at each phonological level with efficient uses of system training data. While the keyword-aware language modeling approach employs keyword information to alleviate the "prior underestimation problem" for multi-word keywords and provides remarkable KWS performance improvement. It is also important to note that the proposed keyword-aware language modeling framework provides a bridge to connect the keyword-filler based KWS and LVCSR-based KWS frameworks theoretically. The two conventional KWS frameworks can be consider as special cases of the proposed KW-aware LM framework. By controlling the keyword priors in the proposed KW-aware LM based KWS, we can easily adjust the missed detection and false alarm rates of the systems. The KWS systems

of the proposed framework thus have much more operation points available comparing to the conventional two KWS frameworks and suit for more application scenarios.

For the utilization of better *system optimization objectives*, Chapter 5 explores the keyword-boosted sMBR training for DNN AMs and discriminative keyword-aware language modeling. The keyword-boosted sMBR training shows consistent improvement over the baseline system in all test configurations. However, the conventional discriminative language modeling techniques, which are popular in ASR tasks with MAP decoding, do not work well in KWS tasks due to the incorrect estimation of word arc posterior probabilities in decoding lattices provided by the discriminative LMs.

Chapter 6 exams two major *data augmentation* approaches to acoustic and language modeling in ASR and KWS systems today. For acoustic modeling, a multilingual acoustic modeling recipe for DNN AMs is proposed. The utilization of foreign language data for acoustic modeling significantly improves the KWS performance, especially when the amount of in-domain training data is small. The success of the proposed training recipe, which includes a multilingual acoustic modeling (shared hidden layers) and target language fine-tuning (split the DNN AM from other tasks), confirms again the concept of "*share & split*" is very effective and can be used in most of the parameter estimation processes. For language modeling, web text augmented language modeling provides substantial improvement to KWS systems, especially when keyword information is unavailable.

Among the three research directions, *data-efficient training* and *data augmentation* are the two strategies providing most significant KWS performance improvement. These two strategies directly address the problem of distribution mismatch in training and test conditions for the KWS system by making efficient use of keyword information and out-of-domain data respectively. On the contrary, utilizing better *system optimization objective* functions provides smaller improvement since the direction aims to enhance system performance by diminishing the damage of choosing incorrect modes (i.e., HMM and *n*-grams) to describe speech signals and the mismatch between the model training objectives and the KWS performance metrics. However, it could still be a key to the

further performance boost of KWS systems after applying techniques in the other two research directions.

For the importance of AM and LM enhancement, the study in section 6.4.2 shows that having a good acoustic model is more important than acquiring a good LM when the amount of training data is small. This is because AMs are playing an important role to provide fundamental information directly relating to the input speech signals in the decoding phase. If serious errors were made by a poorly trained AM in the low-level decoding stage, it would be difficult for LMs to recover the error in the higher-level stage. A bad AM would also diminish the effect of LM enhancement. Therefore, improving AM should always be the top priority in the construction of a KWS systems in any condition.

From the results in this dissertation, a system-building process for KWS systems can therefore be concluded:

When keyword information is not available to a KWS system, the system should first train its AM with multilingual acoustic modeling and other data augmentation approaches to enhance the AM performance if possible. The web-text augmented language modeling process can then be applied to this KWS system with multilingually trained AM to further improve the system performance.

When keyword information is available to a KWS system, the multilingual acoustic modeling process and other data augmentation approaches should be the first training process still to initialize the system AM. The AM can then be fine-tuned with data-efficient keyword modeling and keyword-boosted sMBR training processes. After the system obtains the final enhanced AM, a keyword-aware LM, e.g., a CS-KWLM interpolated LM, can then be trained to work with the enhanced AM.

7.2 Dissertation Contributions

A number of novel contributions to the field of KWS are provided in this dissertation:

1. A comprehensive blueprint of the enhancement of acoustic and language models in KWS systems: three enhancement directions and their combination have been investigated. It can be a guide for future designers to build high performance system acoustic and language models in KWS systems.

2. A thorough investigation of the "keyword prior underestimation" problem for n -gram LMs in LVCSR-based KWS systems: the dissertation shows that the underestimation problem severely degrades detection performance of LVCSR-based KWS systems for multi-word keywords. It also identifies the causes of the problem and provides methods to avoid the problem.
3. The development of a novel "*share & split*" procedure for keyword acoustic modeling: the new training procedure allows keyword AMs to be optimized toward their own pronunciation according to the amount of their training data.
4. The development of the novel keyword-boosted sMBR acoustic modeling process and the study of discriminative keyword-aware language modeling for KWS systems: the new discriminative acoustic modeling process provides significant performance improvement for KWS tasks. The issues of applying discriminative language modeling to KWS systems are also discovered.
5. A novel multilingual acoustic modeling recipe for DNN AMs: detailed studies of combining the proposed keyword-aware language modeling approaches with popular used data augmentation modeling methods are provided.

7.3 Possible Extensions and Improvements

The dissertation provides a comprehensive study of acoustic and language modeling in ASR-based KWS systems. However, there are still some improvements that can be future works extending the research in this dissertation.

In Chapter 3, the "*share & split*" process is proposed only for GMM-HMM based acoustic models. Since in DNN AMs the same parameter sharing mechanism (i.e., senone-level and phone-level sharing) is also used, the idea of the "*share & split*" process could still be applied. However, unlike GMM-HMM based AMs, DNN AMs are discriminative models which have a complete different model adaptation framework from generative models. Realizing the "*share & split*" process in this new framework would be an interesting research.

Another potential research is the use of keyword-specific boosting weight, β , for each keyword word in the keyword-boosted sMBR training. In the current keyword-boosted

sMBR training, all the keyword words share the same boosting weight in the model training phase. However, in Figure 19 and Figure 21, it is clear that the curves of ATWV performance to the global boosting weight, β , have multiple local maximums. In other words, the best boosting weight is in fact different for different groups of keywords. If keyword-specific boosting weight can be used, it is possible to allow every keyword to achieve its best KWS performance and further improve the overall performance. The difficulty of applying the keyword-specific boosting weight would be the weights tuning. Since most of the keywords are competing with each other in the decoding phase, the performance of each keyword is not independent. Changing β for a keyword might also influence the detection performance of another keyword. Therefore, adjusting the boosting weight, β , for each keyword word appropriately to optimize the overall system performance would be an interesting issue to study.

Finally yet importantly, it is shown in Chapter 5 that the GLM-based discriminative language modeling (DLM) approach for ASR tasks is not suitable for KWS tasks because the method is designed for conventional MAP decoding. In MAP decoding, only the word sequence with highest log-likelihood is considered; most of the DLM approaches therefore ignore the sum-to-one constraint when training the n -gram probabilities to simplify the training process. However, these unnormalized n -gram probabilities often lead to incorrect word arc posterior probability estimation in the decoding lattice. Since word arc posterior probability estimation is a key to successful KWS systems and effective ASR systems with discriminative decoding (e.g., MBR decoding), the current DLM approaches could cause problems to these systems. Revising the discriminative language modeling approaches to support correct posterior probability estimation in decoding lattices is therefore an important research for applying DLM to KWS tasks.

APPENDIX A

This appendix briefly explains the reason why the probability $P_{\Lambda^*, \Gamma^*}(W|O)$ derived from maximum likelihood (ML) trained HMM-based AMs, which are used for the evaluation of $P_{\Lambda^*}(O|W)$, and n -gram LM, which are used for the evaluation of $P_{\Gamma^*}(W)$, in ASR/KWS systems do not highly correlate with the true $P(W|O)$ in test conditions.

In ML training, it is assumed that the best parametric distributions for data generation (i.e., the generation of (O, W) pairs) are also the best parametric distributions for classification (i.e., the prediction of W based on a given O). However, this assumption is true only when the parametric distributions, $p_{\Lambda}(O|W)$ and $P_{\Gamma}(W)$, we chosen can represent the true probability distributions, $p(O|W)$ and $P(W)$, of the data. Unfortunately, HMM is an imperfect model family for speech data¹⁹ due to the first-order Markov chain and the output-independent assumptions. The same problem holds for n -gram LMs, which make the $n-1$ order Markov assumption to the word sequences. This "incorrect model problem" makes the ML trained HMM-based AMs and n -gram LMs not necessarily well correlate with the true $P(W|O)$, and thus the loss function, in the test conditions.

On the contrary, discriminative training relaxes the assumption in ML training by untying the parametric distributions for classification and data generation in the training process. Therefore, it usually provides a better recognition performance even when incorrect models are used. The technical report "Discriminative models, not discriminative training" written by Tom Minka [128] provides an insight of this perspective about discriminative training. Readers are encouraged to read the report for more details.

¹⁹ If you generate speech signals from an HMM AM, you will find it is almost impossible to generate a continuous spectrogram, which is essential for natural speech signals. A major reason for this is the two "incorrect" assumptions – the first-order Markov chain and the output-independent assumptions – made by the HMM for speech signals.

APPENDIX B

The map between TIMIT phone labels (the full set of 61 phones) and phonological feature values for the GP systems used in this dissertation. The table is adopted from the publication of S. King & P. Taylor in 2000 [105].

phone	A	I	U	E	S	h	H	N	a	i	u
aa	+	-	-	-	-	-	-	-	-	-	-
ae	+	-	-	-	-	-	-	-	-	-	-
ah	-	-	+	+	-	-	-	-	-	-	-
ao	+	-	+	+	-	-	-	-	-	-	+
aw	+	-	+	-	-	-	-	-	+	-	+
ax	-	-	+	-	-	-	-	-	-	-	-
ax-h	-	-	+	+	-	+	-	-	-	-	-
axr	+	-	+	-	-	-	-	-	-	-	-
ay	+	+	-	-	-	-	-	-	+	+	-
b	-	-	+	-	+	+	-	-	-	-	-
bcl	-	-	+	-	+	-	-	-	-	-	-
ch	-	+	-	-	+	-	+	-	-	-	-
d	+	-	-	-	+	+	-	-	-	-	-
dcl	+	-	-	-	+	-	-	-	-	-	-
dh	+	-	-	-	-	+	-	-	-	-	-
dx	+	-	-	+	+	-	-	-	-	-	-
eh	+	+	-	-	-	-	-	-	-	-	-
el	+	-	-	-	+	-	-	-	-	-	-
em	-	-	+	-	+	-	-	+	-	-	-
en	+	-	-	-	+	-	-	+	-	-	-
eng	-	-	-	+	+	-	-	+	-	-	-
epi	-	-	-	-	-	-	-	-	-	-	-
er	+	-	-	+	-	-	-	-	-	-	-
ey	+	+	-	-	-	-	-	-	-	+	-
f	-	-	+	-	-	+	+	-	-	-	-
g	-	-	-	-	+	+	-	-	-	-	-
gcl	-	-	-	-	+	-	-	-	-	-	-
hh	-	-	-	-	-	+	+	-	-	-	-

phone	A	I	U	E	S	h	H	N	a	i	u
hv	-	-	-	-	-	+	-	-	-	-	-
ih	-	+	-	-	-	-	-	-	-	-	-
ix	-	+	-	+	-	-	-	-	-	-	-
iy	-	+	-	-	-	-	-	-	-	+	-
jh	-	+	-	-	+	-	-	-	-	-	-
k	-	-	-	+	+	+	+	-	-	-	-
kcl	-	-	-	+	+	-	+	-	-	-	-
l	+	-	-	-	+	-	-	-	-	-	-
m	-	-	+	-	+	-	-	+	-	-	-
n	+	-	-	-	+	-	-	+	-	-	-
ng	-	-	-	+	+	-	-	+	-	-	-
nx	+	-	-	-	+	-	-	+	-	-	-
ow	+	-	+	-	-	-	-	-	-	-	+
oy	+	+	+	-	-	-	-	-	-	+	+
p	-	-	+	-	+	+	+	-	-	-	-
pau	-	-	-	-	-	-	-	-	-	-	-
pcl	-	-	+	-	+	-	+	-	-	-	-
q	-	-	-	-	+	-	-	-	-	-	-
r	+	-	+	+	-	-	-	-	-	-	-
s	-	-	-	+	-	+	+	-	-	-	-
sh	-	+	-	-	-	+	+	-	-	-	-
t	+	-	-	-	+	+	+	-	-	-	-
tcl	+	-	-	-	+	-	+	-	-	-	-
th	+	-	-	-	-	+	+	-	-	-	-
uh	-	-	+	+	-	-	-	-	-	-	-
uw	-	-	+	-	-	-	-	-	-	-	-
ux	-	-	+	-	-	-	-	-	-	-	-
v	-	-	+	-	-	+	-	-	-	-	-
w	-	-	+	-	-	-	-	-	-	-	-
y	-	+	-	-	-	-	-	-	-	-	-
z	-	-	-	+	-	+	-	-	-	-	-
zh	-	+	-	-	-	+	-	-	-	-	-
sil	-	-	-	-	-	-	-	-	-	-	-

REFERENCES

- [1] C.-H. Lee and Q. Huo, "On Adaptive Decision Rules and Decision Parameter Adaptation for Automatic Speech Recognition," *Proc. IEEE*, vol. 88, pp. 1241-1269, 2000.
- [2] C.-H. Lee, L. R. Rabiner, R. Pieraccini, and J. G. Wilpon, "Acoustic Modeling for Large Vocabulary Speech Recognition," *Computer, Speech and Language*, vol. 4, pp. 127-165, April 1990.
- [3] B.-H. Juang, S. E. Levinson, and M. M. Sondhi, "Maximum Likelihood Estimation for Multivariate Mixture Observations of Markov Chains," *IEEE Trans. Information Theory*, vol. IT-32, pp. 307-309, 1986.
- [4] G. Hinton, L. Deng, D. Yu, G. Dahl, A.-r. Mohamed, N. Jaitly, *et al.*, "Deep Neural Networks for Acoustic Modeling in Speech Recognition," *IEEE Signal Processing Magazine*, 2012.
- [5] S. F. Chen and J. Goodman, "An Empirical Study of Smoothing Techniques for Language Modeling," *Computer Speech & Language*, vol. 13, pp. 359-393, October 1999 1999.
- [6] B.-H. Juang, W. Chou, and C.-H. Lee, "Minimum Classification Error Rate Methods for Speech Recognition," *IEEE Trans. on Acoustic, Speech and Signal Proc.*, vol. 5, pp. 257-265, May 1997.
- [7] L. R. Bahl, P. F. Brown, P. V. d. Souza, and R. L. Mercer, "Maximum Mutual Information Estimation of Hidden Markov Model Parameters for Speech Recognition," in *Proc. ICASSP*, Tokyo, 1986.
- [8] D. Povey, "Discriminative Training for Large Vocabulary Speech Recognition," Ph.D dissertation, University of Cambridge, Cambridge, UK, 2003.
- [9] D. Povey, D. Kanevsky, B. Kingsbury, B. Ramabhadran, G. Saon, and K. Visweswariah, "Boosted MMI for Model and Feature-Space Discriminative Training," in *Proc. ICASSP*, 2008.
- [10] D. Povey and P. C. Woodland, "Minimum Phone Error and I-smoothing for Improved Discriminative Training," in *Proc. ICASSP*, 2002, pp. 105-108.
- [11] B. Kingsbury, "Lattice-based Optimization of Sequence Classification Criteria for Neural-Network Acoustic Modeling," in *Proc. ICASSP*, 2009.

- [12] Z. Chen, K. Lee, and M.-j. Li, "Discriminative Training on Language Model," in *Proc. ICSLP*, Beijing, 2000.
- [13] H.-K. J. Kuo, E. Fosler-Lussier, H. Jiang, and C.-H. Lee, "Discriminative Training of Language Models for Speech Recognition," in *Proc. ICASSP*, 2002.
- [14] B. Roark, M. Saraclar, and M. Collins, "Discriminative n-gram Language Modeling," *Computer Speech and Language*, vol. 21, pp. 373-392, 2007.
- [15] P. Xu, D. Karakos, and S. Khudanpur, "Self-Supervised Discriminative Training of Statistical Language Models," in *Proc. ASRU*, 2009, pp. 317-322.
- [16] J.-T. Huang, X. Li, and A. Acero, "Discriminative Training Methods for Language Models using Conditional Entropy Criteria," in *Proc. ICASSP*, 2010, pp. 5182-5185.
- [17] S. J. Young, J. J. Odell, and P. C. Woodland, "Tree-based State Tying for High Accuracy Acoustic Modelling," in *Proc. Human Language Technology*, 1994, p. 307.
- [18] K. Vesely, M. Hannemann, and L. Burget, "Semi-Supervised Training of Deep Neural Networks," in *Proc. ASRU*, 2013, pp. 267-272.
- [19] J.-T. Huang, J. Li, D. Yu, L. Deng, and Y. Gong, "Cross-Language Knowledge Transfer using Multilingual Deep Neural Network with Shared Hidden Layers," in *Proc. ICASSP*, Vancouver, BC, 2013, pp. 7304 - 7308.
- [20] I. Bulyko, M. Ostendorf, M. Siu, T. Ng, A. Stolcke, and Ö. Çetin, "Web Resources for Language Modeling in Conversational Speech Recognition," *ACM Transactions on Speech and Language Processing*, vol. 5, pp. 1-25, 2007.
- [21] A. Gandhe, L. Qin, F. Metze, A. Rudnicky, I. Lane, and M. Eck, "Using Web Text to Improve Keyword Spotting in Speech," in *Proc. ASRU*, 2013, pp. 428-433.
- [22] D. Vergyri, I. Shafran, A. Stolcke, R. R. Gadde, M. Akbacak, B. Roark, *et al.*, "The SRI/OGI 2006 Spoken Term Detection System," in *Proc. Interspeech*, 2007, pp. 2393-2396.
- [23] J. Mamou, B. Ramabhadran, and O. Siohan, "Vocabulary Independent Spoken Term Detection," in *Proc. SIGIR*, 2007, pp. 615-622.
- [24] D. R. Miller, M. Kleber, C.-I. Kao, O. Kimball, T. Colthurst, S. A. Lowe, *et al.*, "Rapid and Accurate Spoken Term Detection," in *Proc. Interspeech*, 2007.
- [25] R. Wallace, R. Vogt, and S. Sridharan, "A Phonetic Search Approach to the 2006 NIST Spoken Term Detection Evaluation," in *Proc. Interspeech*, 2007.

- [26] J. Makhoul, F. Kubala, T. Leek, D. Liu, L. Nguyen, R. Schwartz, *et al.*, "Speech and Language Technologies for Audio Indexing and Retrieval," *Proc. IEEE*, vol. 88, pp. 1338-1353, 2000.
- [27] R. L. Warren., "Broadcast Speech Recognition System for Keyword Monitoring," U.S. Patent 6332120 B1, 2001.
- [28] T. Kawahara, C.-H. Lee, and B.-H. Juang, "Key-Phrase Detection and Verification for Flexible Speech Understanding," *IEEE Trans. on Speech and Audio Proc.*, vol. 6, pp. 558-568, Nov. 1998.
- [29] B.-H. Juang and S. Furui, "Automatic Recognition and Understanding of Spoken Language – A First Step Toward Natural Human-Machine Communication," *Proc. IEEE*, vol. 88, pp. 1142-1165, Aug. 2000.
- [30] R. DeMori, Ed., *Spoken Dialogues with Computer*. Academic Press, 1998, p.^pp. Pages.
- [31] R. Rosenfeld, "Two Decades of Statistical Language Modeling: Where Do We Go from Here?," *Proc. IEEE*, vol. 88, pp. 1270-1278, 2000.
- [32] H. Schwenk and J.-I. Gauvain, "Neural Network Language Models for Conversational Speech Recognition," in *Interspeech*, ed, 2004, pp. 3-6.
- [33] R. C. Rose and D. B. Paul, "A Hidden Markov Model based Keyword Recognition System," in *Proc. ICASSP*, Albuquerque, NM, 1990, pp. 129-132.
- [34] R. A. Sukkar and C.-H. Lee, "Vocabulary Independent Discriminative Utterance Verification for Non-Keyword Rejection in Subword Based Speech Recognition," *IEEE Trans. on Speech and Audio Proc.*, vol. 4, pp. 420-429, Nov. 1996.
- [35] J. G. Wilpon, L. R. Rabiner, C.-H. Lee, and E. Goldman, "Automatic Recognition of Keywords in Unconstrained Speech using Hidden Markov Models," *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. 38, pp. 1870-1878, 1990.
- [36] M. G. Rahim, C.-H. Lee, and B.-H. Juang, "Discriminative Utterance Verification for Connected Digits Recognition," *IEEE Trans. on Speech and Audio Processing*, vol. 5, pp. 266-277, 1997.
- [37] M.-W. Koo, C.-H. Lee, and B.-H. Juang, "Speech Recognition and Utterance Verification Based on a Generalized Confidence Score," *IEEE Trans. on Speech and Audio Proc.*, vol. 9, pp. 821-832, Nov. 2001.
- [38] J. Ou, K. Chen, X. Want, and Z. Lee, "Utterance Verification of Short Keywords Using Hybrid Neural-Network/HMM Approach," in *Proc. ICII*, Beijing, 2001, pp. 671-676.

- [39] R. P. Lippmann and E. Singer, "Hybrid Neural-Network/HMM Approaches to Wordspotting," in *Proc. ICASSP*, 1993.
- [40] I.-F. Chen and C.-H. Lee, "A Hybrid HMM/DNN Approach to Keyword Spotting of Short Words," in *Proc. Interspeech*, Lyon, 2013, pp. 1574-1578.
- [41] H.-y. Lee and L.-s. Lee, "Enhanced Spoken Term Detection Using Support Vector Machines and Weighted Pseudo Examples," *IEEE Trans. on Audio, Speech and Language Proc.*, vol. 21, pp. 1272-1284, 2013.
- [42] K. Riedhammer, V. H. Do, and J. Hieronymus, "A Study on LVCSR and Keyword Search for Tagalog," in *Proc. Interspeech*, 2013, pp. 2529-2533.
- [43] I. Szoke, P. Schwarz, P. Matejka, L. Burget, M. Karafiat, M. Fapso, *et al.*, "Comparison of Keyword Spotting Approaches for Informal Continuous Speech," in *Proc. EuroSpeech*, 2005.
- [44] P. Jeanrenaud, E. Eide, U. Chaudhari, J. McDonough, K. Ng, M. Siu, *et al.*, "Reducing Word Error Rate on Conversational Speech from the Switchboard Corpus," in *Proc. ICASSP*, 1995, pp. 53-56.
- [45] D. S. Pallett, "A Look at NIST's Benchmark ASR Tests: Past, Present, and Future," in *Proc. ASRU*, 2003, pp. 483-488.
- [46] J. G. Fiscus, J. Ajot, J. S. Garofolo, and G. Doddington, "Results of the 2006 Spoken Term Detection Evaluation," in *Proc. SIGIR*, 2007.
- [47] I. Szoeke, M. Fapso, and L. Burget, "Hybrid Word-Subword Decoding for Spoken Term Detection," in *Proc. SIGIR*, Singapore, 2008, pp. 42-48.
- [48] F. Metze, Z. A. W. Sheikh, A. Waibel, J. Gehring, K. Kilgour, Q. B. Nguyen, *et al.*, "Models of Tone for Tonal and Non-Tonal Languages," in *Proc. ASRU*, 2013, pp. 261-266.
- [49] P. Ghahremani, B. Babaali, D. Povey, K. Riedhammer, J. Trmal, and S. Khudanpur, "A Pitch Extraction Algorithm Tuned for Automatic Speech Recognition," in *Proc. ICASSP*, 2014, pp. 2494 - 2498.
- [50] J. Gehring, Y. Miao, F. Metze, and A. Waibel, "Extracting Deep Bottleneck Features using Stacked Auto-Encoders," in *Proc. ICASSP*, 2013, pp. 3377-3381.
- [51] J. Gehring, Q. B. Nguyen, F. Metze, and A. Waibel, "DNN Acoustic Modeling with Modular Multi-Lingual Feature Extraction Networks," in *Proc. ASRU*, 2013, pp. 344-349.
- [52] K. M. Knill, M. J. F. Gales, S. P. Rath, P. C. Woodland, C. Zhang, and S.-X. Zhang, "Investigation of Multilingual Deep Neural Networks for Spoken Term Detection," in *Proc. ASRU*, 2013, pp. 138-143.

- [53] Z. Tieske, D. Nolden, R. Schlueter, and H. Ney, "Multilingual MRASTA Features for Low-Resource Keyword Search and Speech Recognition Systems," in *Proc. ICASSP*, 2014, pp. 7854 - 7858.
- [54] J. Richards, M. Ma, and A. Rosenberg, "Using Word Burst Analysis to Rescore Keyword Search Candidates on Low-Resource Languages," in *Proc. ICASSP*, 2014, pp. 7874-7878.
- [55] V. Soto, E. Cooper, L. Mangu, A. Rosenberg, and J. Hirschberg, "Rescoring Confusion Networks for Keyword Search," in *Proc. ICASSP*, 2014, pp. 7138-7142.
- [56] V. Soto, L. Mangu, A. Rosenberg, and J. Hirschberg, "A Comparison of Multiple Methods for Rescoring Keyword Search Lists for Low Resource Languages," in *Proc. Interspeech*, Singapore, 2014, pp. 2464-2468.
- [57] M. Ma, J. Richards, V. Soto, J. Hirschberg, and A. Rosenberg, "Strategies for Rescoring Keyword Search Results Using Word-Burst and Acoustic Features," in *Proc. Interspeech*, Singapore, 2014, pp. 2769-2773.
- [58] K. Audhkhasi, A. Sethy, B. Ramabhadran, and S. S. Narayanan, "Semi-Supervised Term-Weighted Value Rescoring for Keyword Search," in *Proc. ICASSP*, 2014, pp. 7919-7923.
- [59] O. Vinyals and S. Wegmann, "Chasing the Metric: Smoothing Learning Algorithms for Keyword Detection," in *Proc. ICASSP*, 2014, pp. 3325-3329.
- [60] V. T. Pham, H. Xu, N. F. Chen, S. Sivadas, B. P. Lim, E. S. Chng, *et al.*, "Discriminative Score Normalization for Keyword Search Decision," in *Proc. ICASSP*, 2014, pp. 7128-7132.
- [61] J. Chiu and A. Rudnicky, "Using Conversational Word Bursts in Spoken Term Detection," in *Proc. Interspeech*, 2013, pp. 2247-2251.
- [62] H.-y. Lee, Y. Zhang, E. Chuangsuwanich, and J. Glass, "Graph-based Re-Ranking using Acoustic Feature Similarity between Search Results for Spoken Term Detection on Low-Resource Languages," in *Proc. Interspeech*, Singapore, 2014, pp. 2479-2483.
- [63] D. Karakos, R. Schwartz, S. Tsakalidis, L. Zhang, S. Ranjan, T. Ng, *et al.*, "Score Normalization and System Combination for Improved Keyword Spotting," in *Proc. ASRU*, 2013, pp. 210-215.
- [64] J. Mamou, J. Cui, X. Cui, M. J. F. Gales, B. Kingsbury, K. Knill, *et al.*, "System Combination and Score Normalization for Spoken Term Detection," in *Proc. ICASSP*, 2013, pp. 8272-8276.

- [65] S. P. Rath, K. M. Knill, A. Ragni, and M. J. F. Gales, "Combining Tandem and Hybrid Systems for Improved Speech Recognition and Keyword Spotting on Low Resource Languages," in *Proc. Interspeech*, Singapore, 2014, pp. 835-839.
- [66] Y. He, B. Hutchinson, P. Baumann, M. Ostendorf, E. Fosler-lussier, and J. Pierrehumbert, "Subword-based Modeling for Handling OOV Words in Keyword Spotting," in *Proc. ICASSP*, 2014, pp. 7914-7918.
- [67] D. Karakos and R. Schwartz, "Subword and Phonetic Search for Detecting Out-of-Vocabulary Keywords," in *Proc. Interspeech*, Singapore, 2014, pp. 2469-2473.
- [68] V.-b. Le, L. Lamel, A. Messaoudi, W. Hartmann, J.-l. Gauvain, C. Woehrling, *et al.*, "Developing STT and KWS Systems using Limited Language Resources," in *Proc. Interspeech*, Singapore, 2014, pp. 2484-2488.
- [69] J. Chiu, Y. Wang, J. Trmal, D. Povey, G. Chen, and A. Rudnicky, "Combination of FST and CN Search in Spoken Term Detection Center for Language and Speech Processing & Human Language Technology Center of," in *Proc. Interspeech*, Singapore, 2014, pp. 2784-2788.
- [70] C. Weng and B.-H. Juang, "Adaptive Boosted Non-Uniform MCE for Keyword Spotting on Spontaneous Speech," in *Proc. ICASSP*, 2013.
- [71] I.-F. Chen, N. F. Chen, and C.-H. Lee, "A Keyword-Boosted sMBR Criterion to Enhance Keyword Search Performance in Deep Neural Network Based Acoustic Modeling," in *Proc. Interspeech*, Singapore, 2014, pp. 2779-2783.
- [72] X. Zhang, J. Trmal, D. Povey, and S. Khudanpur, "Improving Deep Neural Network Acoustic Models using Generalized Maxout Networks," in *Proc. ICASSP*, 2014, pp. 215-219.
- [73] C. Liu, A. Jansen, G. Chen, K. Kintzley, J. Trmal, and S. Khudanpur, "Low-Resource Open Vocabulary Keyword Search Using Point Process Models," in *Proc. Interspeech*, Singapore, 2014, pp. 2789-2793.
- [74] X. Cui, B. Kingsbury, J. Cui, B. Ramabhadran, A. Rosenberg, M. S. Rasooli, *et al.*, "Improving Deep Neural Network Acoustic Modeling For Audio Corpus Indexing under The IARPA Babel Program," in *Proc. Interspeech*, Singapore, 2014, pp. 2103-2107.
- [75] A. Ragni, K. M. Knill, S. P. Rath, and M. J. F. Gales, "Data Augmentation for Low Resource Languages," in *Proc. Interspeech*, Singapore, 2014, pp. 810-814.
- [76] Z. Tieske, P. Golik, D. Nolden, R. Schlueter, and H. Ney, "Data Augmentation, Feature Combination, and Multilingual Neural Networks to Improve ASR and KWS Performance for Low-Resource Languages," in *Proc. Interspeech*, Singapore, 2014, pp. 1420-1424.

- [77] R. Hsiao, T. Ng, L. Zhang, S. Ranjan, S. Tsakalidis, L. Nguyen, *et al.*, "Improving Semi-supervised Deep Neural Network for Keyword Search in Low Resource Languages," in *Proc. Interspeech*, Singapore, 2014, pp. 1088-1091.
- [78] R. Hsiao, T. Ng, F. Grezl, D. Karakos, S. Tsakalidis, L. Nguyen, *et al.*, "Discriminative Semi-Supervised Training for Keyword Search in Low Resource Languages," in *Proc. ASRU*, 2013, pp. 440-445.
- [79] H. Xu, H. Su, E.-s. Chng, and H. Li, "Semi-Supervised Training for Bottle-Neck Feature based DNN-HMM Hybrid Systems," in *Proc. Interspeech*, Singapore, 2014, pp. 2078-2082.
- [80] M. Singh and D. Klakow, "Comparing RNNs and Log-Linear Interpolation of Improved Skip-Model on Four Babel Languages: Cantonese, Pashto, Tagalog, Turkish," in *Proc. ICASSP*, 2013, pp. 8416-8420.
- [81] A. Gandhe, F. Metze, and I. Lane, "Neural Network Language Models for Low Resource Languages," in *Proc. Interspeech*, Singapore, 2014, pp. 2615-2619.
- [82] A. Gandhe, F. Metze, A. Waibel, and I. Lane, "Optimization of Neural Network Language Models for keyword search," in *Proc. ICASSP*, 2014, pp. 4888-4892.
- [83] J. R. Novak, N. Minematsu, and K. Hirose, "WFST-based Grapheme-to-Phoneme Conversion: Open Source Tools for Alignment, Model-Building and Decoding," in *Proc. International Workshop on Finite State Methods and Natural Language Processing*, Donostia-San Sebastian, 2012, pp. 45-49.
- [84] *Sequitur G2P*. Available: <http://www-i6.informatik.rwth-aachen.de/web/Software/g2p.html>
- [85] G. Chen, O. Yilmaz, J. Trmal, D. Povey, and S. Khudanpur, "Using Proxies for OOV Keywords in the Keyword Search Task," in *Proc. ASRU*, 2013, pp. 416-421.
- [86] M. Saraclar, A. Sethy, B. Ramabhadran, L. Mangu, J. Cui, X. Cui, *et al.*, "An Empirical Study of Confusion Modeling in Keyword Search for Low Resource Languages," in *Proc. ASRU*, 2013, pp. 464-469.
- [87] L. Mangu, B. Kingsbury, H. Soltau, H.-K. Kuo, and M. Picheny, "Efficient Spoken Term Detection using Confusion Networks," in *Proc. ICASSP*, 2014, pp. 7894-7898.
- [88] D. Xu and F. Metze, "Word-based Probabilistic Phonetic Retrieval for Low-Resource Spoken Term Detection," in *Proc. Interspeech*, Singapore, 2014, pp. 2774-2778.

- [89] W. Hartmann, V.-b. Le, A. Messaoudi, L. Lamel, and J.-l. Gauvain, "Comparing Decoding Strategies for Subword-based Keyword Spotting in Low-Resourced Languages," in *Proc. Interspeech*, Singapore, 2014, pp. 2764-2768.
- [90] *TIMIT Acoustic-Phonetic Continuous Speech Corpus*. Available: <https://catalog.ldc.upenn.edu/LDC93S1>
- [91] *BABEL Program*. Available: <http://www.iarpa.gov/Programs/ia/Babel/babel.html>
- [92] *NIST Open Keyword Search 2013 Evaluation (OpenKWS13)*. Available: <http://www.nist.gov/itl/iad/mig/openkws13.cfm>
- [93] *NIST Open Keyword Search 2014 Evaluation (OpenKWS14)*. Available: <http://www.nist.gov/itl/iad/mig/openkws14.cfm>
- [94] *NIST Open Keyword Search 2015 Evaluation (OpenKWS15)*. Available: <http://www.nist.gov/itl/iad/mig/openkws15.cfm>
- [95] NIST, "The Road Rally Word-spotting Corpora (RDRALLY1)," ed. NIST Speech disc 6-1, 1991.
- [96] F. Metze, N. Rajput, X. Anguera, M. Davel, G. Gravier, C. v. Heerden, *et al.*, "The Spoken Web Search Task at MediaEval 2011," in *Proc. ICASSP*, 2012, pp. 5165 - 5168.
- [97] F. Metze, X. Anguera, E. Barnard, M. Davel, and G. Gravier, "The Spoken Web Search Task at MediaEval 2012," in *Proc. ICASSP*, 2013, pp. 8121 - 8125.
- [98] J.-L. Gauvain and L. Lamel, "Large Vocabulary Continuous Speech Recognition: Advances and Applications," *Proc. IEEE*, vol. 88, pp. 1181-1200, August 2000.
- [99] J.-L. Gauvain and C.-H. Lee, "Maximum A Posteriori Estimation for Multivariate Gaussian Mixture Observations of Markov Chains," *IEEE Trans. on Speech and Audio Proc.*, vol. 2, pp. 291-298, April 1994.
- [100] I.-F. Chen and C.-H. Lee, "A Study on Using Word-Level HMMs to Improve ASR Performance over State-of-the-Art Phone-Level Acoustic Modeling for LVCSR," in *Proc. Interspeech*, Portland, U.S.A, 2012.
- [101] I.-F. Chen and C.-H. Lee, "A Resource-Dependent Approach to Word Modeling for Keyword Spotting," in *Proc. Interspeech*, Lyon, 2013, pp. 2544-2548.
- [102] B. Sigurd, M. Eeg-Olofsson, and J. v. d. Weijer, "Word Length, Sentence Length and Frequency -- ZIPF Revisited," *Studia Linguistica*, vol. 58, pp. 37-52, 2004.
- [103] K.-Y. Su and C.-H. Lee, "Speech Recognition using Weighted HMM and Subspace Projection Approaches," *IEEE Transactions on Speech and Audio Processing*, vol. 2, pp. 69-79, Jan. 1994.

- [104] J. Harris, *English Sound Structure*: Oxford: Blackwell, 1994.
- [105] S. King and P. Taylor, "Detection of Phonological Features in Continuous Speech using Neural Networks," *Computer Speech and Language*, vol. 14, pp. 333-354, 2000.
- [106] G. E. Hinton, S. Osindero, and Y.-W. Teh, "A Fast Learning Algorithm for Deep Belief Nets," *Neural Computation*, vol. 18, pp. 1527-1554, 2006.
- [107] A.-R. Mohamed, G. E. Dahl, and G. Hinton, "Acoustic Modeling using Deep Belief Networks," *IEEE Trans. Acoustic, Speech and Signal Proc.*, vol. 20, pp. 14-22, 2012.
- [108] C. Ma and C.-H. Lee, "A Study on Word Detector Design and Knowledge-based Pruning and Rescoring," in *Proc. Interspeech*, Antwerp, Belgium, 2007, pp. 1473-1476.
- [109] I.-F. Chen and H.-m. Wang, "Articulatory Feature Asynchrony Analysis and Compensation in Detection-Based ASR," in *Proc. Interspeech*, Brighton, UK, 2009.
- [110] W. M. Fisher, G. R. Doddington, and K. M. Goudie-Marshall, "The DARPA Speech Recognition Research Data Base: Specifications and Status," in *Proc. DARPA Workshop on Speech Recognition*, 1986, pp. 93-99.
- [111] M. Weintraub, "LVCSR Log-Likelihood Ratio Scoring for Keyword Spotting," in *Proc. ICASSP*, 1995.
- [112] M. Mohri, F. Pereira, and M. Riley, "Speech Recognition with Weighted Finite-State Transducers," in *Springer Handbook of Speech Processing*, ed: Springer Berlin Heidelberg, 2008, pp. 559-584.
- [113] C. Allauzen, M. Mohri, and B. Roark, "Generalized Algorithms for Constructing Language Models," in *Proc. ACL*, Stroudsburg, PA, USA, 2003, pp. 40-47.
- [114] W. Kuch and A. Salomaa, Eds., *Semirings, Automata, Languages*. London, UK: Springer-Verlag, 1986, p.^pp. Pages.
- [115] J. Cui, X. Cui, B. Ramabhadran, J. Kim, B. Kingsbury, J. Mamou, *et al.*, "Developing Speech Recognition Systems for Corpus Indexing under the IARPA Babel Program," in *Proc. ICASSP*, 2013, pp. 6753-6757.
- [116] D. Povey, A. Ghoshal, G. Boulianne, L. s. B. ˇ, O. r. Glembek, N. Goel, *et al.*, "The Kaldi Speech Recognition Toolkit," in *Proc. IEEE 2011 Workshop on Automatic Speech Recognition and Understanding*, Hilton Waikoloa Village, Big Island, Hawaii, US, 2011.

- [117] K. Vesely, A. Ghoshal, L. Burget, and D. Povey, "Sequence-Discriminative Training of Deep Neural Networks," in *Proc. Interspeech*, Lyon, France, 2013, pp. 2345-2349.
- [118] S. M. Katz, "Estimation of Probabilities from Sparse Data for the Language Model Component of a Speech Recognizer," *IEEE Trans. on Acoustic, Speech and Signal Proc.*, vol. 35, pp. 400-401, 1987.
- [119] M. Gibson and T. Hain, "Hypothesis Spaces for Minimum Bayes Risk Training in Large Vocabulary Speech Recognition," in *Proc. Interspeech*, 2006.
- [120] M. Collins, "Discriminative Training Methods for Hidden Markov Models: Theory and Experiments with Perceptron Algorithms," in *Proc. the Conference on Empirical Methods in NLP (EMNLP)*, 2002.
- [121] N. F. Chen, S. Sivadas, B. P. Lim, H. G. Ngo, H. Xu, V. T. Pham, *et al.*, "Strategies for Vietnamese Keyword Search," in *Proc. ICASSP*, 2014, pp. 4149-4153.
- [122] I.-F. Chen, C. Ni, B. P. Lim, N. F. Chen, and C.-H. Lee, "A Novel Keyword+LVCSR-Filler Based Grammar Network Representation for Spoken Keyword Search," in *Proc. ISCSLP*, Singapore, 2014, pp. 192-196.
- [123] S. Wegmann, A. Faria, A. Janin, K. Riedhammer, and N. Morgan, "The Tao of ATWV: Probing the Mysteries of Keyword Search Performance," in *Proc. ASRU*, 2013, pp. 192-197.
- [124] L. J. Rodriguez-Fuentes and M. Penagarikano, "MediaEval 2013 Spoken Web Search Task : System Performance Measures," Department of Electricity and Electronics, University of the Basque Country 2013.
- [125] V. Goel and W. J. Byrne, "Minimum Bayes-Risk Automatic Speech Recognition," vol. 14, pp. 115-135, 2000.
- [126] R. Lyer, M. Ostendorf, and H. Gish, "Using Out-of-Domain Data to Improve In-Domain Language Models," *IEEE Signal Processing Letters*, vol. 4, pp. 221-223, 1997.
- [127] X. Zhu and R. Rosenfeld, "Improving Trigram Language Modeling with the World Wide Web," in *ICASSP*, ed, 2001, pp. 533-536.
- [128] T. Minka, "Discriminative Models, Not Discriminative Training," Microsoft Research Technical Report MSR-TR-2005-144, 2005.