Robust spoken term detection using partial search and re-scoring hypothesized detections techniques

Pham, Van Tung

2019

Pham, V. T. (2019). Robust spoken term detection using partial search and re-scoring hypothesized detections techniques. Doctoral thesis, Nanyang Technological University, Singapore.

<https://hdl.handle.net/10356/82987>

<https://doi.org/10.32657/10220/47558>

ROBUST SPOKEN TERM DETECTION USING
PARTIAL SEARCH AND RE-SCORING
HYPOTHESIZED DETECTIONS TECHNIQUES

PHAM VAN TUNG

SCHOOL OF COMPUTER SCIENCE AND ENGINEERING

2019

ROBUST SPOKEN TERM DETECTION USING
PARTIAL SEARCH AND RE-SCORING
HYPOTHESIZED DETECTIONS TECHNIQUES

PHAM VAN TUNG

School of Computer Science and Engineering

A thesis submitted to the Nanyang Technological University
in partial fulfilment of the requirement for the degree of
Doctor of Philosophy

2019

Abstract

This research focusses on Spoken Term Detection (STD) which aims to detect a textual keyword in a speech corpus. A typical STD system relies on an Automatic Speech Recognition (ASR) system to transform the speech corpus to intermediate textual representations such as 1-best transcriptions or lattices of word and subword for indexing/retrieval. However, the imperfect modelling of ASR results in two types of error, i.e. missing and false alarm. This thesis aims to address both type of errors.

In STD, the subword approach has been attractive because it is able to address the Out-of-Vocabulary (OOV) problems. A standard subword-based STD system, referred to as the full search technique, first converts the keyword into a subword sequence, then searches the subword sequence in the subword lattices. However, due to the high error rate of subword ASR, detecting entire subword sequence in lattices is difficult and results in a high miss rate. This thesis proposes a partial search approach to address this problem. The proposed approach transforms the keyword’s subword sequence into overlapping sub-sequences and then searches these sub-sequences in the index. It reduces the miss rate by accepting hypothesized detections that contain some of the keyword’s sub-sequences.

STD systems rank and make “accept/reject” decisions on hypothesized detections by the confidence scores estimated from the decoding lattices generated by the ASR. Such scores may be inaccurate due to the imperfect modelling of speech and noise. Using the lattice-based posterior probabilities as the detection scores might result in degraded STD performance. Firstly, it is observed that the posterior probabilities are not comparable across keywords. As a result, it is difficult to make “accept/reject” decisions using a single threshold for all keywords. Secondly, a correct detection might have a smaller posterior probability than false alarm detections. Two techniques to re-score and re-rank

hypothesized detections are proposed. These techniques utilize additional information that is not captured by the detection scores, hence improve the STD performance.

The first technique re-scores hypothesized detections using keyword exemplars. A keyword exemplar is a true instance of the keyword obtained from a labelled speech corpus. The main idea is that if a hypothesized detection is acoustically more similar to the keyword exemplars, it is more likely to be a true detection and hence its score should be boosted. Experimental results show that the proposed technique consistently outperforms previous re-ranking methods that do not make use of keyword exemplars.

The second technique re-scores hypothesized detections by exploiting features derived from competing hypotheses. Competing hypotheses of a detection are its alternative hypotheses which have similar time information as the detection in the corresponding lattice. From the competing hypotheses, several novel features are derived. These features reflect the relative confidence of the hypothesized detection to its competing hypotheses. These features are informative and can be used to re-score detections. Experimental results show that using these features result in improved STD performance.

Statement of Originality

I hereby certify that the work embodied in this thesis is the result of original research, is free of plagiarised materials, and has not been submitted for a higher degree to any other University or Institution.

16 January 2019

.....

Date

.....
Pham Van Tung

Supervisor Declaration Statement

I have reviewed the content and presentation style of this thesis and declare it is free of plagiarism and of sufficient grammatical clarity to be examined. To the best of my knowledge, the research and writing are those of the candidate except as acknowledged in the Author Attribution Statement. I confirm that the investigations were conducted in accord with the ethics policies and integrity standards of Nanyang Technological University and that the research data are presented honestly and without prejudice.

16 January 2019

.....
Date

.....
Assoc. Prof. Chng Eng Siong

Authorship Attribution Statement

This thesis contains material from 5 papers published in the following peer-reviewed journal and conferences where I was the first author.

Chapter 3 is published as V. T. Pham, H. Xu, X. Xiao, N. F. Chen, E. S. Chng, “Pruning strategies for partial search in Spoken Term Detection”, In Proc. of SoICT 2017, pp. 114-119.

The contributions of the co-authors are as follows:

- I and Dr. Xu initialized the idea of the proposed partial search framework.
- I co-designed the study with Dr. Xiao. Then, I performed all experimental works.
- Dr. Chen and Prof. Chng assisted the experimental analyses.
- I prepared the manuscript draft. Then, it was revised by Dr. Xu and Prof. Chng.

Chapter 4 is published as:

- V. T. Pham, H. Xu, X. Xiao, N. F. Chen, E. S. Chng, H. Li, “Keyword search using query expansion for graph-based rescoring of hypothesized detections”, in Proc. of ICASSP, 2016, pp. 6035-6039.
- V. T. Pham, H. Xu, X. Xiao, N. F. Chen, E. S. Chng, H. Li, “Rescoring hypothesized detections of Out-of-Vocabulary keywords using subword samples”, in Proc. of Interspeech 2016, pp. 933-937.
- V. T. Pham, H. Xu, X. Xiao, N. F. Chen, E. S. Chng, H. Li, “Re-ranking Spoken Term Detection with acoustic exemplars of keywords”, in Speech Communication, Vol 104, November 2018, pp. 12-23.

The contributions of the co-authors are as follows:

- Dr. Chen suggested the idea of using keyword exemplars for re-ranking in STD.
- I designed the keyword exemplar generation method with the assistance from Dr. Xu and Prof. Li.
- I examined the acoustic similarity estimation and re-ranking methods with the assistance from Dr. Xiao.
- Dr. Chen and Prof. Chng assisted me in designing experiments and analyses. Then, I performed all experimental works.
- I prepared the manuscript draft. Then, it was revised by all other authors.

Chapter 5 is published as V.T. Pham, H. Xu, N. F. Chen, S. Sivadas, B. P. Lim, E. S. Chng, H. Li, "Discriminative score normalization for keyword search decision", in Proc. of ICASSP 2014, pp. 7128-7132.

The contributions of the co-authors are as follows:

- I and Dr. Xu initialized the idea of using competing hypotheses for re-scoring.
- I proposed novel features with the suggestions from Dr. Chen.
- Dr. Chen and Dr. Xu suggested case studies for experiments.
- Dr. Sivadas and Dr. Lim helped to build baseline STD system, then I conducted all experimental works related to the proposed method.
- I prepared the manuscript draft. The manuscript was revised together with Prof. Chng and Prof. Li.

16 January 2019

.....
Date

.....
Pham Van Tung

Acknowledgments

I would like to express my sincere thanks and appreciation to my supervisor, Dr. Chng Eng Siong (NTU), and co-supervisor, Dr. Li Haizhou (I²R and NUS) for their invaluable guidance, support and suggestions. Their deep knowledge and skillful supervision help me to overcome the difficulties encountered in my research.

I also want to thank my colleagues in the speech group in NTU and I²R, for their generous help. I want to thank Dr. Xu Haihua, Dr. Chen F. Nancy, Dr. Xiao Xiong for fruitful discussions. I also want to thank to Dr. Do Van Hai, Dr. Nguyen Duc Hoang Ha for many valuable advices and suggestions. I am very glad to collaborate with my team mates, Vu Thi Ly, Vu Tuan Thanh, Chong Tze Yuang, Tian Xiaohai, Wu ZhiZheng, Sunil Sivadas, Lim Boon Pang, Lim Zhi Hao, Xu Chenglin, Rao Wei, Hou Nana.

Last but not least, I would like to thank my wife Nga and my daughter Huyen Trang, for their support, encouragement, unwavering love. I also want to thank my family in Vietnam, for their constant love and support.

List of Abbreviations

ASR	Automatic Speech Recognition
ATWV	Actual Term-Weighted Value
BNF	Bottleneck feature
CN	Confusion Network
DET	Decision Error Trade-Off
DTW	Dynamic Time Warping
DNN	Deep Neural Network
GBR	Graph-based Re-ranking
GRW	Graph Random Walk
GTP	Global Threshold Pruning
HMM	Hidden Markov Model
IV	In-vocabulary
IR	Information Retrieval
KWS	Keyword Search
KLD	Kullback-Leibler Divergence
KSP	Keyword-Specific Pruning
KST	Keyword Specific Threshold
LM	Language Model
LVCSR	Large-vocabulary Continuous Speech Recognition
MAP	Mean Average Precision
MFCC	Mel Filterbank Cepstral Coefficients
MLP	Multi-layer Perceptron
MTWV	Maximum Term-Weighted Value
NIST	National Institute of Standards and Technology
NN	Neural Network
OOV	Out Of Vocabulary
PLP	Perceptual Linear Prediction
PRF	Pseudo Relevance Feedback
PSPL	Position Specific Posterior Lattice
QbE-STD	Query-by-Example Spoken Term Detection
RAE	Re-ranking by Acoustic Exemplar
RAE-SF	RAE by score fusion
RAE-SG	RAE by similarity graph

SDR	Spoken Document Retrieval
STD	Spoken Term Detection
STO	Sum-To-One
STWV	Supremum Term-Weighted Value
TTS	Text-to-Speech
TWV	Term-Weighted Value
WER	Word Error Rate
WFST	Weighted Finite State Transducer

Contents

Abstract	ii
Statement of Originality	iv
Supervisor Declaration Statement	v
Authorship Attribution Statement	vi
Acknowledgments	viii
List of Abbreviations	ix
List of Figures	xvi
List of Tables	xx
1 Introduction	1
1.1 Motivation	1
1.2 Approaches to speech-based information retrieval	1
1.2.1 Spoken Term Detection	1
1.2.2 Keyword Spotting	2
1.2.3 Spoken Document retrieval (SDR)	3
1.2.4 Query-by-Example Spoken Term Detection (QbE-STD)	3
1.3 Challenges and existing approaches for STD	4
1.4 Contributions	7
1.5 Thesis structure	9
2 Fundamentals and Previous Works	10
2.1 Spoken Term Detection	10
2.1.1 STD queries	11
2.1.2 Common architecture of STD systems	11
2.1.3 Automatic Speech Recognition	12

2.1.3.1	Feature extraction	12
2.1.3.2	Acoustic model	13
2.1.3.3	Language model	14
2.1.3.4	Lexicon	14
2.1.3.5	Decoding	15
2.1.3.6	ASR lattices	16
2.1.4	Indexer and search engine	17
2.1.4.1	Indexing N-best ASR transcriptions	18
2.1.4.2	Indexing and retrieving ASR lattices	19
2.1.4.3	Fast retrieval	21
2.1.5	Post-processing	22
2.1.5.1	Re-scoring hypothesized detections	22
2.1.5.2	System combination	23
2.1.6	Evaluation metrics for STD	26
2.2	Approaches to reduce miss rate of STD systems	28
2.2.1	Subword approaches for STD	29
2.2.2	Approximate matching for STD	31
2.2.2.1	Estimating the confusion model	31
2.2.2.2	Approximate matching techniques	33
2.3	Re-scoring Spoken Term Detection	33
2.3.1	Lattice-based posterior probability and its limitations	33
2.3.1.1	Lattice-based posterior probability estimation	33
2.3.1.2	Limitations of the lattice-based posterior probability	36
2.3.2	Techniques to re-score hypothesized detections	37
2.3.2.1	Score normalization	37
2.3.2.2	Score re-estimation	39
2.4	Summary	42

3	Pruning Strategies for Partial Search in Spoken Term Detection	44
3.1	Introduction	44
3.2	The proposed partial search framework	45
3.2.1	Partial sequence generation	47
3.2.2	Partial sequence retrieval	47
3.2.3	Merging detections lists	47
3.2.4	Pruning of merged results	48
3.2.5	Scoring the pruned list	50
3.3	Experiment setup	50
3.3.1	NIST OpenKWS15 dataset	50
3.3.2	Keyword Search systems	51
3.3.3	Keyword lists and evaluation metric	51
3.3.4	Baseline systems	53
3.4	Experiment results and analysis	55
3.4.1	Results of baseline approaches	55
3.4.2	Results of proposed approaches	56
3.5	Summary	59
4	Re-ranking Spoken Term Detection with Acoustic Exemplars of Key-words	61
4.1	Introduction	61
4.2	Re-ranking Spoken Term Detection with acoustic keyword exemplars . .	62
4.3	Keyword exemplar generation	64
4.3.1	Exemplar generation for a seen keyword	64
4.3.2	Exemplar generation for an unseen keyword	64
4.3.2.1	Subword samples extraction	65
4.3.2.2	Subword samples concatenation	65
4.4	Acoustic similarity estimation	68
4.4.1	Dynamic Time Warping	69
4.4.2	Score calibration techniques	69
4.4.2.1	Min-max normalization	69

4.4.2.2	Logistic regression calibration	70
4.4.2.3	Neural network classifier	70
4.5	Re-ranking methods	71
4.5.1	RAE by score fusion	71
4.5.2	RAE by similarity graph	73
4.6	Experiment settings	76
4.6.1	KWS datasets	76
4.6.2	Setup for keyword exemplar generation	76
4.6.3	Setup for acoustic similarity estimation	77
4.7	Experiment results and analysis	77
4.7.1	The effect of different acoustic similarity estimation methods	78
4.7.2	The effect of keyword exemplars for re-ranking hypothesized detections	79
4.7.3	Analysis of RAE methods	81
4.8	Summary	83
5	Re-scoring Hypothesized Detections Using Lattice-based Competing Hypotheses	85
5.1	Introduction	85
5.2	Novel features derived from competing hypotheses set of each detection	86
5.3	Two-stage confidence estimation	88
5.4	Experimental setup	89
5.5	Experimental results	91
5.5.1	Effect Size of Mann-Whitney U-test	91
5.5.2	The efficiency of novel features for re-scoring STD results	92
5.6	Discussion	95
5.7	Summary	97
6	Conclusion and Future Work	99
6.1	Contributions	99
6.1.1	A partial search approach for subword-based STD systems	99
6.1.2	A re-ranking framework that uses keyword exemplars	100

6.1.3	A re-scoring framework that uses lattice-based competing hypotheses	102
6.1.4	Combining proposed approaches to improve STD performance . .	102
6.2	Future work	103
List of Publications		106
References		107

List of Figures

2.1	A typical STD system.	12
2.2	A typical ASR system.	13
2.3	A typical left-to-right HMM.	14
2.4	Examples of the Carnegie Mellon University (CMU) pronunciation dictionary. Each row presents a pronunciation, i.e. sequence of phones, of a word.	15
2.5	The search space of the decoder which is constructed at different hierarchy levels.	16
2.6	Example of an ASR lattice [1].	17
2.7	A raw lattice and the corresponding PSPL and CN structures.	18
2.8	The inverted index structure.	19
2.9	WFST indexing.	21
2.10	ASR output combination [2] using WFST.	25
2.11	The search result combination procedure.	25
2.12	Example of the grapheme subword units.	31
2.13	Example of a confusion model [3].	34
2.14	Illustration of the sets of arcs before and after the arc a , i.e. $PREC(a)$ and $SUC(a)$	35
2.15	The keyword specific threshold as a function of number of keyword's occurrences	38

3.1	Proposed partial search framework. Several partial sequences are derived from the subword sequence of the keyword q . These partial sequences are then searched on subword lattices to produce search results L_1, L_2, L_3 . $A1, A2, A3$ are audio files. Each item in L_i ($i = 1, 2, \dots$) has format (audio, start.time, end.time, score). For example, the item $r_{1,1} \in L_1$ (red color) means that the partial sequence $p_1p_2p_3$ appears in audio file $A1$ at the start time 3.2 and end time 4.0, and it has a confidence score 0.5 which is produced by the WFST indexing/retrieval framework (see Section 2.1.4.2). Items $r_{1,1}, r_{2,2}, r_{3,1}$ form an overlapping detections set (in Section 3.2.3) which are merged into one single detection d_1 in L_{merged}	46
3.2	The process to obtain morpheme-based dictionary and language model. .	52
3.3	Keyword distribution w.r.t keyword length of the <i>IV</i> and <i>OOV</i> keyword lists of Swahili.	53
3.4	The baseline full search approach.	54
3.5	The MTWV of baseline systems on development data while varying the number of expansions per keyword, i.e. the N_{best} value in Equation 3.4. The left and right figures correspond to the grapheme and morpheme systems respectively.	55
3.6	The MTWV on the development data while varying the values of parameters N and T_{global} . The left and right figures correspond to <i>IV</i> and <i>OOV</i> keywords respectively, while the top and bottom figures correspond to grapheme and morpheme systems respectively.	57
3.7	The ATWV of partial search with KSP and full search on the <i>IV</i> and <i>OOV</i> keyword lists with different keyword length. The left figure is for the morpheme system while the right figure is for the grapheme system. .	58
3.8	The average processing time per keyword of the proposed partial search approach as compared to the query expansion and full search approach. The left figure is for the grapheme system while the right figure is for the morpheme system. Experiments are run on a workstation with 256 GB memory, 20 cores CPU.	59
4.1	Proposed re-ranking by acoustic exemplars (RAE) framework.	63

4.2	Acoustic exemplar generation for an unseen keyword $q = nafasi = na fa si$. Rounded blocks of the same column are instances of the same subword, each might have different phonetic context, speaker, gender, channel information. Such information will be incorporated into the target and concatenation cost functions to improve the generated exemplars. Using these cost functions, a dynamic programming algorithm estimates K-best concatenation sequences. The best sequence goes through the shade blocks in this figure.	66
4.3	The conventional graph-based re-ranking (a) and the proposed RAE by similarity graph (b). Hypothesized detections d_1, d_2 , are in black color while keyword exemplars $e_1, e_2, ..$ are in red color. The conventional graph-based re-ranking only uses detections to perform re-ranking while the proposed RAE-SG uses both detections and keyword exemplars for re-ranking.	73
4.4	The DET curves of different acoustic similarity estimation methods on the morpheme system for IV keywords (top) and OOV keywords (bottom). .	79
4.5	The comparison between RAE-SG and the GBR on rare, normal and frequent keywords.	81
4.6	The comparison between RAE-SG and the GBR on short, medium and long keywords for IV keywords (top figure) and OOV keywords (bottom figure).	82
4.7	The MTWV score of the RAE-SG on IV keywords when varying each of three parameters, α, γ and δ , and fixing the others.	83
5.1	A detection hypothesis d and its competing hypotheses $T(d)$	87
5.2	The two-stage process to estimate the new confidence score. The binary classifier produces discriminative confidence scores, while a metric-based score normalization outputs a new set of scores that lead to optimum TWV metric.	89
5.3	DET curves of NN-based re-scoring models trained with and without novel features.	93
5.4	Approximation of indicator function.	96

6.1	A block diagram of the end-to-end re-scoring model.	104
-----	---	-----

List of Tables

3.1	Detailed acoustic data for OpenKWS15 Evaluation.	51
3.2	The optimum values of the length of a partial sequence N and the global threshold T_{global} that are obtained from the development data.	56
3.3	The KWS results of proposed partial search approach as compared to its baselines on <i>evalpart1</i> test set.	58
4.1	Summary of sub-costs for keyword exemplars generation. h_i and h_{i+1} are instances of i^{th} and $(i + 1)^{th}$ subword units respectively	77
4.2	The distribution of the multilingual data for training the bottleneck feature (BNF) extractor.	78
4.3	A comparison among three acoustic similarity estimation methods.	78
4.4	KWS performance of the proposed re-ranking methods, i.e. RAE by score fusion (RAE-SF) and RAE by similarity graph (RAE-SG), on NIST OpenKWS Swahili language pack (evalpart1). “Word”, “morpheme” and “grapheme” refer to results obtained by the word-based, morpheme-based and grapheme-based KWS systems respectively. “MTWV” denotes Maximum Term-Weighted Value, while “MAP” denotes Mean Average Precision. PRF and GBR denote Pseudo Relevance Feedback and Graph-based Re-ranking which are the corresponding baselines of RAE-SF and RAE-SG.	80
4.5	A comparison between the two proposed methods RAE-SF and RAE-SG. M , N , T are number of detections, keyword exemplars and iterations respectively.	81
5.1	The configuration to train the neural network based binary classifier.	90
5.2	The ρ values of original ASR scores and the new confidence scores.	92

5.3	The MTWV and MAP metrics of different models trained with and without novel features.	93
5.4	The MTWV and MAP metrics with and without the second stage, namely TWV-based score normalization.	94
5.5	The MTWV and MAP performance of TWV-scaled and Smooth-ATWV objective functions compared to that of the two-stage framework.	97
6.1	The ability of combining proposed approaches.	103

Chapter 1

Introduction

1.1 Motivation

Recently, computing power and connectivity bandwidth have increased dramatically while storage devices have become cheaper. This has led to more and more data such as broadcast news, voice mail, lecture and presentation recordings being archived. However, these data are often not transcribed and hence cannot be retrieved easily. Therefore, an effective way to retrieve information from large speech corpora has become an important subject of research [4].

Various research topics are directly related to “speech-based information retrieval”, including Keyword Spotting [5], Spoken Document Retrieval (SDR) [6], Spoken Term Detection (STD) [7] (also known as Keyword Search (KWS)) and Query-by-Example Spoken Term Detection (QbE-STD). This thesis focusses on the STD task which aims to detect a given keyword in a text form from a speech corpus. We are now describing these approaches for the speech-based information retrieval.

1.2 Approaches to speech-based information retrieval

1.2.1 Spoken Term Detection

This section introduces the STD task, also known as KWS. The notion of STD was first introduced by the National Institute of Standards and Technology (NIST)[7] in the context of the NIST STD06 evaluation. According to NIST, STD refers to the task

of rapidly detecting the presence of a term (also called keyword or query ¹) in vast, heterogeneous audio archives. Note that the earlier approach for detecting a keyword in a speech corpus is known as Keyword Spotting [8]. However, there is a major difference between STD and Keyword Spotting, which will be mentioned in Section 1.2.2.

Approaches to STD often rely on an ASR, which follow the transcribe-and-search paradigm. Specifically, audio files of the speech corpus are first automatically segmented and then processed by ASR to generate top-1 transcriptions or word/subword lattices. Retrieval techniques can then be applied on these lattices to produce a list of putative occurrences. Each putative occurrence contains an audio name, location and a confidence score. An actual occurrence of the keyword in the speech corpus might be successfully detected or could be missed by the system. A putative occurrence, on the other hand, might be either a correct detection or a mistake. In this thesis, we call an actual occurrence of a keyword a true occurrence, and a putative occurrence returned by the STD system a hypothesized detection (or detection for short). If a detection corresponds to a true occurrence, we call it a hit or correct detection. Otherwise, the detection is a false alarm (FA).

1.2.2 Keyword Spotting

Formerly, the task of detecting keywords in a speech corpus was referred to as Keyword Spotting [8]. It includes Acoustic Keyword Spotting and Event-based Keyword Spotting.

Acoustic Keyword Spotting

A classical approach for Keyword Spotting is to build Hidden Markov Model (HMM) for each keyword and a garbage (or filler) model for other spoken words [5, 9, 10]. The keyword model is constructed by concatenating models of keyword's phonemes while the filler model is constructed using phone loop. Each phoneme can be modeled using a Hidden Markov Model/Gaussian Mixture Model (HMM/GMM) and can be trained with discriminative criteria such as maximizing the likelihood ratio test [11], Minimum Classification Error [12] or Figure-of-Merit [13].

This approach has been shown to be efficient in noisy acoustic conditions [13]. However, its major drawback is that the keyword list needs to be predefined prior to search.

¹In this thesis, we use “keyword” and “query” interchangeably

Hence, the speech has to be re-transcribed if new queries are given. This is also the major difference between the STD and the traditional Keyword Spotting task.

Event-based Keyword Spotting

Recently, Jansen and Niyogi [14] have proposed an event-based Keyword Spotting system. The idea of this approach is to represent speech signal by a set of sparse phonetic events (or landmarks). The keyword observation, which should be a spike pattern, is modeled by the Poisson process model. It is reported that in noisy and sparse training data conditions, the proposed event-based approach achieved comparable performance to the HMM-based approach. Another event-based approach was proposed in [15], where the phone posteriorgram sequence of speech signal is reduced to a shorter sequence of phonetic events for the task of digit detection. [16] combined above two approaches to achieve smaller set of events while still obtaining better performance compared to individual approaches.

1.2.3 Spoken Document retrieval (SDR)

Another speech-based information retrieval task, which resembles Google search, is SDR. SDR refers to the task of finding a subset of spoken documents from a speech corpus that are relevant to a text query [6, 17–19]. SDR and STD have a fundamental difference: SDR aims to retrieve entire speech segments and does not concern about the exact position of each keyword’s occurrence; on the other hand, STD aims to find all occurrences of the keyword. Besides the above difference, many techniques can be used for both STD and SDR such as subword approaches, lattice indexing, etc.

1.2.4 Query-by-Example Spoken Term Detection (QbE-STD)

Recently, a variation of the STD task, namely QbE-STD has been introduced and begun to draw more attention in the speech research community. Similar to the STD task, the QbE-STD task aims to locate occurrences of a keyword in a speech corpus. However, there are some major differences between the STD and QbE-STD tasks. First, in the STD task, queries are in textual form; while in the QbE-STD task, queries are audio snippets that contain a keyword of interest [20]. Second, in the standard STD, linguistic resources including parallel speech/text transcription and pronunciation dictionary are

available; whereas in the QbE-STD task, researchers are often interested in the scenario where only speech signals are available [21–23].

Approaches for QbE-STD can be divided into two categories. The first category is to apply a phone recognizer trained using a labeled data to generate nbest/lattice representations for both query and audio corpus, then matching is performed on these representations [24, 25]. The second category is based on the template matching paradigm [26]. Specifically, acoustic features, such as Mel Frequency Cepstral Coefficient (MFCC) [27–29], posteriorgam [20, 25, 30, 31], spectrogram image [32, 33] and bottleneck features [34, 35], are first extracted for both query and test audio files. Then Dynamic Time Warping (DTW) or its variants such as segmental DTW [30] and segment-based DTW [31], is used to match the query features with test audio features.

1.3 Challenges and existing approaches for STD

There are two problems with common STD systems. The first problem is related to the high miss rate, i.e. true keyword occurrences do not appear in the index of STD systems. There are two main reasons for this problem. The first reason is that the keyword to be searched does not appear in the lexicon of the ASR component of the STD system, hence it cannot be recognized and hence cannot be searched. It is referred to as Out-of-Vocabulary (OOV) problem. The second reason is that the keyword hypothesis is pruned during ASR decoding process due to low acoustic or language models likelihood. As a result, STD systems might have high miss rate.

The second problem is the high false alarm rate due to the low quality of common detection’s confidence score. STD systems rank and make “accept/reject” decisions based on the detection scores and hence these scores play an important role in performance. The detection scores are commonly the lattice-based posterior probabilities [36, 37] of the keyword at detected locations. These probabilities are estimated from the acoustic and language model scores using the forward-backward algorithm. However, these scores might not be robustly estimated due to imperfect modelling of acoustic and language models, e.g. overfitting problem [38, 39]. Past research also observed that posterior probabilities are not comparable across keywords [40]. Specifically, the same posterior

probability threshold results in different performance for different keywords. Therefore, the task of re-scoring hypothesized detections [40–55] has recently received much attention.

Several strategies have been proposed to reduce the miss rate of STD systems. To address the OOV problem, subword-based approaches have been proposed. In subword-based STD systems, the keyword is transformed into a subword sequence, which is then used to search on subword lattices generated by the ASR for the corpus. By doing so, the OOV problem is mostly solved since subword dictionaries have much higher coverage than word-based dictionaries. Some typical subword units are phones [56–61], syllables [62–64], morpheme [65].

To address the problem of missing true keyword hypothesis in the STD index, there are two strategies. The first strategy, referred to as query expansion [66, 67], transforms the keyword into other word/subword sequences that might be confused with the actual keyword, then use them additionally to perform retrieval. The second strategy, referred to as fuzzy matching [57, 63, 68], estimates the distance between the subword sequence of the keyword and a lattice, and produces a matching if the distance is lower than a specific threshold. In these strategies, a confusion model that encodes the insertion/deletion/substitution costs between subword units is required. Additionally, these strategies are also computationally expensive.

Various re-scoring methods have been proposed to address the high false alarm rate problem, and they can be divided into two categories: score normalization and re-estimation. The score normalization methods, e.g. Keyword Specific Threshold (KST) [41, 42] and Sum-to-One (STO) [43] normalization, aim to normalize lattice-based posterior probability to optimize the Term Weighted Value (TWV) [7], which is a widely used measure to compare the performance of STD systems. The TWV metric has a larger penalty for missing a rare keyword than missing of a frequent keyword. Additionally, the penalty of missing a true occurrence is also higher than that of a false alarm. The KST and STO methods take into consideration the properties of the TWV metric to improve STD performance. These methods, however *monotonically* transform scores of hypothesized detections and hence do not re-rank hypothesized detections.

The score re-estimation methods, on the other hand, utilize additional sources of information to perform re-scoring. They are able to re-rank hypothesized detections.

One group of score re-estimation methods, e.g. Pseudo Relevance Feedback (PRF) [47] and Graph Based Re-ranking (GBR) [44–47], rely on the acoustic similarity between hypothesized detections [44–48]. In PRF, the hypothesized detections are sorted according to their scores and the top ranked hypothesized detections are assumed to be correct while the bottom ranked hypothesized detections are assumed to be false alarms. The score of each hypothesized detection is then re-estimated based on its similarity to the top/bottom ranked hypothesized detections. In GBR, a graph is built from the hypothesized detections, with each node representing a hypothesized detection and each edge representing the similarity between two hypothesized detections. The graph-based scores are first initialized with their lattice-based posterior scores and then refined iteratively by the Graph Random Walk (GRW) algorithm which makes use of both initial scores and the similarities between hypothesized detections. Ultimately, the GRW tends to give similar scores to two acoustically similar hypothesized detections and vice versa. One advantage of PRF and GBR is that they do not require any training data. However, these two methods highly depend on the quality of the original lattice-based posterior probability, which may be poor.

Another group of score re-estimation methods rely on a score prediction model trained from data. Usually, the data consist of both hits and false alarm detections of keywords, and the model is trained as a two-class classification problem, i.e. to predict whether an input detection is a true or false class. These methods differ in the features and model structures used. Features that have been used for score prediction are lexical features [49–51], lattice-based features [44, 52, 53], prosody features [13, 50, 51], etc. Besides features, various modeling methods have been employed, including Logistic Regression [49, 69], Vector Quantizer [70], Multi-Layer Perceptron (MLP) [40, 50, 71], Support Vector Machines [40, 50, 72], and Conditional Random Field [54, 55]. Although many features have been studied, to our knowledge, there has not been any work that fully takes into consideration the relationship between a hypothesized detection and its alternative hypotheses at the detected location. Such information is important since it demonstrates how the hypothesized detection dominates its alternative hypotheses.

1.4 Contributions

In this thesis, three novel methods have been proposed to address the high miss and false alarm rates of STD systems. These methods are:

1) A partial search approach. In the first study, a partial search approach is proposed to address the problem of a high miss rate of subword-based STD systems. In particular, the subword sequence of the keyword is first decomposed to overlapping sub-sequences, or partial sequences, which are then used to search on the subword lattices generated by subword-based ASR systems. The proposed approach reduces the miss rate by capturing detections which contain a portion of partial sequences. To limit false alarm (FA), two pruning methods, Global Threshold Pruning (GTP) and Keyword-Specific Pruning (KSP) are proposed. The former uses the same threshold for all keywords while the latter gradually adjusts a specific threshold for each keyword. Experimental results on the National Institute of Standards and Technology (NIST) OpenKWS15 Evaluation dataset show that the partial search approach with KSP is effective. For OOV keywords on a grapheme-based STD system, our proposed approach improves the Maximum Term-Weighted Value (MTWV) and Supremum Term-Weighted Value (STWV)[73] by 6.9% and 7.4% absolute over the baseline that uses the entire keyword to search. The proposed method also outperforms the query expansion technique and offers faster execution.

The proposed partial search approach is described in Chapter 3, and was published in:

V. T. Pham, H. Xu, X. Xiao, N. F. Chen, E. S. Chng, "Pruning strategies for partial search in Spoken Term Detection", In Proc. of SoICT 2017, pp. 114-119 .

2) A re-scoring framework that uses keyword exemplars. In this study, a re-scoring framework that uses acoustic exemplars of keywords is proposed to address the limitations of PRF and GBR. An acoustic exemplar is a true instance of the keyword found in an annotated speech corpus, e.g. the training data of ASR systems. The basic idea of this approach is to utilize the acoustic similarity scores between the speech signal of hypothesized detections and that of keyword exemplars. Specifically, if a hypothesized detection is acoustically more similar to the keyword exemplars, it is more likely to be a true detection, and its score should be boosted. Given the acoustic similarity, two re-ranking methods are proposed, i.e. re-ranking by score fusion and re-ranking by similarity

graph. Experimental results on the NIST OpenKWS15 Evaluation dataset show that the proposed approach consistently outperforms both PRF and GBR techniques that do not make use of keyword exemplars.

This work is presented in Chapter 4, and was published in:

- V. T. Pham, H. Xu, X. Xiao, N. F. Chen, E. S. Chng, H. Li, “Keyword search using query expansion for graph-based rescoring of hypothesized detections”, in *Proc. of ICASSP, 2016*, pp. 6035-6039.
- V. T. Pham, H. Xu, X. Xiao, N. F. Chen, E. S. Chng, H. Li, “Rescoring hypothesized detections of Out-of-Vocabulary keywords using subword samples”, in *Proc. of Interspeech 2016*, pp. 933-937.
- V. T. Pham, H. Xu, X. Xiao, N. F. Chen, E. S. Chng, H. Li, “Re-ranking Spoken Term Detection with acoustic exemplars of keywords”, in *Speech Communication, Vol 104, November 2018*, pp. 12-23.

3) A re-scoring technique that uses detection’s competing hypotheses. In the third study, a re-scoring technique that makes use of lattice-based competing hypotheses of each detection is proposed. Specifically, for each detection, its competing hypotheses are candidates in the lattice that have similar time information as the detection. From such a set, we derive various features, namely *relative_to_max*, *ranking_score*, *number_of_competing_hypotheses* and *max_sim_hypotheses*, that reflect the relative confidence of the detection as compared to its competing candidates. These features are not captured in lattice-based posterior probability and hence are informative for the re-scoring task.

The re-scoring method is described in Chapter 5 and was published in:

V.T. Pham, H. Xu, N. F. Chen, S. Sivadas, B. P. Lim, E. S. Chng, H. Li, “Discriminative score normalization for keyword search decision”, in *Proc. of ICASSP 2014*, pp. 7128-7132.

1.5 Thesis structure

The rest of this thesis is organized as follows:

In Chapter 2, an overview of STD is first provided. Subsequently, a review of techniques to reduce the miss rate of STD systems is presented, followed by a review of current state-of-the-art re-scoring techniques to address the high false alarm rate in STD.

Chapter 3 describes the proposed partial search framework to address the high miss rate problem of subword-based STD systems. In particular, it first demonstrates how to generate partial sequences for a keyword. It then shows how to search and combine the results of each partial sequence. Subsequently, two pruning strategies, i.e. GTP and KSP, are introduced to control the number of false alarms in the combined results. Finally, detailed experiments are carried out to demonstrate the effectiveness of the proposed framework.

In Chapter 4, the re-scoring framework that uses keyword exemplars is presented. A process to generate keyword exemplars is first introduced, followed by methods to estimate the acoustic similarity scores between hypothesized detections and keyword exemplars. Then, given the acoustic similarity scores, two re-scoring methods are introduced. Experiments and analyses are conducted to demonstrate that the proposed technique outperforms other re-scoring techniques that do not use keyword exemplars.

Chapter 5 then studies a re-scoring technique that makes use of lattice-based competing hypotheses. Specifically, the chapter first introduces the competing hypotheses for each hypothesized detection, as well as several novel features derived from the set. It then describes a two-stage scheme to incorporate those novel features into a re-scoring model, i.e. a binary classifier. Detailed experiments are conducted to demonstrate the performance of the proposed technique.

Finally, the thesis concludes in Chapter 6 with a summary of the contributions and potential future research directions.

Chapter 2

Fundamentals and Previous Works

This chapter introduces background and presents a literature review on the research topic of the thesis. It consists of three main parts described as follows.

In Section 2.1, an introduction to the STD task is presented. In particular, we first introduce STD queries, followed by a block diagram of a typical STD system. We then discuss in detail about individual components of the system. Finally, we describe evaluation metrics for the STD task.

Next, Section 2.2 reviews techniques to tackle the problem of missing true keyword occurrences. In particular, this problem might arise due to two reasons: (1) the keyword to be searched is not in the ASR lexicon, which is referred to as the OOV problem; (2) the keyword occurrence does not appear in the ASR output due to errors of the ASR system and hence cannot be detected. Two corresponding techniques, i.e. subword modelling and approximate matching, are presented to address above issues.

Lastly, in Section 2.3, we review techniques to re-score hypothesized detections. In particular, we first introduce the common scores assigned to hypothesized detections, i.e. lattice-based posterior probabilities, and their limitations. We then review two re-scoring strategies, i.e. score normalization and score re-estimation.

2.1 Spoken Term Detection

This section introduces the STD task, also known as KWS ¹. We first introduce the STD queries. We then describe the block diagram of a typical STD system which consists of

¹In this thesis, we use STD and KWS interchangeably

several components, i.e. Automatic Speech Recognition, indexing/retrieval engine and post-processing. Finally, we introduce several evaluation metrics for the STD task.

2.1.1 STD queries

STD queries consist of a word or sequence of more than one word. They have no linguistic correlation but range in grammatical scope from single words to phrases. The concept of a “word” has different meanings in different languages. For example, in English, a word includes the morphological prefixes and suffixes in a typical written text. However, a “word” in Mandarin and Vietnamese typically corresponds to a single sound (syllable). A STD query is required to have a “recognizable, complete meaning” that a typical user would like to find. For example, the phrase “crosby v. o.” is not a potential query because it is not recognizable by itself.

When evaluating the STD performance, homographs are not differentiated, i.e. transcript comparisons are case-insensitive. For example, the word “bill” and “Bill” are considered the same. In addition, morphological variations of a keyword are not counted as positive matches. Hence “students” and “student” are not matched.

2.1.2 Common architecture of STD systems

Figure 2.1 shows the architecture of a typical STD system. It consists of the following operational blocks: Automatic Speech Recognition, indexing/retrieval engine and post-processing. Specifically, audio segments of the test corpus are processed by ASR to generate outputs such as n-best or lattices, which are representations of multiple recognition hypotheses. These outputs are then used by the indexing engine to form an efficient search index. In the retrieval phase, the retrieval engine takes a textual query and finds all hypothesized detections in the index. Each hypothesized detection is associated with an audio file and timing information, which indicates the detected location, and a confidence score. Once hypothesized detections are generated, post processing procedures are often applied to calibrate scores across different keywords, or to combine results from different STD systems. In following subsections, we briefly describe these modules.

Figure 2.1: A typical STD system.

2.1.3 Automatic Speech Recognition

Figure 2.2 shows the block diagram of a typical ASR system, which consists of five modules: feature extraction, acoustic model, language model, a lexicon, and decoding. These modules are described from Section 2.1.3.1 to 2.1.3.5 respectively. Subsequently, in Section 2.1.3.6, we describe a special type of ASR output, i.e. the lattice, which is normally used to create an efficient STD index.

2.1.3.1 Feature extraction

The feature extraction module is used to generate speech feature vectors $\mathbf{X} = o_1 o_2, \dots$ from the speech waveform s . The aim of this component is to extract useful information and remove irrelevant information such as noise from the speech signal. Usually, the features are computed in a frame-by-frame manner. Specifically, each frame corresponds to a window of 25ms speech and the window shift is 10ms. In the short duration, speech

Figure 2.2: A typical ASR system.

is assumed to be stationary. Some popular features for ASR are Mel Filterbank Cepstral Coefficients (MFCC) [74] and Perceptual Linear Predictive (PLP) [75].

2.1.3.2 Acoustic model

The acoustic model is used to model the statistics of speech features for speech units, e.g. phones, that make up a spoken word. The HMM is the standard technique used in the state-of-the-art acoustic models. It is a powerful statistical framework to model data series.

An HMM is specified by a group of states, transition probabilities and emission probabilities distribution. For speech processing, HMM is typically constrained by the left-to-right topology. Figure 2.3 shows an example of an HMM for a phone with transition probabilities a_{ij} and emission probabilities b_j . The observations, i.e. acoustic features, sequence $\mathbf{X} = o_1 o_2 o_3 o_4 o_5$ can be generated by going through the state sequence 1,2,2,3,4,4,5. The probability of generating the observation sequence given the HMM, which is denoted as H , can be estimated as

$$P(\mathbf{X}|H) = a_{12}b_2(o_1)a_{22}b_2(o_2)a_{23}b_3(o_3)a_{34}b_4(o_4)a_{44}b_4(o_5) \quad (2.1)$$

The emission probabilities are usually modeled by a Gaussian Mixture Model (GMM) [76] or a Deep Neural Network (DNN) [77–79]. In these cases, the acoustic models are denoted as HMM/GMM and HMM/DNN respectively.

Figure 2.3: A typical left-to-right HMM.

2.1.3.3 Language model

The aim of the language model is to capture the linguistic knowledge, e.g. syntactic and semantic, and use such information to improve the recognition accuracy. Specifically, it assigns a probability to each hypothesis word sequence, and hence helps to distinguish between hypotheses that have similar pronunciation, i.e. similar acoustic model scores. The most common language model is n-gram [80]. In an n-gram language model, the probability of a word is conditioned on the previous n-1 words. As a result, the probability of the sentence $\mathbf{W} = W_1 W_2 \dots W_m$ is computed as

$$P(\mathbf{W} = W_1 W_2 \dots W_m) = \prod_{i=1}^m P(W_i | W_{i-1}, W_{i-2}, \dots, W_1) \approx \prod_{i=1}^m P(W_i | W_{i-1}, W_{i-2}, \dots, W_{i-n}) \quad (2.2)$$

If n is equal to 2 or 3, we have “bigram” or “trigram” language models, respectively.

2.1.3.4 Lexicon

A lexicon, or pronunciation dictionary, bridges the acoustic and language models. For instance, if an ASR system uses a phone acoustic model and a word language model, then the word lexicon defines the mapping between the words and the phone set, which is demonstrated as in Figure 2.4. In Figure 2.4, each row presents a pronunciation of a word. Specifically, the word at the leftmost column can be pronounced as a sequence of phones in the remaining columns of the row.

```
ABE  EY1 B
ABED AH0 B EH1 D
ABEDI AH0 B EH1 D IY0
ABEE AH0 B IY1
ABEL EY1 B AH0 L
ABELA AA0 B EH1 L AH0
ABELARD AE1 B IH0 L ER0 D
ABELE AH0 B EH1 L
ABELES AH0 B EH1 L Z
ABELES(1) EY1 B AH0 L IY2 Z
ABELL EY1 B AH0 L
ABELLA AH0 B EH1 L AH0
ABELN AE1 B IH0 L N
ABELOW AE1 B AH0 L OW0
ABELS EY1 B AH0 L Z
```

Figure 2.4: Examples of the Carnegie Mellon University (CMU) pronunciation dictionary. Each row presents a pronunciation, i.e. sequence of phones, of a word.

2.1.3.5 Decoding

The decoding block, which is also called decoder or search block, determines the best hypothesized word sequence that can be selected using the combination between the acoustic and language model scores for the given input signal. Mathematically, the goal of this block is to find the string sequence $\mathbf{W}$ given the input features $\mathbf{X}$ which maximizes probability $P(\mathbf{W}|\mathbf{X})$. Using the Bayes Rule, we have:

$$\hat{\mathbf{W}} = \arg \max_{\mathbf{W}} P(\mathbf{W}|\mathbf{X}) = \arg \max_{\mathbf{W}} \frac{P(\mathbf{X}|\mathbf{W})P(\mathbf{W})}{P(\mathbf{X})} \cong \arg \max_{\mathbf{W}} P(\mathbf{X}|\mathbf{W})P(\mathbf{W}) \quad (2.3)$$

In Equation 2.3, $P(\mathbf{X}|\mathbf{W})$ and $P(\mathbf{W})$ are given by acoustic and language models.

In order to determine the best word sequence given input speech signal, the decoder constructs a search space, i.e. a network of HMM states, from the acoustic model, language model and the lexicon. The connection's roles within this network are defined at different hierarchy levels as shown in Figure 2.5. Specifically, words are connected using language model rule; whereas each word is constructed from a phone sequence defined by the pronunciation dictionary; and each phone is presented by an HMM, i.e. a state sequence, in the acoustic model. The optimal state sequence, or consequently the optimal word sequence, can be found using Viterbi [81, 82] or token passing algorithm [83].

Figure 2.5: The search space of the decoder which is constructed at different hierarchy levels.

2.1.3.6 ASR lattices

Roughly speaking, an ASR lattice is a hypothesis space explored during a decoding process for a speech utterance. For example, considering a token-passing approach for the decoder [83], these lattices are produced by propagating multiple tokens corresponding to hypotheses in addition to that which yields the best score after every word.

Formally, a lattice is essentially a directed, acyclic graph consisting of collections of nodes (vertices) and arcs (edges). Each arc corresponds to a hypothesis generated by the ASR system. The arcs also encode the acoustic and language model likelihoods for the hypothesis. Each node is associated with particular time information within the utterance to be decoded. Additionally, each node may be connected to a number of preceding or following arcs. Figure 2.6 shows an example of an ASR lattice.

From the lattice of an utterance, we can obtain another type of output, i.e. n-best transcriptions, by selecting n-best paths which have highest likelihood scores. In the context of the speech recognition research community, the 1-best transcriptions are used to measure the Word Error Rate of an ASR system. The lattice representation is preferred over the N-best transcriptions due to two major reasons: (1) It contains more hypotheses than N-best, which results in higher recall in a STD system; (2) STD systems can readily access information pertaining to the hypothesized words and their associated temporal information.

Figure 2.6: Example of an ASR lattice [1].

Researchers in [84–86] highlighted that raw lattices generated by ASR systems contain redundant information. In particular, a lattice might contain multiple overlapped arcs with the same label. Hence, various techniques have been proposed to compress the original lattices to efficient representations prior to indexing. Those representations include Position Specific Posterior Lattice (PSPL) [84], Confusion Network (CN) [19, 87, 88] and Time-based Merging for Indexing [85, 86]. Figure 2.7 shows an example of CN and PSPL representations generated from the same raw lattice. It can be seen that the CN and PSPL structures are more compact than the raw lattice; hence the indexes obtained from these representations require less memory than those obtained from the raw lattice [62]. However, these representations are lossy, i.e. some word sequences included in the original lattice are discarded. For example, the sequences $W_3W_4W_5$ in the raw lattice cannot be found in the CN structure. In this thesis, we perform indexing on the raw lattice.

2.1.4 Indexer and search engine

Once ASR outputs are generated, they need to be indexed prior to search for efficiency [7]. In general, speech-based indexing is more difficult than classical text-based indexing since various types of ASR outputs, such as n-best transcriptions or lattices, can be used by the indexer. Section 2.1.4.1 presents techniques to index n-best ASR transcriptions. Subsequently, Section 2.1.4.2 describes various techniques to index ASR lattices. Finally, two-stage approaches are introduced to further speed up the retrieval process.

Figure 2.7: A raw lattice and the corresponding PSPL and CN structures.

2.1.4.1 Indexing N-best ASR transcriptions

Although indexing and retrieval on lattices can achieve high performance, this approach is expensive in terms of indexing/retrieval time and storage. When the audio corpus is very large, e.g. more than 10,000 hours of audio, indexing and retrieval on lattices might be impractical [89]. In this situation, using the 1-best or n-best outputs instead of lattices can reduce complexity.

A common technique to index n-best ASR output is inverted index [68, 90, 91], which is borrowed from the traditional text-based IR. The inverted index is a data structure that maps each “term” to a list of “documents” (also called the postings list) that contain the term. Figure 2.8 shows the structure of the inverted index. In the context of STD, the inverted index is used to index 1-best phone sequences in [90]. Specifically, 1-best phoneme sequences are transformed into overlapped N-grams phoneme sequences and these N-grams are then stored in the inverted index. In other words, N-grams play the role of “terms” while speech utterances play the role of “documents” in the inverted index. During the search phase, a query phoneme sequence is also converted into a

list of N-gram sequences before searching on the index. The N-gram inverted index technique is also used to index n-best syllable sequences in [64], which supports ASR error compensation. Specifically, to allow substitutions, all possible syllables at a position can be combined with all possible syllables at another position when generating lists of N-gram. Experimental results show that the performance of that method is comparable with the standard Dynamic Time Warping method while reducing the search time significantly.

Figure 2.8: The inverted index structure.

Suffix array is another technique for indexing 1-best transcription [89]. It is a data structure to store the sorted indexes of all suffixes of a sequence. This method has some advantages when indexing and retrieval with large database sequences. Firstly, the binary search can be performed on the suffix array as the indexes are sorted by alphabet order. Secondly, only a small amount of memory is required since the array holds only the indexes. Additionally, this data structure allows approximate matching to compensate for recognition errors.

2.1.4.2 Indexing and retrieving ASR lattices

Once ASR lattices are generated, the retrieval process can be performed directly on lattices by traveling on these lattices to match the desired keyword with arc labels. However, this technique is slow and does not scale well on a large speech database. Hence, it is required to index the ASR lattices prior to search for efficiency [7]. Various techniques have been proposed to index and search on ASR lattices, which can be divided

into two groups: N-gram inverted index and Weighted Finite State Transducer (WFST). Those are discussed as below.

N-gram inverted index

The first group of techniques for lattice indexing relies on the inverted index. Specifically, the index term can be unigram word [60, 92] or n-gram subword [57, 93–96]; while posting list might be lattice arcs [92] or lattice paths [96] that contain the term. During the retrieval, if the keyword to be searched contains only one index term, the posting list of the term is retrieved and returned to users. In another case, when the keyword contains more than one index term, posting lists of these terms are obtained and merged together to generate the list of hypothesized detections. The confidence score of a hypothesize detection is approximated, e.g. by summing or averaging, from scores of items in individual posting lists. It is claimed that this technique only leads to sub-optimum results [97] due to the process of merging multiple posting lists.

Weighted Finite State Transducer based indexing

Another group of techniques to index ASR lattices [97–100] relies on Weighted Finite State Transducer (WFST). In these studies, the authors aimed to build an exact index structure which efficiently stores all substring occurrences in a lattice, thus addressing the problem of the N-gram inverted index. At the indexing stage, each utterance is considered as a transducer where the states of the transducer are nodes in the corresponding lattice. New starting and ending states are then created which connect to all existing states as shown in Figure 2.9. At the retrieval stage, the keyword is first converted into the transducer format, which is actually an acceptor (input and output labels are identical), and then composed with the index transducer to generate the result, i.e.

$$\textit{Search result} = K \circ I. \quad (2.4)$$

where K is the keyword acceptor and I is the index transducer.

The advantage of WFST over the n-gram inverted index technique is that WFST allows searching with any sequence of labels without breaking it to several sub-sequences. In particular, since any sequence of arcs in the lattice can be connected to the new starting and ending states, it is “accepted” by the transducer and so its label sequence can be retrieved.

The second advantage of WFST is fast retrieval. Specifically, after applying transducer operators such as minimization and determinization, the index WFST becomes deterministic. As a result, the search time is linear with the query length. This benefit can be seen more clearly when applying the WFST technique on subword STD where the subword representation of the query is often long.

Figure 2.9: WFST indexing.

Another advantage of WFST is that it easily supports approximate matching [3, 101, 102]. Specifically, approximate matching allows some degree of mismatches, i.e. insertion/deletion/substitution, between the query’s WFST and the index’s WFST. Such capacity is important to STD to reduce the miss rate of STD systems. The approximate matching is discussed in detail in Section 2.2.2.

Due to above advantages, in this thesis, we choose the WFST as the indexing/retrieval framework. It is realized in the Kaldi toolkit [103]. All experiments in following chapters of this thesis are conducted on the Kaldi toolkit.

2.1.4.3 Fast retrieval

There are various applications that require fast retrieval capacity, e.g. applications to monitor broadcasts or call center operations [104]. A common approach for fast retrieval is based on a two-stage search: the first stage is to rapidly select a small selection of possible items, i.e. lattices or paths of a lattice, that putatively match with the query; and the second stage is to apply a more detailed search on the putative items to obtain the final results.

The two-stage search approach was first proposed in [105] for the SDR task. In that study, a phone-trigram inverted index was used to perform the first stage search. Specifically, speech regions that contain a certain number of phone-trigrams of the query are identified as putative hits. Note that there is a trade-off between the speed and the performance of the retrieval system when selecting trigram threshold, i.e. the value to determine if a region potentially contains the keyword. If the threshold is high, only a small number of putative regions are returned which leads to faster search at the second stage. But this leads to a higher miss rate. On the other hand, if the threshold is low, a large number of putative regions are returned in the first stage leading to a lower miss rate but slower search at the second stage.

In [61], Nozouzian et. al combined the query expansion technique with the two-stage search on a word-based index to form an efficient open vocabulary search framework. In this method, the inverted index is used: index items are words in the dictionary and their posting entries are lattice paths that potentially include the words. In the first phase, each word of a query is expanded to the most similar words in terms of pronunciation and then the corresponding posting entries, i.e. lattice paths, of those words are retrieved for the second phase. In the second phase, each lattice path is converted to a phoneme sequence and then approximate matching is applied between the query and the phone sequence.

2.1.5 Post-processing

Once search results are generated, post-processing procedures can be applied to improve the KWS performance. These include re-scoring hypothesized detections, which aims to calibrate scores from different keywords, and system fusion which aims to combine search results from diverse STD systems.

2.1.5.1 Re-scoring hypothesized detections

The common confidence score of each hypothesized detection is the lattice-based posterior probability [36, 37] of the keyword at a detected location. In the rest of this thesis, we denote such score as the raw ASR score or raw detection score. This score is estimated from the acoustic model and language model scores. However, the raw ASR score might

not be robustly estimated due to imperfect modelling of acoustic and language models. The limitation of the lattice-based posterior probability will be discussed in detail in Section 2.3.1.2. As a result, the task of estimating a new set of confidence scores is necessary and has received much attention recently.

The new scores can be derived from lattice-based posterior probabilities through monotonic transformation functions, which are referred to as score normalization. The new scores can also be estimated using modelling techniques such as Multi-Layer Perceptron (MLP) [40, 50, 71], Support Vector Machines [40, 50, 72], and Conditional Random Field [54, 55], which are referred to as confidence estimations [51, 106]. In all these cases, the original lattice-based posterior probabilities are used to estimate the new confidence scores. Therefore, we refer to this task as re-scoring hypothesize detections. The literature review on re-scoring techniques will be presented in detail in Section 2.3.

2.1.5.2 System combination

Under noisy and diverse acoustic conditions, it is challenging to build a single robust STD system. One suggested solution to address this issue is to build multiple STD systems with different robust components. In such the scenario, the task of system combination is desirable. For example, the system combination is useful when combining word and subword systems since they are complementary. Specifically, word-based STD systems normally yield higher accuracy than subword system, but they cannot deal with the OOV problem. Subword-based STD systems, on the other hand, can address OOV problem but they often have lower accuracy. Several strategies can be applied for system combination, i.e. prior combination, ASR output combination and search results combination.

Prior combination

Prior combination refers to a strategy to combine word and subword units at the decoding level. In this strategy, the lexicon and training transcriptions contain both word and subword units which can be used to train a hybrid word/subword ASR. The most difficult step in this strategy is how to obtain the hybrid word/subword transcriptions from the original training word transcriptions to train the Language Model (LM). Generally, in order to ensure that enough LM training data is generated, only the most frequent words in the training word transcriptions are preserved, other words are converted to subword representations [107, 108].

In [2], researchers combined words and phonemes dictionaries to form a hybrid word-phoneme recognizer. Experimental results showed that the hybrid system outperforms the phoneme system for both OOV and In-Vocabulary (IV) queries. The prior combination was further studied in [4, 107, 109] in which various subword units such as grapheme and multigram were combined with word units to form hybrid systems. Generally, such hybrid systems produce a comparable performance for both IV and OOV queries compared to stand-alone systems. The primary advantage of this approach is faster decoding speed and smaller index size as compared to the stand-alone systems.

ASR output combination

ASR output combination is performed after decoding but before indexing. Word and subword outputs, e.g. ASR lattices, are generated separately and then combined together to obtain the final output for indexing and retrieval.

In [2], Yu et. al proposed a simple approach, namely utterance-level combination, to combine word and phone lattices into a single lattice. This is done by creating new start and end nodes that connect to the start and end nodes of the original lattices. However, this approach only produced a moderate retrieval result, which might be due to the mismatch between word and phoneme language model scores. Another work for ASR output combination was proposed in [110] which used WFST framework. In that study, for each utterance, word CN and phoneme CN were first generated and presented as automata, and then the composition operator was applied on these automata to obtain the combined network. Figure 2.10 demonstrates the combination procedure. The advantage of this method is that it supports multi-pronunciation and mixed word-phone queries since queries are also represented as flexible automata. However, a disadvantage of this approach is that incorporating word and subword lattices can increase the index size dramatically.

Search results combination

Search results combination is performed after searching. For each query, hypothesized detections returned by different STD systems are merged into a single detection if they overlap in time. This strategy is described in Figure 2.11. The key issue of the combination is how to combine scores of different systems.

Several data fusion methods for IR were investigated for search results combination. The fused score can be estimated by taking the maximum (CombMax) [43, 111] or

Figure 2.10: ASR output combination [2] using WFST.

Figure 2.11: The search result combination procedure.

summing (CombSum) [43, 112] on scores of participated systems. To emphasize a combined detection that appears multiple times, CombMNZ [43] can be used. In another method, WCombMNZ, was proposed [43] to incorporate different contribution of participating systems. In this method, the contribution weights are proportional to Maximum Term-Weighted Value (MTWV) of the participated systems. Experimental results in [43] indicated that WCombMNZ slightly outperforms CombSum and CombMNZ. Those approaches offer several advantages such as simplicity, fast combination and also do not require any training model. However, ad-hoc rules such as CombSum or CombMNZ might not fully capture systems and keyword properties, and hence these methods might not lead to the optimum combination.

Several complex combination models, which aim to capture systems and keyword characteristics, were proposed in [43, 112, 113]. In these approaches, inputs are scores of participating systems as well as other features such as query length, number of system

voting, etc. and the outputs are 0/1 values which indicate whether the combined detection is correct or not. These approaches help to integrate useful information into the combination procedure, thus improving the STD performance.

2.1.6 Evaluation metrics for STD

The search result produced by a STD system is a list of hypothesized detections. Each detection is represented as a tuple (audio, start time, duration, score). To evaluate the performance of the STD system, each hypothesized detection is aligned with the reference transcriptions to determine the correctness of the detection. If the hypothesized detection corresponds to a true occurrence, it is called a hit; otherwise, it is called a false alarm (FA). A true occurrence that is not detected by the STD system is referred to as a miss. A hypothesized detection is only considered as a true occurrence if the time gap between the mid-point of the hypothesized detection and mid-point of the true occurrence is less than 0.5 seconds [7].

STD performance can be evaluated via several metrics [60,99]. In particular, the STD can be considered as a retrieval task and popular information retrieval metrics such as Mean Average Precision (MAP) or precision-recall, can be used to evaluate the performance of a STD system. Besides, NIST has also proposed to use other metrics called Term-Weighted Value (TWV) and Detection Error Tradeoff (DET) curves to evaluate the STD performance. These metrics are briefly discussed as follows.

Mean Average Precision: MAP is one of the most popular evaluation metrics which measures the ranking performance of an IR system. Given the set of query $Q = \{q_k \mid k = 1, 2, \dots\}$ we define N_{q_k} as total number of true occurrences of the keyword q_k in the test corpus. Assume that each keyword q_k has n_{q_k} hypothesized detections $\{d_i \mid i = 1, 2, \dots, n_{q_k}\}$. Then the MAP of the STD system is formally defined as:

$$MAP = \frac{1}{|Q|} \sum_{q_k \in Q} \frac{1}{N_{q_k}} \sum_{i=1}^{n_{q_k}} \frac{\mathbf{I}_{[d_i \text{ is correct}]}}{i} \quad (2.5)$$

where $|Q|$ denotes the number of keywords in the set Q and $\mathbf{I}_{[condition]}$ denotes the indicator function which equals 1 when *condition* is true.

Term-Weighted Value: Currently, TWV is the major evaluation metric for the STD task [7]. It is calculated as follows:

$$TWV(\theta) = 1 - \frac{1}{|Q|} \sum_{q_k \in Q} ((P_{miss}(q_k, \theta) + \beta P_{fa}(q_k, \theta)) \quad (2.6)$$

where $P_{miss}(q_k, \theta)$ and $P_{fa}(q_k, \theta)$ are probability of miss and probability of false alarm of the query q_k with respect to a detection threshold θ . Specifically,

$$P_{miss}(q_k, \theta) = 1 - \frac{N_{correct}(q_k, \theta)}{N_{q_k}} \quad (2.7)$$

$$P_{fa}(q_k, \theta) = \frac{N_{spurious}(q_k, \theta)}{T_{speech} - N_{q_k}} \quad (2.8)$$

where:

- $N_{correct}(q_k, \theta)$ is the number of correct detections of q_k (the keyword) which have scores greater than or equal to the threshold θ ,
- $N_{spurious}(q_k, \theta)$ is the number of incorrect detections which have scores greater than or equal to the threshold θ ,
- T_{speech} is total length of speech in the test data (in seconds).

Note that the denominator in Equation 2.8 is the number of opportunities for incorrect detections of q_k in the corpus. Additionally, the β in Equation 2.6 reflects the relative importance of the false alarm cost as compared to the missing cost, which is calculated by

$$\beta = \frac{C}{V} (Pr_{keyword}^{-1} - 1) \quad (2.9)$$

where

C: The cost of incorrect detection, conventionally defined as 0.1.

V: The value of correct detection, conventionally defined as 1.

$Pr_{keyword}$: The prior probability of a *keyword*, conventionally defined as 10^{-4} .

The cost-value ratio, $\frac{C}{V}$, indicates how willing the users accept false alarms to obtain more hits. Ideally, this ratio should be chosen based on certain applications to control the miss and the false alarm rate. For example, in audio surveillance applications, users might prefer low miss rate, thus the cost-value ratio should be small.

The TWV of a STD system ranges from 1 (a perfect system) to $-\infty$ (the worst system). A STD system that outputs nothing has TWV equals 0. There are two significant values

of TWV: Actual Term-Weighted Value (ATWV) which measures the system performance for a particular chosen configuration, i.e., the TWV at a certain threshold; and the Maximum Term Weighted Value (MTWV) which is the maximum TWV over all possible threshold values. It can be seen that ATWV is sensitive to threshold values. Practically, the potential threshold is chosen to be the optimum point based on results evaluated on the development data. Usually, the threshold that is selected on the development data is not optimal for the evaluation set, and hence the ATWV usually does not equal the MTWV.

Detection Error Tradeoff curve: Although ATWV and MTWV provide convenient measures to compare the performance of STD systems, these measures are single-point values. To see the full picture of the performance of a STD system with respect to the threshold θ , which controls the probability of miss and false alarm, the DET curve can be used. The DET curve is a visualization of a STD system performance by plotting P_{miss} versus P_{fa} when varying the threshold values. Specifically, in the DET curve, the x-axis represents the probability of a false alarm and the y-axis represents the probability of a miss.

2.2 Approaches to reduce miss rate of STD systems

One major issue of common STD systems is their high miss rate. Note that missing a true occurrence of a keyword occurs in two conditions: (1) the true keyword occurrence does not appear in the index, hence it cannot be detected by the STD system; (2) the true keyword occurrence is still detected by the STD system with low confidence score, thus it is rejected when being compared with a threshold value. The second condition can be tackled with re-scoring techniques, which are discussed in Section 2.3. This section reviews techniques to address the first condition.

There are two main reasons for missing a true keyword occurrence in an index of a STD system. The first reason is that the keyword to be searched does not appear in the lexicon of the ASR component of the STD system. The second reason is that the keyword hypothesis is pruned during ASR decoding process due to low acoustic or language models likelihood. For the OOV problem, various subword modelling techniques

have been studied which will be introduced in Section 2.2.1. In the second case, it is desirable to allow “approximate matching” between the keyword and the STD index, which will be reviewed in Section 2.2.2.

2.2.1 Subword approaches for STD

In the context of STD and ASR, OOV words are those words absent in the vocabulary of an ASR system. As a result, these words cannot be recognized and hence cannot be detected. The OOV problem is critical for the STD task because of two reasons. Firstly, many OOV words are name entities [114], hence they are informative and potential to be keywords. Secondly, the OOV problem will become more and more serious if we do not address them because there are more and more OOV words have been being created. Specifically, it is reported [115] that there are around 1,000 new words have been created every year. Therefore, it is essential for STD systems to support searching OOV keywords. A common solution is to use subword units.

A wide range of subword units have been proposed, including linguistical units such as phones [56–61, 92, 98, 116], syllables [62–64] and morpheme [65], as well as data-driven units called word-fragment [117], multigrams [94, 118], graphone [107] and particles [101]. Linguistical units are derived using linguistic knowledge such as phonological and stress information [119], and data-driven units are derived using statistical and informational theoretic principles [1].

Linguistical subword units

Typically, phonemes are the common linguistical subword units to handle OOV queries. The phoneme-based approach was first proposed for SDR task in [68, 92, 120] and has been applied for STD in [56–61]. In this approach, a query is converted to a phoneme sequences using a pronunciation dictionary or a Letter-to-Sound tool for OOV queries. Then the phoneme sequence is searched on the 1-best phoneme sequences or phoneme lattices of a corpus. In general, the use of phoneme improves recall but comes at the cost of a high false-alarm rate. In particular, phoneme recognition accuracy is lower than word recognition accuracy due to the short acoustic evidence of the phoneme units as well as the poor language model predictability [1]. Generally speaking, the shorter the acoustic unit, the lower the recognition accuracy.

For some languages such as German, Chinese, Japanese or Vietnamese, using syllables as subword units for the STD task is more effective than the phoneme units. In [63], syllables were used as subword units for German STD because they presented a good tradeoff between robustness in terms of acoustic context and inventory size. For Chinese, syllables were also used [62] as the number of syllables in Chinese is small, and most Chinese words are made from one or two syllables. For Japanese, there are only about 110 syllables, and hence syllables are suitable for Japanese STD [64]. Since we need knowledge about the specific language, it might be costly to extract these linguistic subword units.

Data-driven subword units

The data-driven approach is an alternative strategy to derive subword units in the absence of linguistic information. Data-driven subword units can be obtained by maximizing an objective function on a given training data. These subword units include word-fragments, graphemes, morpheme.

Word-fragments are data-driven units that were used for SDR task in [117] and then for the STD task in [96]. Word-fragments are variable length phoneme sequences which are selected using a bottom up clustering algorithm. Specifically, at each step, two units (phone or phone sequences) which have high mutual information are merged to form a single unit. It is believed that word-fragments can lead to better phone accuracy compared to phone decoder due to the strong constraints on the phonetic string [121].

In [107], Akbacak et. al proposed to use subword units called graphemes for STD task. Each grapheme is a pair of a letter sequence and a phoneme sequence of possibly different lengths obtained during training the Joint Multigram Model (JMM) on a dictionary. Figure 2.12 shows an example of graphemes. For the STD task, the hybrid word/grapheme STD system [107] is able to search for both IV and OOV keywords.

Other subword units, namely morphemes, are effective for morphological rich languages such as Turkish or Tamil [122]. Morphemes are smallest grammatical units, i.e. they cannot be broken into smaller meaningful units. For example, the word “unpredictable” consists of three morphemes “un”, “predict” and “able”. Such units can be obtained by a data-driven approach using maximum likelihood estimation [123]. For languages such as Finnish, Turkish or Tamil, words are composed of many morphemes,

Figure 2.12: Example of the grapheme subword units.

and so the OOV rate for such languages is high. Representing data, i.e. keywords and ASR outputs, in morphemes units addresses the OOV problem. Additionally, previous studies [48, 124] shown that the morphemes-based approach works well for IV keywords.

2.2.2 Approximate matching for STD

Subword modelling is insufficient to address the high miss rate problem. Even if the keyword and ASR outputs are converted to subword representations, the true keyword’s occurrences might be miss-recognized by the subword ASR due to the inherent high error rate of the subword ASR system. Specifically, subword units are short and have low constraint than word units, and so subword ASR systems often have lower accuracy than word-based systems. Therefore, approximate matching is desirable to compensate for the low subword recognition accuracy. This can be done through two approaches: query expansion and fuzzy searching. In these techniques, a confusion model describing how the confusion among subword units is needed. The confusion model is described in Section 2.2.2.1, then the two approximate matching techniques, i.e. query expansion and fuzzy searching, are presented in Section 2.2.2.2

2.2.2.1 Estimating the confusion model

As mentioned above, approximate matching techniques require a confusion model that captures insertion, deletion and substitution mismatches between subword units, usually phonemes. The mismatch costs can be evaluated in several ways: using heuristic setting [10, 41, 57], using confusion matrix constructed from a phone recognizer [59, 125] or using a model distance [126, 127].

A simple way to derive the mismatch costs between phonemes is based on prior empirical knowledge. In [10, 41], the phoneme substitution, insertion, deletion penalties are set equally. The obvious limitation of this approach is that it does not describe the variety of subword units. In [57], the penalties are decided by some rules based on the broad acoustic-phonetic classes. However, the heuristic class-based penalties are not representative in case of substitutions because the confusions between different phoneme models also depend on the amount of training data, nature of training data and other factors.

In [59, 125], the mismatch cost was derived from a confusion matrix. The model can be constructed using a labelled data by the following steps:

- (i) generate the subword decoding transcription of the data set;
- (ii) generate the true subword sequence of the data using force-alignment;
- (iii) for each audio file, align the decoding sequence with the corresponding true sequence to get the insertion/deletion/substitution errors information;
- (iv) gather the errors on entire data set to build the confusion model.

Confusion matrix helps to improve the precision of mismatch costs, hence making KWS more accurate when comparing with empirical cost. Note that although the confusion of bigram or trigram subwords can be considered [68], it is not widely used because of lack of training data.

The mismatch costs of phonemes can also be derived from the similarity between the models of these phonemes. In [126], Itoh et. al estimated the phoneme confusion using the distance between two HMMs. The basic idea of this method is to calculate the distance between two distributions using the Bhattacharya distance. Iwata et. al in [127] evaluated the distance of two phonemes using Kullback-Leibler divergence (KLD) distance. This was motivated by the fact that KLD is more general and effective in measuring acoustic distortion compared to phoneme edit distance or other heuristic measures.

2.2.2.2 Approximate matching techniques

Approximate matching can be realized through two techniques: query expansion and fuzzy matching. In query expansion, the keyword to be searched is first transformed into new sequences, that potentially appear in the ASR output, then those sequences are used to search [112, 128]. In other words, this approach tries to “mimic” the behavior of the subword ASR system. The query expansion technique is also used to address the OOV problem. Specifically, OOV keywords are expanded to the most similar IV words [3, 101, 102], called as proxies, in terms of phonemic distance, and then these proxies are used to search on a word-based index.

In fuzzy matching [68, 129], a distance between the keyword to be searched and the ASR outputs is first estimated. Then, any hypothesis in the ASR output that has the distance lower than a pre-defined threshold is considered as a detection of the keyword.

Under the well-known WFST-based framework, both query expansion and fuzzy matching can be realized in an uniform framework by using the composition operator. In this framework, the keyword, confusion model and the index are converted to WFST format, then retrieval results are obtained by composing these WFSTs as follows:

$$\textit{Search result} = K \circ C \circ I. \quad (2.10)$$

where $\circ$ is the composition operator, K is the keyword WFST, I is the index WFST and C is the confusion model which encodes insertion, deletion and substitution costs as shown in Figure 2.13.

2.3 Re-scoring Spoken Term Detection

This section first introduces the lattice-based posterior probability, which is the common confidence score assigned to a hypothesized detection, and its limitations. Then, a literature review of state-of-the-art re-scoring techniques for STD is presented.

2.3.1 Lattice-based posterior probability and its limitations

2.3.1.1 Lattice-based posterior probability estimation

Given a speech signal $\mathbf{X}$, the posterior probability of a word/subword hypothesis W at a specific location can be estimated from the ASR lattice, denoted as L , generated during

Figure 2.13: Example of a confusion model [3].

decoding $\mathbf{X}$. To do so, we first estimate the posterior probability scores of all arcs in the lattice L . We then accumulate the posterior probabilities of arcs in L that are considered to be the same hypothesis W . These two steps are discussed below.

Posterior probability of an arc in a lattice

The posterior probability of an arc a in L is computed as the ratio between the probability of paths that pass through a over the probability of the observation $\mathbf{X}$, i.e.

$$p(a|\mathbf{X}, L) = \frac{\sum_{\mathcal{P} \in L, a \in \mathcal{P}} p(\mathcal{P}, L)}{p(\mathbf{X})} \quad (2.11)$$

In Equation 2.11, the numerator is the sum of all paths $\mathcal{P}$ that pass through the arc a while the denominator is the probability of the observation $\mathbf{X}$. Estimating $p(\mathbf{X})$ requires summing over all possible hypotheses, which is intractable. In practice, $p(\mathbf{X})$ can be approximated by summing of likelihood scores over all complete paths in the lattice L , i.e.

$$p(\mathbf{X}) = \sum_{\mathcal{P}' \in L} p(\mathcal{P}'|L) \quad (2.12)$$

In Equation 2.12, $p(\mathcal{P}'|L)$ is the likelihood score of a complete path $\mathcal{P}'$, i.e. a path comes from the starting node to the ending node of the lattice. Substituting Equation 2.12 into the denominator in Equation 2.11, we have:

$$p(a|\mathbf{X}, L) = \frac{\sum_{\mathcal{P} \in L, a \in \mathcal{P}} p(\mathcal{P}, L)}{\sum_{\mathcal{P}' \in L} p(\mathcal{P}'|L)} \quad (2.13)$$

Equation 2.13 can be efficiently computed using forward-backward algorithm [130]. Specifically, let $\alpha(a)$ be the forward probability which is the probability of partial paths

(i.e. arc sequences) from the start node of the arc a until the start node L_{START} of the lattice L ; and $\beta(a)$ be the backward probability which is probability of partial paths from the end node of a to the end node L_{END} of the lattice L . $\alpha(a)$ and $\beta(a)$ can be recursively estimated as

$$\alpha(a) = \sum_{a' \in PREC(a)} \alpha(a') \cdot p_{AM}(a) \cdot p_{LM}(W_a|h') \quad (2.14)$$

$$\beta(a) = \sum_{a'' \in SUC(a)} \beta(a'') \cdot p_{AM}(a'') \cdot p_{LM}(W_{a''}|h'') \quad (2.15)$$

where $PREC(a)$ and $SUC(a)$ denote the sets of arcs before and after the arc a respectively (see Figure 2.14), p_{AM} and p_{LM} denote acoustic model and language model likelihood scores, W_a and $W_{a''}$ are labels of a and a'' respectively, h' and h'' are word history of a and a'' respectively. It can be seen that the numerator in Equation 2.13 is computed as a product of $\alpha(a)$ and $\beta(a)$; while the denominator is computed as $\alpha(L_{END})$. In other words, Equation 2.13 can be written as:

$$p(a|L) = \frac{\alpha(a) \cdot \beta(a)}{\alpha(L_{END})} \quad (2.16)$$

Figure 2.14: Illustration of the sets of arcs before and after the arc a , i.e. $PREC(a)$ and $SUC(a)$

Posterior probability of a word/subword hypothesis

Once the posterior probabilities of all arcs in the lattice L are computed, the posterior probability of a word/subword hypothesis W at a location can be derived by the posterior probability of an arc a in L that has label W and intersects with the desired location. However, it is observed that there are normally lots of arcs in L that have the same

word/subword label and slightly different starting time or ending time. Hence it is necessary to take into consideration all arcs that have the same label W and overlap with the desired location. One possible way, which is proposed in [37], is to take the maximum of the posterior probability scores of those arcs.

2.3.1.2 Limitations of the lattice-based posterior probability

The first problem of the lattice-based posterior probability is that the measure is usually over-estimated. Note that during decoding, hypotheses having relatively low likelihoods are discarded, and so the ASR lattice is just a pruned search space of the ASR decoder. Subsequently, using all paths in the lattice to approximate $p(\mathbf{X})$, as in Equation 2.12, often leads to under-estimation of $p(\mathbf{X})$, or in other words, results in over-estimation of the posterior probability.

Another problem of estimating a precise lattice-based posterior probability relates to the modelling capacity of the acoustic and language models. From Equations 2.14 and 2.15, the posterior probability score is optimal if the underlying acoustic and language models are optimal, i.e. they are based on optimal probability distribution functions. However, this assumption is impractical for both acoustic model and language model as explained below.

- For acoustic modelling, the well known approaches, i.e. HMM/GMM or HMM/DNN, have several deficiencies: (1) small number of HMM parameters results in a poor representation the variation of speech signal while increasing the number of parameters might lead to overfitting problem. (2) The HMM-based approach makes use of frame-by-frame independence assumption which is unrealistic for speech signal.
- For language modelling, the standard N-gram language model relies only on a fixed length history in the past to estimate the probability of a word, and hence cannot capture long-range context.

Using the lattice-based posterior probabilities as the detection scores might result in degraded STD performance. Firstly, it is observed that the posterior probabilities are not comparable across keywords. In particular, at the same probability scores, different keywords have different precision rates. This is a problem with the TWV metric since

a global threshold is applied to all keywords. Secondly, a correct detection might have a smaller posterior probability than false alarm detections due to poor acoustic or language models. In other words, correct detections might have worse ranking than false alarm ones, which results in low MAP performance. Therefore, re-scoring and re-ranking hypothesized detections has become an important research topic. Various re-scoring techniques have been proposed, which are described in the following section.

2.3.2 Techniques to re-score hypothesized detections

Techniques for re-scoring hypothesized detections can be grouped into two categories. The first category is score normalization which normalizes the lattice-based posterior probability scores so that a single threshold can be applied to all keywords. The second category is score re-estimation which uses other sources of information to estimate new confidence scores. These are presented in the following sections.

2.3.2.1 Score normalization

For the TWV, i.e. the primary metric of STD, it is required to use a single threshold for all keywords to make “accept/reject” decisions. However, it is observed that lattice-based posterior probability scores are not comparable across keywords. Hence it is necessary to normalize detection scores so that a single threshold can be applied to all keywords.

Various score normalization techniques have been proposed, such as Keyword Specific Threshold (KST) [41, 131] and Sum-to-One (STO) normalizations [43]. The common aim of these techniques is to optimize the TWV metric based on its characteristics. These score normalization techniques are described as follows.

Keyword-specific threshold

The main idea of this method is to estimate a specific threshold for each keyword to optimize for TWV, then normalize detection scores so that all keywords share a global threshold. The two steps are described as follows.

For a keyword q_k , its optimum threshold θ_{q_k} can be estimated by a posterior ratio test, i.e.

$$\frac{\theta_{q_k}}{1 - \theta_{q_k}} = \frac{fc_{q_k}}{mc_{q_k}} \quad (2.17)$$

Figure 2.15: The keyword specific threshold as a function of number of keyword's occurrences

where fc_{q_k} is the cost that a false alarm of q_k is wrongly predicted as a hit, and mc_{q_k} is the cost that a hit of the keyword q_k is judged as a false alarm. According to the definition of TWV in Equation 2.6, we have: $fc_{q_k} = \frac{\beta}{T_{speech} - N_{q_k}}$ and $mc_{q_k} = \frac{1}{N_{q_k}}$; where $\beta = 999.9$, T_{speech} is the corpus duration, and N_{q_k} is the number of true occurrences of the keyword q_k in the test corpus. Therefore, the threshold is estimated as

$$\theta_{q_k} = \frac{\beta N_{q_k}}{T_{speech} - (\beta - 1)N_{q_k}} \quad (2.18)$$

Note that at testing stage, the number of references N_{q_k} is unknown. However, it can be fairly approximated as the sum of posterior probabilities of all detections, i.e. $N_{q_k} = \sum_{d_i} s(d_i)$ where d_i is a detection of q_k and $s(d_i)$ is the posterior probability of d_i . Figure 2.15 shows the θ_{q_k} as a function of N_{q_k} in case of $T = 36000$ (10 hours). It can be seen that θ_{q_k} is proportional to N_{q_k} ; hence, a rare keyword q_k , i.e. the keyword with low N_{q_k} , has a low threshold θ_{q_k} .

Since TWV metric requires a single threshold for all keywords, we need an extra step to normalize the detection scores so that a single threshold, denoted as θ_{global} , can be used. Any appropriate normalization formula should ensure that the normalized scores

together with the global threshold has identical “accept/reject” decisions to the original posterior probabilities. One appropriate formula, which is used in the Kaldi toolkit [103], to normalize the score of a detection d_i of the keyword q_k is as follow:

$$s'(d_i) = s(d_i) \frac{\log(\theta_{\text{global}})}{\log(\theta_{q_k})} \quad (2.19)$$

It can be seen that if $s(d_i) > \theta_{q_k}$ then $s'(d_i) > \theta_{\text{global}}$. In other words, Equation 2.19 satisfies the requirement that the new confidence scores have the same decisions as the original scores.

Sum-to-one normalization

As mentioned above, the cost of a miss, i.e. $\frac{1}{N_{q_k}}$ is inversely proportional to the number of true occurrence N_{q_k} , while the cost of a false alarm, i.e. $\frac{\beta}{T_{\text{speech}} - N_{q_k}}$, is almost independent to N_{q_k} since T_{speech} is much bigger than N_{q_k} . Therefore, missing a rare keyword is expensive and missing a frequent keyword is cheap. STO normalization technique normalizes scores to reduce the miss rate of rare keywords. For each keyword K , the normalized score of its detection d_i is obtained as follows:

$$s'(d_i) = \frac{s(d_i)}{\sum_j s(d_j)} \quad (2.20)$$

For a rare keyword, the denominator is the sum of a few values, so we can expect that at least one normalized score will be high. In the special case when there is only one hypothesized detection for a keyword, the normalized score of such detection will be 1.0. Thus this normalization helps to boost detection scores of rare keywords, and so reduces the miss rate of such keywords.

The advantage of the KST and STO methods is that they are unsupervised, which do not require any data to train the normalization. However, the KST and STO methods monotonically transform scores of hypothesized detections, and so are not able to re-rank these detections.

2.3.2.2 Score re-estimation

Score re-estimation techniques use not only lattice-based posterior probabilities but also additional sources of information to estimate new confidence scores. Therefore, these techniques can re-rank hypothesized detections. These techniques can be divided into

two groups which are discussed as follows.

a) Score re-estimation by acoustic similarity

One group of score re-estimation methods relies on the relationship between the hypothesized detections themselves. In these methods, the similarity or distance between the acoustic signals of the hypothesized detections is estimated, and then used to perform re-scoring. Such group of methods include PRF[47] and GBR [44–48].

In PRF [47], the hypothesized detections are sorted according to their scores. The top ranked hypothesized detections are assumed to be correct, while the bottom ranked hypothesized detections are assumed to be false alarms. The score of each hypothesized detection is re-estimated based on the similarity between the speech signal of the hypothesized detection and that of the top/bottom ranked detections. Specifically, if a hypothesized detection is acoustically closer to the top ranked detections, its score will be boosted, and vice versa. Note that, the similarity between two signals is usually computed using template matching, i.e. Dynamic Time Warping (DTW), in the acoustic feature space which is complementary with the model-based acoustic model scores.

In GBR [44–47], a graph is built from the hypothesized detections, with each node representing a detection and each edge representing the similarity between two hypothesized detections. The graph-based scores are first initialized (for example with their ASR posterior scores) and then refined iteratively by the graph random walk algorithm which makes use of both initial scores and the similarities between hypothesized detections. Ultimately, the GRW tends to give similar scores to two acoustically similar hypothesized detections, and vice versa.

One advantage of PRF and GBR is that they do not need any training data to train the re-scoring models. However, these two methods have their own limitations. The PRF method makes assumptions about the distribution of the hypothesized detections [48]. Specifically, it assumes that the top hypothesized detections are correct, which may not always be true. In GBR, a hypothesized detection is affected by other acoustically similar hypothesized detections, and hence the method might not work well if many false detections have large ASR scores. In conclusion, the effectiveness of PRF and GBR methods highly depends on the quality of initial search results which may be poor.

b) Score re-estimation by a prediction model

It has been shown that there are various factors, which might not be captured by the lattice-based posterior probability, can affect the reliability of hypothesized detections [50, 51, 128, 132]. For example, at the same probability scores, hypothesized detections of longer keywords are more likely to be a hit than those of shorter keywords [128]. Additionally, detections at the beginning or ending of an utterance are not as reliable as those in the middle [50, 51, 132]. It is desirable to incorporate such factors into a single confidence score.

The process of incorporating various factors to estimate a new score can be described as a score prediction model, i.e.

$$\text{Confidence_score} = f(c_1, c_2, \dots) \quad (2.21)$$

where $c_1, c_2, \dots$ are features contributing to the final confidence score such as posterior probability, query length, etc.; and f is a prediction function to combine such features into a final confidence score. Various score prediction models have been proposed. These methods differ in the features and model structures they use.

Various features have been used for score prediction, which can be categorized into the following groups.

- **Lexicon features:** Lexicon features include the number of words, graphemes, phonemes, syllables and vowels, consonants of the keywords [49–51]. These features describe lexical properties of the keyword in terms of keyword length and consonant-vowel structure. It has been observed that the more words, syllables, phonemes, or vowels a keyword has, the more confident its detections are.
- **Prosodic features:** These features provide information about the prosody of speech, i.e. max/min/mean pitch and energy, voicing percentage of each detection. Intuitively, abnormal pitch or energy values are often associated with false alarm detections.
- **Duration features:** These features include duration and speaking rate of the detection. Here, speaking rate is computed as the detection duration divided by the number of phonemes of the detection. The motivation for these features is that abnormal duration or speaking rate is highly associated with a false alarm [133].

- Utterance/conversation features: These features are computed using information of its entire utterance/conversation. The first feature is contextual consistence [52, 53, 134] which captures the consistency, i.e. the co-occurrence, of the detection with its neighboring words in the corresponding utterance. Another feature is word burst [135, 136] which is motivated by the observation that once a keyword has been uttered in a conversation, the probability of detecting that keyword in the rest of the conversation is higher than normal. In other words, detections which belong to the same conversation and close to each other will be boosted.

Modeling methods, i.e. the mapping functions f in Equation 2.21 that map features to the final confidence score, also contribute to the quality of the estimated score. Various modeling methods have been employed, including Logistic Regression [49, 69], Vector Quantizer [70], Multi-Layer Perceptron (MLP) [40, 50, 71], Support Vector Machines [40, 50, 72], and Conditional Random Field [54, 55]. Usually, these models are trained in a two-class classification setting, i.e. to predict whether an input detection is of true class or false class given a set of features mentioned above. Once the models are trained, the posterior probability of a detection belonging to the true class can be used as the new confidence score.

The major advantage of this rescoring strategy is that these models provide a flexible and uniform framework to integrate a vast number of features into a single confidence score. However, similar to other machine learning methods that rely on training data, this strategy may suffer from overfitting and may not generalize well to keywords that are not observed during training.

2.4 Summary

This chapter first introduced the STD query terms. Then, it described the block diagram of a typical STD system; followed by a description about various evaluation metrics for the STD task such as ATWV, MTWV. Subsequently, it reviewed techniques to deal with the problem of missing true occurrences of a keyword in a STD system, with the techniques grouped into two categories. The first category are subword-based approaches which cover both linguistic and data-driven subword units. The second category are

approximate matching techniques which aim to compensate for the insertion, deletion and substitution errors produced by ASR components. Finally, this chapter reviewed techniques to re-score hypothesized detections, which include score normalization and score re-estimation methods. Score normalization aims to normalize detection scores to maximize for the TWV metric, while score re-estimation methods utilize additional information to re-score and re-rank hypothesized detections. In the next chapter, we will present our proposed strategy to tackle the high miss rate problem in subword-based STD systems.

Chapter 3

Pruning Strategies for Partial Search in Spoken Term Detection

3.1 Introduction

The common approach for KWS is to first utilize a word-based ASR, i.e. a Large Vocabulary Continuous Speech Recognition (LVCSR), to transcribe audio files to word-based transcriptions or lattices then perform indexing/retrieval on such word-based representations. A spoken word cannot be recognized by the ASR if it is not present in the dictionary of the ASR, and hence cannot be retrieved by the word-based KWS. Subword-based approach has been used to address this particular OOV problem [41, 60, 120, 137]. Various types of subwords have been studied, including linguistic units such as phones [57, 58, 60], syllables [62, 63, 138], morphemes [65, 122], characters [62]; and data driven units such as word-fragments [117], multigrams [94], graphemes [107] and particles [101]. The subword-based approach not only addresses the OOV problem but also works well for IV keywords [48, 124].

In a subword-based KWS system, a standard retrieval technique is to convert the keyword into a subword sequence, then search the subword sequence in subword lattices [92, 103]. In the rest of this chapter, this technique is referred to as the full search approach. However, due to the inevitable high error rate of subword ASR systems, it is difficult to find the entire sequence in a lattice. Therefore, the full search approach often has high miss rates, especially for long keywords.

In this work, we propose a partial search approach to overcome the high miss rate of subword KWS systems. Specifically, overlapping sub-sequences, also called partial

sequences, are derived from the keyword’s subword sequence. These partial sequences are then used to search on subword lattices to produce intermediate result lists which are later merged to generate a final detection list. In order to reduce the miss rate, the proposed approach retains detections in the final list that contain only a portion of partial sequences. Although this deals with the issue of a high miss rate, it may result in many false alarms in the detection list. To deal with this issue, we proposed two pruning methods, namely Global Threshold Pruning (GTP) and Keyword-Specific Pruning (KSP). The first method empirically uses a global threshold for all keywords while the second gradually adjusts a threshold till a detection is found.

The proposed partial search approach is motivated by the success of the partial matching approach on the query-by-example [139, 140] and repeating sequence detection [141] tasks. However, in [139, 140], each partial sequence is a sequence of acoustic vectors rather than a sequence of subword units. In the context of KWS, the idea of deriving sub-sequences from the subword sequence of a keyword is similar to the ngram-based approach [93, 142, 143]. However, in [93, 142, 143], after merging search result lists of ngram sequences, only detections that appear in all result lists are retained. Hence the ngram-based approach may still lead to a high miss rate for long keywords.

The proposed partial search approach is also inspired by the idea of query expansion [3, 66, 102, 138]. Specifically, the seed keyword is expanded into another representation, i.e. partial sequences, then searched on the test corpus to reduce the miss rate. Additionally, by retaining detections that contain only some of partial sequences, the proposed approach implicitly compensates for insertion/deletion/substitution errors of the ASR system; hence leads to further miss rate reduction.

This chapter is organized as follows. In Section 3.2, we describe the proposed partial search approach. Section 3.3 presents the experimental setup. Section 3.4 shows the experimental results and analysis. Finally, Section 3.5 concludes our work.

3.2 The proposed partial search framework

The proposed partial search approach is shown in Figure 3.1. First, from the subword sequence of the keyword q , overlapping sub-sequences (or partial sequences) of length N

are derived. Then, these partial sequences are searched on the subword index to produce intermediate result lists L_1 , L_2 , etc. Next, these list L_i ($i = 1, 2, \dots$) are merged into a single list L_{merged} . Subsequently, pruning process is applied on L_{merged} to obtain L_{pruned} , followed by a scoring step to estimate the confidence score for each detection.

Figure 3.1: Proposed partial search framework. Several partial sequences are derived from the subword sequence of the keyword q . These partial sequences are then searched on subword lattices to produce search results L_1 , L_2 , L_3 . A_1 , A_2 , A_3 are audio files. Each item in L_i ($i = 1, 2, \dots$) has format (audio, start_time, end_time, score). For example, the item $r_{1,1} \in L_1$ (red color) means that the partial sequence $p_1 p_2 p_3$ appears in audio file A_1 at the start time 3.2 and end time 4.0, and it has a confidence score 0.5 which is produced by the WFST indexing/retrieval framework (see Section 2.1.4.2). Items $r_{1,1}$, $r_{2,2}$, $r_{3,1}$ form an overlapping detections set (in Section 3.2.3) which are merged into one single detection d_1 in L_{merged} .

3.2.1 Partial sequence generation

Suppose the keyword q is represented by a sequence of l subwords, i.e. $q = p_1p_2...p_l$. If $l < N$, then the entire keyword q is considered as a partial sequence. Otherwise, the keyword q is split into overlapping sub-sequences (or partial sequences), each having length N . Number of partial sequences should be $n = l - N + 1$. Figure 3.1 shows the results of partial sequence generation for $l = 5$ and $N = 3$. Specifically, the keyword's subword sequence consists of 5 subword units, which is then transformed into 3 partial sequences of length $N = 3$, i.e. $q = p_1p_2p_3p_4p_5 = p_1p_2p_3 \ p_2p_3p_4 \ p_3p_4p_5$.

The length N controls the trade-off between the miss rate and false alarm rate of the subword partial search systems. Small N implies a low miss rate at the cost of a high false alarm rate while large N indicates a high miss rate but low false alarm rate. It can be seen that the full search approach is a special case of the partial search where N is set to a very large value.

3.2.2 Partial sequence retrieval

In this work, we use WFST-based indexing/retrieval framework [97], which is widely used for the STD task [122, 144], for partial sequences retrieval. Specifically, a lattice is converted to a WFST that allows any sequence to be searched in the lattice. Each partial sequence is also converted to a linear transducer and then composed with the lattice WFST. This process produces result lists $L_1, L_2, ..., L_n$ for partial sequences of the keyword q . Each item in L_i ($i = 1, 2, ...$) has format (audio, start.time, end.time, score). For example, the item $r_{1,1} \in L_1$ (red color) means that the partial sequence $p_1p_2p_3$ appears in audio file $A1$ at the start time 3.2 and end time 4.0, and it has a confidence score 0.5 which is produced by the WFST indexing/retrieval framework (see Section 2.1.4.2)

3.2.3 Merging detections lists

At the end, we only need a single list of detections for the keyword q , hence we need to merge $L_1, L_2, ..., L_n$ into a single list L_{merged} . To do that, we find sets of overlapping items across $L_1, L_2, ..., L_n$, then merge each set into a single detection. Note that, overlapping

items, denoted by $M = \{r_{i,j} \mid r_{i,j} \subseteq L_i\}$, are these satisfy two of the following conditions: (1) they come from the same audio file; (2) their locations (time) are close enough to each other. The start time and end time of the merged detection are derived as the minimum start time and maximum end time of items in the set M respectively. For example in Figure 3.1, $r_{1,1}$, $r_{2,2}$, $r_{3,1}$ form a set of overlapping items and they are merged into a final detection d_1 in L_{merged} . The start time of d_1 is set by the minimum start time of $r_{1,1}$, $r_{2,2}$, $r_{3,1}$, which is 3.2; while the end time of d_1 is set by the maximum end time of $r_{1,1}$, $r_{2,2}$, $r_{3,1}$, which is 4.3.

The size of set M can be interpreted as the number of partial sequences included in d_1 . We denote such measurement as $c(d_1)$ which will be used for the pruning process in the next subsection. For example, in Figure 3.1, we have $c(d_1) = 3$, $c(d_2) = 2$ and $c(d_3) = 1$. Note that $c(d_i)$ might be smaller than the number of partial sequences, i.e. $c(d_i) < n$, since some partial sequences might be missing.

3.2.4 Pruning of merged results

The resulting list L_{merged} after the merging process in the previous subsection can be excessively lengthy with a lot of false alarms. Hence, pruning the merged list L_{merged} is necessary.

Ideally, a true detection of the keyword q should contain all n partial sequences. Hence, a straightforward pruning approach, which is introduced in [93, 143], is to discard a detection $d_i \in L_{merged}$ such that $c(d_i) < n$. However, due to a high error rate of a subword ASR, it might be difficult to detect all partial sequences of q , especially when q is a long keyword. Hence, the approach in [93, 143] generally results in a high miss rate.

Since the cost of missing a true detection is higher than the cost of producing a false alarm in the context of TWV metric [43], we encourage a low miss rate by allowing some partial sequences to be missed, i.e. $c(d_i) < n$. Specifically, we propose two pruning methods as follow:

Global Threshold Pruning (GTP): The merged detection $d_i \in L_{merged}$ is discarded if the ratio of number of partial sequences included in d_i over the total number of partial sequences of the keyword q is smaller than a fixed threshold T_{global} , i.e. $\frac{c(d_i)}{n} < T_{global}$. Note that the previous work [93, 143] is a special case of this method

where $T_{global} = 1$. Smaller T_{global} will encourage a low miss rate at the cost of a high false alarm rate. The optimum T_{global} is tuned in a development data. In Figure 3.1, the L_{pruned} list is obtained by applying the GTP technique with the global threshold $T_{global} = 0.5$. Since the fraction $\frac{c(d_i)}{n}$ are 1.0, 0.667 and 0.333 for d_1 , d_2 , d_3 respectively, the detection d_3 is discarded.

Keyword-Specific Pruning (KSP): Ideally, each keyword q has its own threshold T_q . Short keywords are easy to be detected hence they should have high thresholds. Long keywords, on the other hand, are difficult to be detected hence they should have low thresholds. This is the motivation of the KSP method. Specifically, for the keyword q , a keyword-specific pruning threshold is initialized as $T_q = 1$. Then, if there is no detection in L_{merged} to satisfy the threshold, i.e. $\frac{c(d_i)}{n} < T_q$ for all $d_i \in L_{merged}$, we gradually reduce T_q until finding at least one detection that passes the threshold. The formal description of this method is presented in Algorithm 1.

Algorithm 1 The Keyword Specific Pruning (KSP) algorithm.

Input:

n : Number of partial sequences of keyword q .

L_{merged} : List of merged detections of keyword q

Output: The list of detections after pruning, i.e. L_{pruned}

$L_{pruned} \leftarrow \{\}$

$T_q \leftarrow 1.0$

while $T_q > 0$ **do**

for $d_i \in L_{merged}$ **do**

if $\frac{c(d_i)}{n} \geq T_q$ **then**

$L_{pruned} \leftarrow L_{pruned} \cup d_i$

end if

end for

if L_{pruned} is empty **then**

$T_q \leftarrow T_q - 0.1$

else

break

end if

end while

3.2.5 Scoring the pruned list

One important quantification is to estimate the confidence score of a detection $d_i \in L_{pruned}$. Intuitively, d_i has a high confidence score in two conditions: (1) the overlapping detection set M has many elements, i.e. $c(d_i)$ is large; (2) each overlapping items $r_{k,j} \in M$ has a high confidence score. Therefore, the score of d_i can be estimated as follow:

$$score(d_i) = \left(\frac{1}{c(d_i)} \sum_{r_{k,j} \in M} score(r_{k,j}) \right)^{\frac{1}{1+\log(c(d_i))}} \quad (3.1)$$

where $score(r_{k,j})$ is the confidence score of $r_{k,j}$. The expression $\frac{1}{c(d_i)} \sum_{r_{k,j} \in M} score(r_{k,j})$ implies the above second condition while the exponent $\frac{1}{1+\log(c(d_i))}$ reflects the first condition. For example, score of the detection $d_1 \in L_{pruned}$ is calculated as

$$score(d_1) = \left(\frac{1}{3} (score(r_{1,1}) + score(r_{2,2}) + score(r_{3,1})) \right)^{\frac{1}{1+\log(3)}} = 0.54 \quad (3.2)$$

3.3 Experiment setup

3.3.1 NIST OpenKWS15 dataset

To support research in KWS, NIST has organized series of KWS evaluations, called NIST OpenKWS Evaluations. The experiments in this work are conducted on Swahili, which is the surprise language for the NIST OpenKWS15 Evaluation. Swahili is a language spoken in some East African countries such as Kenya, Tanzania, Democratic Republic of the Congo and Uganda. There are a large number of dialects spoken in the dataset which vary by regions. Also note that the language contains a large number of loan words from English, e.g. “lake”, “warrior”, “year”.

The released acoustic data includes training data to train the ASR systems, development data for parameter tuning of KWS, and the evaluation data to evaluate the KWS performance. Since NIST only released the reference for a part of the evaluation data, we only conducted experiments on this subset (called *evalpart1* set). The detailed information about the amount of data for training, development and testing is presented in Table 3.1.

Training data	Development data	Evaluation data (<i>evalpart1</i>)
40 hours	10 hours	15 hours

Table 3.1: Detailed acoustic data for OpenKWS15 Evaluation.

3.3.2 Keyword Search systems

We have built two subwords-based ASR systems, one morpheme system which uses morpheme dictionary and morpheme language model to generate morpheme-based output; and a grapheme system which uses grapheme-based dictionary and language model. It is worth noting that the organizer did not provide the pronunciation dictionary, hence in this work, all ASR dictionaries use graphemes as basic decoding units.

In order to build a subword based ASR system (morpheme or grapheme), we need (i) subword transcriptions to train a subword language model and (ii) a subword based lexicon that maps each subword to the basic decoding units, i.e. graphemes. This process is easy for the grapheme-based system since the dictionary is just an identical mapping from letters to themselves, and converting word transcriptions to letter transcriptions is trivial. However, it is less straightforward to obtain the morpheme-based dictionary and transcriptions since we need to have a word-to-morpheme conversion model. In this work, the conversion model is obtained using the Morfessor toolkit [123]. The toolkit uses word transcriptions to train the word-to-morpheme segmentation. Once the model is trained, it can be used to convert word transcriptions to morpheme transcriptions. From the morpheme transcriptions, the morpheme inventory is obtained and then the morpheme-based dictionary can be generated. The process to generate the morpheme-based transcriptions and dictionary is described in Figure 3.2.

From subword transcriptions, we have built the 4-gram language model using the SRILM toolkit [145]. To train the acoustic models for both systems, we used filter-bank features to train a DNN acoustic model. It consists of 5 hidden layers, and each layer has 2048 hidden units.

3.3.3 Keyword lists and evaluation metric

NIST released an evaluation keywords list which consists of 4454 keywords. We removed keywords that are not in *evalpart1*, which results in 1860 keywords. Since the number

Figure 3.2: The process to obtain morpheme-based dictionary and language model.

of OOV keywords in the keyword list is small, we manually added 933 OOV keywords to come up with total 2793 keywords. The number of IV keywords and OOV keywords are 1787 and 1006 respectively. Figure 3.3 presents the keywords distribution w.r.t keyword length. Besides evaluation dataset, NIST also provided 10 hours of development data for parameter tuning. In this work, we used the dataset to tune parameter N and the threshold T_{global} . Therefore, we also need to prepare the keyword list for the development data. To do so, we first used a keyword list shared among participants of OpenKWS15 competition, which consists of 2461 keywords. We discarded some “unreasonable” keywords, which were very short (less than 3 characters), then applied the same strategy for the evaluation keyword list. This process resulted in IV and OOV keyword lists which included 1972 and 1334 keywords respectively.

To evaluate KWS performance, both ATWV and MTWV are used in this work. Since the proposed approach aims to reduce the miss rate of subword systems, we also report the Supremum Term Weighted Value (STWV) metric which is related to miss rate of a KWS system. In particular, the STWV is obtained by setting scores of all false alarms to 0, which is equivalent to removing all false alarms out of the hypothesized detections lists. Thus, the STWV metric can be interpreted as the recall, which is inversely proportional to the miss rate, of a KWS system. Note that, as mentioned in Section 2.1.6, the ATWV and MTWV scores are in the range $(-\infty, 1]$, where the score of 1 indicates a perfect STD system. Also note that, the score of 0.3 was set as the benchmark in NIST OpenKWS 2013 [144].

Figure 3.3: Keyword distribution w.r.t keyword length of the *IV* and *OOV* keyword lists of Swahili.

3.3.4 Baseline systems

The first baseline system is the full search approach [97] which is described in Figure 3.4. Specifically, at the indexing stage (the right branch in Figure 3.4), a subword lattice is indexed by the WFST technique (see Section 2.1.4.2) which results in the index WFST I . At the search stage, the keyword is first converted to a subword sequence, e.g. “organization = or ga ni za tion”, then the subword sequence is transformed to the WFST format which is denoted as K in the left branch in Figure 3.4. Finally, the search result is obtained by the composition operator between the keyword K and the index I , which can be described as

$$\text{Search result} = K \circ I. \quad (3.3)$$

where $\circ$ denotes the composition operator.

Figure 3.4: The baseline full search approach.

Another baseline is the query expansion technique [112, 128] which was presented in Section 2.2.2.2. Specifically, for the grapheme-based system, the retrieval process can be described as follows

$$\text{Search result} = Nbest(K \circ C) \circ I. \quad (3.4)$$

where C is the confusion model that encodes the insertion/deletion/substitution costs between graphemes, $Nbest$ denotes the degree of expansion. Such model can be constructed using the ASR training data by following steps:

- (i) First generate hypothesized transcriptions of training data using the grapheme-based ASR.
- (ii) Then generate the reference grapheme transcriptions of the training data using word-to-grapheme conversion.
- (iii) Then for each utterance, align the decoding sequence with the corresponding reference sequence to obtain the insertion/deletion/substitution errors information.
- (iv) Finally gather the errors on the entire data set to build the confusion model.

For the morpheme-based system, since the number of morpheme units are over 3000, it is difficult to estimate the confusion matrix C . Therefore, to perform query expansion, we used the technique in [3, 102] which transforms the subword sequence of the keyword into other subword sequences using grapheme-to-grapheme confusion model. This process can be described as

$$\text{Search result} = Nbest(K \circ L \circ C \circ L^{-1}) \circ I. \quad (3.5)$$

where L denotes the lexicon WFST that maps morphemes to their grapheme sequences, and L^{-1} denotes the inverse mapping of L . Basically, $K \circ L$ converts the keyword K into a grapheme sequence, then $K \circ L \circ C$ expands the keyword's grapheme sequence into other similar grapheme sequences (or proxy sequences) that have small edit distance to the keyword sequence. Finally, the composition $K \circ L \circ C \circ L^{-1}$ is used to map back these proxy sequences into morpheme representations so that they can be searched on the morpheme index.

3.4 Experiment results and analysis

3.4.1 Results of baseline approaches

Figure 3.5: The MTWV of baseline systems on development data while varying the number of expansions per keyword, i.e. the $Nbest$ value in Equation 3.4. The left and right figures correspond to the grapheme and morpheme systems respectively.

In this section, we report the experimental results of baseline techniques which were presented in Section 3.3.4, i.e. full search and query expansion. The results are shown in Figure 3.5. Note that for query expansion, its performance depends on the degree of expansion, i.e. N_{best} . It is observed that the query expansion technique is effective for the grapheme system and OOV keywords. Specifically, for the grapheme system, using OOV and IV keywords, we achieve 15.3% and 3% relative MTWV improvement with query expansion at $N_{best} = 80$ and $N_{best} = 60$ respectively.

3.4.2 Results of proposed approaches

The performance of partial search with GTP and KSP methods is affected by the partial length N . Additionally, the performance of the GTP method is also affected by the global threshold T_{global} . Figures 3.6 presents the MTWV performance of GTP and KSP methods while varying the values of N and T_{global} on the development set. Note that the KSP technique is not affected by T_{global} , while the baseline full search is independent to both N and T_{global} . The optimum values of N and T_{global} are summarized in Table 3.2.

	Grapheme system		Morpheme system	
	IV keywords	OOV keywords	IV keywords	OOV keywords
N	6	6	3	2
T_{global}	0.8	0.5	0.7	0.6

Table 3.2: The optimum values of the length of a partial sequence N and the global threshold T_{global} that are obtained from the development data.

Table 3.3 shows the KWS performance of the partial search on *evalpart1* test set using two pruning methods, i.e. GTP and KSP for both *IV* and *OOV* keyword lists. We report the performance of the two baselines (see Section 3.3.4) full search and query expansion, as well as the ngram inverted index (NII) [93, 143], for comparison. From Table 3.3, we have several observations: (1) The KSP technique is always better than the GTP technique. This observation confirms that having different thresholds for different keywords results in better performance than applying a global pruning threshold for all keywords. (2) The partial search with KSP is effective across all conditions, especially for the grapheme system. For example, on *IV* keyword list of the grapheme system, partial search with KSP outperforms the full search baseline by 4.1% and 2.2% absolute MTWV

Figure 3.6: The MTWV on the development data while varying the values of parameters N and T_{global} . The left and right figures correspond to IV and OOV keywords respectively, while the top and bottom figures correspond to grapheme and morpheme systems respectively.

and STWV respectively. (3) On the *OOV* keyword list, the performance improvement is more significant than on the *IV* keyword lists. For example, on the *OOV* keyword list of the grapheme system, the partial search with KSP outperforms the full search by 6.9% and 7.4% absolute MTWV and STWV respectively. This might be because that OOV keywords are more difficult to be detected, i.e. the miss rate of OOV keywords is much higher than *IV* keywords. Moreover, OOV keywords appear rarely, hence missing these keywords is very expensive. Therefore, the partial search benefits us more in the case of *OOV* keywords than the *IV* keywords.

We then analyze the performance improvement of the proposed partial search with KSP over the baseline full search approach for different keyword lengths. Figure 3.7 shows the comparison results. It can be seen that for all conditions, the partial search with KSP

Keyword lists	Systems	Approaches	Swahili		
			ATWV	MTWV	STWV
IV	Grapheme	Full search [97] (baseline)	0.2678	0.2759	0.5140
		NII [93, 143] (baseline)	0.2761	0.3007	0.5149
		Query expansion [112, 128] (baseline)	0.2847	0.2931	0.5535
		Partial search	GTP	0.2746	0.2998
			KSP	0.2911	0.3169
	Morpheme	Full search [97] (baseline)	0.4165	0.4187	0.6614
		NII [93, 143] (baseline)	0.4185	0.4205	0.6623
		Query expansion [112, 128] (baseline)	0.4165	0.4187	0.6614
		Partial search	GTP	0.4193	0.6631
			KSP	0.4271	0.6763
OOV	Grapheme	Full search [97] (baseline)	0.2586	0.2629	0.3886
		NII [93, 143] (baseline)	0.2660	0.2734	0.3917
		Query expansion [112, 128] (baseline)	0.2881	0.2920	0.4280
		Partial search	GTP	0.2787	0.4156
			KSP	0.3227	0.3315
	Morpheme	Full search [97] (baseline)	0.3444	0.3569	0.4970
		NII [93, 143] (baseline)	0.3596	0.3607	0.5166
		Query expansion [112, 128] (baseline)	0.3745	0.3797	0.5425
		Partial search	GTP	0.3633	0.5222
			KSP	0.4035	0.4048

Table 3.3: The KWS results of proposed partial search approach as compared to its baselines on *evalpart1* test set.

Figure 3.7: The ATWV of partial search with KSP and full search on the *IV* and *OOV* keyword lists with different keyword length. The left figure is for the morpheme system while the right figure is for the grapheme system.

is more helpful for long keywords. For example, on *OOV* keyword list, keywords having more than 17 graphemes achieve over 40% absolute improvement with the proposed approach. This is understandable since long keywords are difficult to be detected with the full search baseline.

The proposed partial search with KSP not only outperforms the query expansion technique but also offers several advantages. Firstly, the proposed technique does not require any training model, e.g. confusion model, as in the query expansion. Secondly, the proposed approach is faster than the query expansion as shown in Figure 3.8. This advantage is essential for real-time applications.

Figure 3.8: The average processing time per keyword of the proposed partial search approach as compared to the query expansion and full search approach. The left figure is for the grapheme system while the right figure is for the morpheme system. Experiments are run on a workstation with 256 GB memory, 20 cores CPU.

3.5 Summary

We proposed a partial search approach to tackle the high miss rate problem of subword STD systems. The proposed approach first converts the keyword’s subword sequence into several overlapping sub-sequences (or partial sequences). This helps to reduce the miss rate since matching sub-sequences are easier than matching the entire long sequence. The proposed approach further reduces the miss rate by allowing some partial sequences to be missed from the index. To reduce the false alarm rate, we examined two pruning

methods, i.e. GTP and KSP. The first method uses the same pruning threshold for all keywords while the second method adaptively adjusts a specific threshold for each keyword until finding at least one detection. Experimental results demonstrated that the partial search with KSP outperforms its baselines, i.e. the full search, ngram inverted index and query expansion techniques, on the NIST OpenKWS15 evaluation dataset. Experimental analysis showed that the proposed approach is more helpful for long keywords. This is because the long keywords are difficult to be detected, hence they normally have high miss rate. Additionally, the proposed approach also benefits more for the *OOV* keywords than for the *IV* keywords. This is because *OOV* keywords appear rarely, hence missing these keywords is very expensive.

Chapter 4

Re-ranking Spoken Term Detection with Acoustic Exemplars of Keywords

4.1 Introduction

As mentioned in Section 2.1.2, many STD systems rely on ASR to transcribe spoken utterances into lattices, then indexing/retrieval techniques are applied on those lattices to produce a list of hypothesized detections for each keyword. As a result, the performance of these STD systems heavily depends on the ASR accuracy, which in turn, depends on its acoustic and language models. In common scenarios, the test database might be spoken by many speakers, under different acoustic environments. It is thus difficult to train an acoustic model that is well matched to all conditions. As a result, the detection scores derived from ASR lattices, i.e. the lattice-based posterior probability [36, 37] might not be reliably estimated. Therefore, it is desirable to directly utilize the information in acoustic signal to re-score hypothesized detections.

Several re-scoring techniques, which make use of acoustic signal, has been proposed such as Pseudo Relevance Feedback (PRF) [47] and Graph-Based Re-ranking (GBR) [44–48]. Those techniques rely on the relationship between the hypothesized detections themselves to perform re-scoring. In the PRF method, top detections are assumed to be relevant, while bottom detections are assumed to be irrelevant to the keyword. Then, for each detection, its acoustic similarity scores to pseudo relevant and irrelevant detections are used to perform re-scoring. The GBR method, on the other hand, builds a fully

connected graph, where each node represents a detection, then re-scores detections using a graph-based algorithm such as random walk [146].

Although they are shown to be effective when applying to the KWS task, both have their own limitations. The PRF method makes assumptions about the distribution of the hypothesized detections [48]. Specifically, it assumes that the top ranked detections are correct and bottom ranked detections are false alarms, which may not be always true. In GBR, a hypothesized detection is affected by other acoustically similar hypothesized detections, hence, the method might not work well if many false detections have large scores. In conclusion, the effectiveness of PRF and GBR methods highly depends on the quality of initial search results, which may be poor.

To address the limitation of PRF and GBR, in this work we propose a re-scoring framework that makes use of acoustic exemplars of the keyword to be searched. An acoustic exemplar is an instance of the keyword found in an annotated speech corpus, e.g. the training data of ASR systems. The main idea is that if a hypothesized detection is acoustically more similar to the exemplars, it is more likely to be a correct detection, and its score should be boosted. Since keyword exemplars are true instances of the keyword, using those exemplars to re-score hypothesized detections voids making any assumptions as in PRF and GBR methods. We call this proposed framework **Re-ranking by Acoustic Exemplars (RAE)**.

In this chapter, we first introduce the proposed framework in Section 4.2. Then, Section 4.3 presents the process to generate acoustic keyword exemplars. Section 4.4 describes approaches to estimate the acoustic similarity scores between detections and keyword exemplars. Given the acoustic similarity, Section 4.5 demonstrates two re-ranking methods. Experimental setup is described in Section 4.6, then results are reported in Section 4.7. Finally, we conclude in Section 4.8.

4.2 Re-ranking Spoken Term Detection with acoustic keyword exemplars

Given a keyword q , a baseline KWS system searches in the word/subword lattices generated by an ASR system to produce a list of hypothesized detections, each with start/end

Figure 4.1: Proposed re-ranking by acoustic exemplars (RAE) framework.

times and the estimated confidence score, i.e. the lattice-based posterior probability. The proposed RAE framework to re-rank the list of hypothesized detections is shown in Figure 4.1. It consists of three steps: keyword exemplar generation, acoustic similarity estimation and re-ranking.

The first step is to generate a list of exemplars for the keyword q from a speech corpus, i.e. the training data of the ASR system. If the keyword has instances in the corpus, we simply extract the speech signals of the instances. Otherwise, we can generate the exemplars by concatenating instances of smaller units such as words or subwords.

The second step is to compute the acoustic similarity between the speech signals of detections and keyword exemplars. DTW is used to compute the similarity scores as the signals might be of different lengths. The similarity measures can be improved by incorporating extra information.

In the third step, given the acoustic similarity scores, two methods, namely re-ranking by score fusion and re-ranking by similarity graph, are applied to re-rank the hypothesized detections. The first method estimates the average similarity between each hypothesized detection and all keywords exemplars, then fuses such measurement with the original detection score. The graph-based method, on the other hand, builds a fully connected

graph, where each node represents a hypothesized detection or an exemplar, then applies the random walk algorithm to estimate a new set of detection scores. In following sections, we will describe the above steps.

4.3 Keyword exemplar generation

Let's consider a keyword q which consists of $nw(q)$ words, i.e. $q = W_1W_2...W_{nw(q)}$. To extract exemplars for this keyword in the training data, we first generate time boundaries for each word or subword in the training data using forced alignment. Specifically, given a training audio and the corresponding text transcription (in word/subword format), the Viterbi [81, 147] algorithm is used to align the transcription to the audio file which results in each word/subword time-aligned in the transcription. Then, depending on whether the entire keyword appears in the training data or not, i.e. whether q is a *seen* or *unseen* keyword, the keyword exemplar generation is performed through two scenarios as will be described in Section 4.3.1 and 4.3.2.

4.3.1 Exemplar generation for a seen keyword

If the entire sequence $W_1W_2...W_{nw(q)}$ appears in the training data, then we extract the whole speech segments at the found locations as exemplars for q . For each exemplar, if there is a long silence between consecutive words W_i and W_{i+1} , i.e. the gap between the end time of W_i and start time of W_{i+1} is larger than 0.5 second, we remove such silence out of the exemplar.

4.3.2 Exemplar generation for an unseen keyword

In case the sequence $W_1W_2...W_{nw(q)}$ cannot be found in the training data, i.e. q is an unseen keyword, we extract samples for constituent subwords of the keyword q , then concatenate these samples to generate exemplars for the keyword q . The detailed procedure is as follow.

4.3.2.1 Subword samples extraction

For each word W_i , if it does not appear in training data, we split it into subword representations¹, otherwise we keep W_i unchanged. This strategy favors the use of longer units, which generally produces better concatenated speech [148, 149]. The conversion process results in a new representation of keyword q , i.e. $q = U_1 U_2 \dots U_l$ where U_i are either words or subwords. For simplicity, in the rest of this chapter we treat all U_i as subwords. Then, given the time boundaries of subwords in the training data, we extract all instances for each subword U_i . For each instance, we store its feature vectors, e.g. MFCC or bottleneck features, and the meta-data information such as speaker ID, gender, recording channel, and left/right phonetic contexts. This information can be used to improve the quality of the generated exemplars.

4.3.2.2 Subword samples concatenation

Given samples of subword U_1, U_2 , etc., we concatenate them to form samples of entire keyword q . Let suppose that each subword U_i has $\|U_i\|$ samples, which are denoted as $h_i^{(1)}, h_i^{(2)}$, etc. The number of concatenated instances will be $\|U_1\| \times \|U_2\| \times \dots \times \|U_l\|$, which is huge. In this work, we only select a fixed K concatenated instances as the exemplars for q . Since each subword sample has its own acoustic properties, e.g. speaker, gender, etc., arbitrarily concatenating them might result in poor concatenated exemplars.

In this work, we adopt the idea of unit selection [150] in Text-to-speech research, i.e. our aim is to generate smooth and natural concatenated exemplars. In particular, a transition network is first constructed to represent concatenation space; then two cost functions, namely target and transition (or concatenation) costs, are defined and incorporated into the network; finally a dynamic programming algorithm [81, 147] is applied to find the K -best sequences through the network. This process is shown in Figure 4.2 and the detailed procedure will be described as follows.

Constructing the transition network. The first step of the keyword exemplar concatenation procedure is to build a transition network. In this network, the i^{th} ($i = 1 \dots l$) layer represents the subword unit U_i , and the j^{th} ($j = 1 \dots \|U_i\|$) state in this layer corresponds to a sample $h_i^{(j)}$ of U_i . Without confusion, a sample and its corresponding

¹In this work, we use morpheme as the subword unit

Figure 4.2: Acoustic exemplar generation for an unseen keyword $q = nafasi = na fa si$. Rounded blocks of the same column are instances of the same subword, each might have different phonetic context, speaker, gender, channel information. Such information will be incorporated into the target and concatenation cost functions to improve the generated exemplars. Using these cost functions, a dynamic programming algorithm estimates K-best concatenation sequences. The best sequence goes through the shade blocks in this figure.

network state will be used interchangeably in the rest of this section. Additionally, without loss of generality, we omit the superscript in $h_i^{(j)}$ and use h_i to denote a sample of U_i .

Defining two cost functions. Once the transition network is constructed, two cost functions, namely target and transition costs, are defined to reflect the mismatches during concatenation of subword samples.

- For each state h_i , we define a target cost $P_o(h_i, U_i)$ which reflects the mismatch of the sample h_i to the target unit U_i . Mismatches occur when the left/right phonetic contexts of the sample h_i differ from the corresponding left/right contexts of the subword unit U_i .
- For each concatenation between two samples h_{i-1} and h_i , we define a transition

cost $P_t(h_{i-1}, h_i)$ which reflects the acoustic mismatch between the two samples. Note that there is a key differences between this work and the unit selection: in unit selection, the speech corpus to generate the inventory is spoken by a single speaker while in our case, the speech corpus is spoken by multiple speakers of different genders in different channels. As a result, the transition cost is re-defined to take into consideration the phonetic contexts, speaker, gender and audio channel consistency to optimize the quality of concatenated exemplars. For example, a mismatch between phonetic contexts of instances of two consecutive subwords should be penalized since it might lead to unexpected coarticulation effect. Additionally, male and female speech has shown to be different in term of fundamental frequency [151], intensity [152], etc.; hence it is discouraged to concatenate two instances from opposite genders. At the end, the transition cost $P_t(h_{i-1}, h_i)$ is estimated as the sum of all mismatches between h_{i-1} and h_i , where each mismatch has its own value as will be mentioned in Section 4.6.2.

Finding K-Best concatenated keyword exemplars. Finding K-best concatenated exemplars equivalents to finding K-best paths on the transition network, which can be solved efficiently using the dynamic programming. Specifically, we first define a cost $P(\pi)$ for a path $\pi = h_1 h_2 \dots h_i$ as follows,

$$P(\pi) = \sum_{j=1}^i [P_o(h_j, U_j) + P_t(h_{j-1}, h_j)] \quad (4.1)$$

We then define $\phi(h_i)$ as a set of K-best scores, i.e. lowest costs, for all paths from the beginning of the network until the state h_i , i.e.

$$\phi(h_i) = \underset{\pi \text{ ends at } h_i}{KBest} [P(\pi)] \quad (4.2)$$

where $KBest$ is the function to find the K-best values in a set, e.g. sort the set then select top K items. Let $\phi_k(h_i)$, where $k = 1..K$, be elements of the set $\phi(h_i)$. We then have

$$\phi(h_{i+1}) = \underset{h_i \in U_i \text{ and } k=1..K}{KBest} [\phi_k(h_i) + P_t(h_i, h_{i+1})] + P_o(h_{i+1}, U_{i+1}) \quad (4.3)$$

To trace back the desired paths that result in K-best concatenation costs $\phi(h_{i+1})$, we need to keep the argument of the $KBest$ function, denoted as $argKBest$, in Equation

(4.3). We store such information in an array $\psi(h_{i+1})$. In summary, the procedure to find K-best paths through the transition network can be described as follows.

1) Initialization:

$$\begin{aligned}\phi(h_1) &= P_o(h_1, U_1) \\ \psi(h_1) &= \{\};\end{aligned}\tag{4.4}$$

2) Recursion:

$$\begin{aligned}\phi(h_i) &= \underset{h_{i-1} \in U_{i-1} \text{ and } k=1..K}{KBest} [\phi_k(h_{i-1}) + P_t(h_{i-1}, h_i)] + P_o(h_i, U_i) \\ \psi(h_i) &= \underset{h_{i-1} \in U_{i-1} \text{ and } k=1..K}{argKBest} [\phi_k(h_{i-1}) + P_t(h_{i-1}, h_i)]\end{aligned}\tag{4.5}$$

2) Termination:

$$\begin{aligned}\phi^* &= \underset{h_l \in U_l \text{ and } k=1..K}{KBest} [\phi_k(h_l)] \\ \psi^* &= \underset{h_l \in U_l \text{ and } k=1..K}{argKBest} [\phi_k(h_l)]\end{aligned}\tag{4.6}$$

Note that the above algorithm is similar to the Viterbi [81, 147] algorithm. The major difference is that instead of using *min* operator to find the best path ending at a state, we use the *KBest* function to achieve the K-best paths ending at that state.

We now consider the computational complexity of the proposed subword sample concatenation. Suppose each subword U_i ($i = 1..l$) has A samples in average. In other words, the network in Figure 4.2 has l columns, each has A nodes. For each node, we need to calculate the expression $[\phi_k(h_i) + P_t(h_i, h_{i+1})]$ in Equation (4.3) $K \times A$ times, then select K-best values from $K \times A$ values, which has the complexity $O(K \times A)$. Hence the complexity for each node is $O(K \times A)$. Since we have $l \times A$ nodes, the total complexity should be $O(l \times K \times A^2)$.

4.4 Acoustic similarity estimation

Once keyword exemplars are generated, the next step is to estimate the acoustic similarity between pairs of speech signals of hypothesized detections and keyword exemplars. To achieve this goal, we first estimate the Dynamic Time Warping (DTW) distance between two speech signals in some feature spaces such as MFCC or bottleneck features. Since the DTW distance might not be compatible with the ASR score, some score calibration techniques are required to convert DTW distance to the similarity score. The DTW and score calibration techniques are described in Section 4.4.1 and 4.4.2.

4.4.1 Dynamic Time Warping

Suppose that we need to estimate the acoustic distance between two speech segments X and Y , which is denoted as $D(X, Y)$. Generally, those segments are presented by sequences of feature vectors, i.e. $X = x_1, x_2, \dots, x_{t_X}$ and $Y = y_1, y_2, \dots, y_{t_Y}$ where x_i and y_j are vectors of features such as MFCCs or bottle-neck features. To estimate the acoustic distance between X and Y , the DTW method constructs a t_X -by- t_Y matrix, called the local distance matrix $\mathcal{L}$. The (i, j) -th element of the local distance matrix represents the distance between x_i and y_j , which is estimated by the Cosine distance, i.e. $\mathcal{L}(i, j) = \frac{x_i \cdot y_j}{\|x_i\| \|y_j\|}$. Two additional matrices, A and L are defined to store the accumulated distance and the length of the alignment between X and Y . These matrices are filled with the dynamic programming algorithm as follow:

$$\Omega = \{(i-1, j), (i-1, j-1), (i, j-1)\} \quad (4.7)$$

$$(r^*, s^*) = \underset{(r,s) \in \Omega}{\operatorname{argmin}} \frac{A(r, s) + \mathcal{L}(i, j)}{L(r, s) + 1} \quad (4.8)$$

$$\begin{cases} A(i, j) = A(r^*, s^*) + \mathcal{L}(i, j) \\ L(i, j) = L(r^*, s^*) + 1 \end{cases} \quad (4.9)$$

In above equations, r and s represent the row and column indexes respectively; and the point (r, s) can be one of three points in the set Ω . The acoustic distance between X and Y is defined as the average distance along the warping path of X and Y , i.e. $D(X, Y) = \frac{A(t_X, t_Y)}{L(t_X, t_Y)}$.

4.4.2 Score calibration techniques

Once the DTW distance between X and Y is computed, it can be converted to a similarity score using the *min-max* normalization, logistic regression or using the neural network classifier as described below.

4.4.2.1 Min-max normalization

The distances from DTW is converted to similarity score by using *min-max* normalization [153], i.e.

$$S(X, Y) = 1 - \frac{D_{max} - D(X, Y)}{D_{max} - D_{min}} \quad (4.10)$$

where $D(X, Y)$ denotes the DTW distance between X and Y , D_{max} and D_{min} are the maximum and minimum DTW scores obtained from all pairs of speech signals for a particular keyword. Equation 4.11 transforms the distance score $D(X, Y)$ into a similarity score $S(X, Y)$ in the range $[0, 1]$. Particularly, the minimum DTW score will be mapped to the similarity score 1 while the maximum DTW score will be mapped to the similarity score 0. This normalization is necessary to make the similarity scores comparable across keywords, as well as compatible with the original detection scores.

4.4.2.2 Logistic regression calibration

We note that the DTW algorithm, which averages the per-frame distances along the alignment path of two speech signals, produces scores that are more like log-likelihoods. Hence, it might not be sufficient to just normalize DTW score in to range of $[0, 1]$ as in *min - max* normalization. It would be reasonable to calibrate the DTW distance using logistic regression which is widely used to calibrate log likelihood ratio in speaker verification [154, 155]. Specifically, the calibrated score, i.e. the acoustic similarity $S(X, Y)$, is obtained from the DTW distance as

$$S(X, Y) = \text{sigmoid}(a * DTW(X, Y) + b) \quad (4.11)$$

where *sigmoid* is the logistic sigmoid function, a and b are parameters trained on development data using positive and negative samples of the same keyword.

4.4.2.3 Neural network classifier

The DTW distance is an average of the per-frame distances along the alignment path between two speech segments. This measure might not fully capture the similarity between the two segments. Other factors such as some local similarity measures, which describe the partial matching of the two speech segments, and the length ratio between these segments, might provide complementary information to estimate the similarity. Therefore, a more general approach is to use a classifier, i.e. a Neural Network (NN), to incorporate various sources of information to predict the similarity score. The network is trained for a two-class classification setup, where class 1 means the two speech segments contain the same word sequence and class 0 means the opposite. In this work, the NN

has 2 hidden layers, each contains 50 nodes. The features used to train the NN include: the DTW distance between two speech signals; distances from others variants of DTW such as subsequence DTW [156] and partial DTW [139] since they are informative and complement the standard DTW; the length ratio of the two signals since abnormal length ratio indicates that the two signals are less likely to have the same content. At test time, the similarity score of two input signals is estimated as the posterior probability of the class 1.

4.5 Re-ranking methods

Given the acoustic similarity estimated in Section 4.4, two re-ranking methods are applied, namely RAE by score fusion (denoted as RAE-SF) and RAE by similarity graph (denoted as RAE-SG), to refine detection scores. The two methods are introduced in Section 4.5.1 and Section 4.5.2.

4.5.1 RAE by score fusion

For a keyword q , let

- $d_1, d_2, \dots, d_N$ be the hypothesized detections of q
- $e_1, e_2, \dots, e_M$ be the exemplars of q .

We first estimate the average acoustic similarity between d_i and all exemplars as follow

$$S(d_i) = \frac{1}{M} \sum_{j=1}^M S(d_i, e_j) \quad (4.12)$$

Then, a new confidence score for d_i is obtained by fusing $S(d_i)$ with the original score produced by the ASR-based KWS system:

$$C'(d_i) = C_{asr}(d_i)^\delta S(d_i)^{1-\delta} \quad (4.13)$$

where $C_{asr}(d_i)$ is the ASR-based score of d_i , δ is a factor that controls the weights of the two types of scores. The exponential interpolation is used in Equation 4.13 instead of additive interpolation because we observed that for some keywords detection scores are

in a much smaller range than the acoustic similarity scores; and using the exponential interpolation makes detection scores and similarity scores become more comparable than using additive interpolation. The optimal value of δ will be tuned using the development data.

The proposed score fusion method, as demonstrated in Equation (4.12-4.13), is related to the PRF method [47]. In the PRF method, it is assumed that hypothesized detections with high ASR scores (or the top sorted detections) are correct, which form a pseudo positive set Ψ . On the other hand, hypothesized detections with low scores (or bottom detections) are assumed to be false alarms and form a pseudo negative set Φ . Then, for each detection d_i , a similarity score between d_i and the top/bottom ranked detections is estimated by the following equation:

$$S_{\text{PRF}}(d_i) = \frac{1}{|\Psi|} \sum_{d_j \in \Psi} S(d_i, d_j) - \frac{1}{|\Phi|} \sum_{d_k \in \Phi} S(d_i, d_k) \quad (4.14)$$

where $S(.,.)$ denotes the acoustic similarity, $|\Psi|$ and $|\Phi|$ are the sizes of the two sets Ψ and Φ respectively. Finally, the S_{PRF} is normalized [153] then fused with the original ASR score, which is similar to Equation 4.13. The PRF score of the detection d_i will be high if d_i is acoustically similar to top ranked detections but different from the bottom ranked detections, and vice versa.

Although PRF and RAE-SF exploit similar ideas, i.e. to fuse the lattice posterior probability and acoustic similarity scores, they are different in the way they estimate the acoustic similarity scores. Specifically, PRF estimates the acoustic similarity between each detection and other detections, i.e. the top and bottom ranked detections, with the assumption that top ranked detections are correct and bottom ranked detections are false alarms. Since this assumption is not always true, the PRF method might not work well in all cases. RAE-SF, on the other hand, estimates the acoustic similarity between each detection and keyword exemplars without using other detections. Since the keyword exemplars are extracted from a labeled dataset, i.e. the training data of ASR systems, they are true instances of the keyword. As a result, we do not need to make any assumption as in the PRF method.

4.5.2 RAE by similarity graph

Although the RAE by score fusion method is straightforward, it does not make full use of the similarity relationship between all hypothesized detections and keyword exemplars. In this section, we propose a similarity graph based re-ranking method. The method is extended from the previous GBR method [44–47, 157]. Note that the idea of GBR is to exploit the relationship between hypothesized detections themselves to perform re-score and re-rank. Specifically, a fully connected graph is built where each node represents a detection and each edge represents the similarity between two detections. Then graph-based algorithms, e.g. Graph Random Walk, ensure that two nodes with high similarity will have similar graph-based scores. In other words, a detection which is similar to other high-score detections is boosted.

Our proposed method extends the GBR by introducing keyword exemplars. The proposed method is demonstrated in Figure 4.3(b), while the original GBR method is also shown in Figure 4.3(a). The major difference is that true keyword exemplars, i.e. $e_1, e_2, \dots$, are introduced to the graph.

Figure 4.3: The conventional graph-based re-ranking (a) and the proposed RAE by similarity graph (b). Hypothesized detections d_1, d_2 , are in black color while keyword exemplars $e_1, e_2, \dots$ are in red color. The conventional graph-based re-ranking only uses detections to perform re-ranking while the proposed RAE-SG uses both detections and keyword exemplars for re-ranking.

The first step of the proposed method is to construct a directed graph to represent the similarity relationship of hypothesized detections and exemplars of the keyword q . Each node represents either a hypothesized detection or an exemplar, hence, there are totally $N+M$ nodes. A weighted edge between two nodes represents the similarity between these two nodes. In the beginning, the graph is fully connected and undirected. To improve efficiency, we prune the graph by using the κ -nearest neighbor criterion. Specifically, two nodes are connected to each other if one of them is among κ highest similarity scores of another. In preliminary experiments, we observed that $\kappa = 5$ provides the best results, hence in this work, we manually fix $\kappa = 5$ for all experiments.

In order to apply the graph random walk (GRW), which will be described later, to estimate a new set of detection scores on the constructed graph, we further normalize the edge weights of the graph, i.e. the similarity scores, as follow:

$$S'(\mathcal{N}_i, \mathcal{N}_j) = \frac{S(\mathcal{N}_i, \mathcal{N}_j)}{\sum_{\mathcal{N}_k \in H_i} S(\mathcal{N}_i, \mathcal{N}_k)} \quad (4.15)$$

where H_i represents the set of nodes which are the nearest neighbors of $\mathcal{N}_i$. Note that $\sum_{\mathcal{N}_k} S'(\mathcal{N}_i, \mathcal{N}_k) = 1$, hence the normalization in Equation (4.15) converts the similarity scores to the transition probabilities so that the random walk algorithm can be applied on the graph. Also note that after normalization, the constructed graph becomes directed since $S'(\mathcal{N}_i, \mathcal{N}_j)$ generally differs from $S'(\mathcal{N}_j, \mathcal{N}_i)$.

Each node $\mathcal{N}_i$ of the graph is associated with a score $C(\mathcal{N}_i) \in [0, 1]$ which represents the likelihood of the node being a true instance of the keyword, i.e.:

$$C(\mathcal{N}_i) = \begin{cases} C_{ast}(\mathcal{N}_i) & \mathcal{N}_i \text{ is a hypothesized detection} \\ 1 & \mathcal{N}_i \text{ is a keyword exemplar} \end{cases}$$

Note that $C(\mathcal{N}_i)$ is set to 1 if $\mathcal{N}_i$ is a keyword exemplar since a keyword exemplar is surely a true instance of the keyword.

After the graph is constructed, as illustrated in Figure 4.3(b), we are ready to use the GRW algorithm [146] to refine the scores. The graph based score for a node $\mathcal{N}_i$ ($i = 1, 2, \dots, N+M$) is denoted as $G(\mathcal{N}_i)$. It can be estimated iteratively until convergence, e.g. by using the power method [158] as in [47, 157]. Specifically, it is initialized by

$G^0(\mathcal{N}_i) = C(\mathcal{N}_i)$. At a step t of the GRW algorithm, the graph-based score $G^t(\mathcal{N}_i)$ of the node $\mathcal{N}_i$ is updated as follows

$$\begin{aligned} G^t(\mathcal{N}_i) = & (1 - \alpha - \gamma)C(\mathcal{N}_i) \\ & + \alpha \sum_{\mathcal{N}_j \in H_{i,d}} G^{t-1}(\mathcal{N}_j)S'(\mathcal{N}_j, \mathcal{N}_i) \\ & + \gamma \sum_{\mathcal{N}_k \in H_{i,e}} G^{t-1}(\mathcal{N}_k)S'(\mathcal{N}_k, \mathcal{N}_i) \end{aligned} \quad (4.16)$$

where $H_{i,d}$ is the set of nearest neighbor nodes of $\mathcal{N}_i$ that represent hypothesized detections. Similarly, $H_{i,e}$ is the set of nearest neighbor nodes of $\mathcal{N}_i$ that represent exemplars.

Equation (4.16) shows that the new score for the node $\mathcal{N}_i$ is a weighted sum of its original score $C(\mathcal{N}_i)$, the weighted sum of scores of its neighboring detection nodes, and the weighted sum of the scores of its neighboring exemplar nodes. Two parameters α and γ are used to control the importance of scores from neighboring detection nodes and that from neighboring exemplar nodes respectively. These two parameters will be tuned using the development data.

In (4.15), the contribution of a neighbor node $\mathcal{N}_j$ to the node $\mathcal{N}_i$ is proportional to the similarity between them, i.e. $S(\mathcal{N}_j, \mathcal{N}_i)$. Hence, nodes with high acoustic similarity tend to have similar graph-based scores after applying the GRW algorithm. As a result, scores of detection nodes which are acoustically similar to exemplar nodes, will be boosted after using the GRW.

Once the graph based scores are estimated, the final scores are then computed as

$$C'(\mathcal{N}_i) = C(\mathcal{N}_i)^\delta G(\mathcal{N}_i)^{1-\delta} \quad (4.17)$$

where δ controls the balance of ASR based scores and the graph-based scores. We found that although $C(\mathcal{N}_i)$ is already included in the calculation of $G(\mathcal{N}_i)$ in Equation 4.16, interpolating between $C(\mathcal{N}_i)$ and $G(\mathcal{N}_i)$ still results in better performance. This interpolation was also suggested in the GBR method [45, 47].

The complexity of the graph-based method is approximated by that of the graph construction and random walk algorithms. In graph construction, we first construct the fully connected graph with $(M + N)$ nodes, which involves the computation of $(M + N)(M + N - 1)$ acoustic similarity scores between all pairs of nodes. Therefore, the complexity of the graph construction is $O((M + N)(M + N - 1))$. In the GRW algorithm,

there are $(M + N)$ equations like 4.16, each having at most $(M + N)$ additions. Therefore, if the number of iterations is T , then the complexity of the GRW algorithm is $O(T(M + N)^2)$. Overall, the complexity of the proposed graph-based method is $O(T(M + N)^2)$. In our experiments, we observed that T is usually less than 100, the number of exemplars M for each keyword is normally less than 100, and the number of detections N is usually less than 1000. Therefore, the complexity of the RAE-SG is generally low.

4.6 Experiment settings

4.6.1 KWS datasets

We use the same dataset, i.e. NIST OpenKWS15 Swahili, as in Chapter 3. We perform re-scoring on the list of detections produced by the partial search with the GTP technique. Besides morpheme and grapheme systems, we also build a word-based system. This system was built with trigram language model trained on the transcriptions of the training acoustic data. The acoustic model of the word-based system is a DNN with 5 layers, each has 2048 nodes. The Word Error Rate of the system on the development data is 48.2%.

For KWS, we use the WFST framework [97] to perform indexing and search. The well-known KST normalization [41, 131] is applied on top of raw ASR scores as well as all re-ranking methods.

To evaluate the STD results, in this work we report the MTWV results as the evaluation metric instead of ATWV to eliminate the effect of the threshold selection when estimating ATWV. Additionally, it is expected that after re-ranking, correct detections should have higher ranks than the false alarm ones. Thus to measure the ranking performance, MAP at top 100 is used as another evaluation metric in this work. It is worth noting that MTWV and MAP are not always correlated to each other.

4.6.2 Setup for keyword exemplar generation

As mentioned in Section 4.3, target and concatenation costs consist of various mismatch penalties (or sub-costs). The different types of sub-costs are summarized in Table 4.1. In this work, except for the concatenation smoothness sub-cost, other sub-costs are assigned

Type of sub-costs	Descriptions of sub-costs
Target sub-costs	h_i and i^{th} subword have different left context
	h_i and i^{th} subword have different right context
Concatenation sub-costs	h_i and h_{i+1} come from different audio
	h_i and h_{i+1} have different speaker's gender
	h_i and h_{i+1} are spoken by different speakers
	Smoothness cost: Euclidean distance of MFCC frames at the concatenation point [159]

Table 4.1: Summary of sub-costs for keyword exemplars generation. h_i and h_{i+1} are instances of i^{th} and $(i+1)^{th}$ subword units respectively

by the fix value of 1. For the concatenation smoothness, the Euclidean distances between all pairs of instances of two adjacent subwords are estimated then transformed into $[0,1]$ by using the *min-max* normalization [153]. The normalized Euclidean distance will be used as the concatenation smoothness sub-cost.

4.6.3 Setup for acoustic similarity estimation

The DTW-based acoustic similarity (see Section 4.4.1) is estimated on multilingual bottleneck feature (BNF). The BNF has been shown to work well for various tasks such as speech recognition [34, 35] and speaker recognition [160–162]. It outperforms other types of features such as MFCC [27, 29, 74] or posteriorgram [20, 163] in the tasks of acoustic word embedding [164] and QbE-STD [165], which are closely related to the task of acoustic similarity estimation. Therefore, we perform DTW on multilingual BNFs. When estimating the DTW, we use Cosine distance as per-frame distance metric [165]. The data used for training the BNF is obtained from various corpora as in Table 4.2. From such data, we extract 22 dimensional filter-bank features plus 3 dimensional pitch features to train a DNN as a feature extractor. The detailed procedure can be found in [166].

4.7 Experiment results and analysis

In this section, we first present the KWS performance of different acoustic similarity estimation techniques, which were described in Section 4.4. We then demonstrate that

Languages	Speech data (hour)	Lexicon Size (k)
Cantonese	141.3	20
Turkish	77.2	45
Pashto	78.4	23
Tagalog	84.5	33
Vietnamese	87.7	9.6
Tamil	69.3	69

Table 4.2: The distribution of the multilingual data for training the bottleneck feature (BNF) extractor.

Methods	Need training?	Training time	Inference time	Performance
Min-max normalization	No	N. A.	Very fast	Lowest
Logistic regression	Yes	Very fast	Very fast	Medium
Neural network	Yes	Fast	Fast	Highest

Table 4.3: A comparison among three acoustic similarity estimation methods.

keyword exemplars are effective for re-ranking KWS results. Finally, we will analyse in detail the KWS performance of proposed re-ranking methods.

4.7.1 The effect of different acoustic similarity estimation methods

We now analyse the effect of different acoustic similarity estimation methods, i.e. *min* – *max* normalization, logistic regression calibration and neural network, on re-ranking performance. For simplicity, we report the results on morpheme-based systems. We also select RAE-SF as the re-ranking method. The DET curves of different similarity estimation methods are shown in Figure 4.4. It is observed that logistic regression is slightly better than *min* – *max* normalization; and these are inferior compared to the neural network based method. Therefore, we conclude that acoustic similarity estimation, and hence re-ranking performance, can be improved using the neural network approach. Table 4.3 summaries strength and weaknesses of the three methods. For all experiments in the following sections, we will use the neural network method to estimate the acoustic similarity.

Figure 4.4: The DET curves of different acoustic similarity estimation methods on the morpheme system for IV keywords (top) and OOV keywords (bottom).

4.7.2 The effect of keyword exemplars for re-ranking hypothesized detections

In this section, we focus on comparing re-ranking methods with and without using keyword exemplars. The experimental results are shown in Table 4.4. In the table, column 1 shows the results without re-scoring, column 2 and 4 are the results produced by the two baseline techniques, i.e. PRF and GBR, respectively; while column 3 and 5 are the results of proposed RAE methods that make use of keyword exemplars.

It is observed that the performance in terms of MTWV and MAP of proposed re-scoring methods is significantly better than that of the baseline original systems (i.e. without re-scoring). For example, overall, RAE-SG outperforms the baseline by 3.9% absolute MTWV and 3.2% MAP. Another observation is that the proposed methods outperform their corresponding baselines, i.e. PRF and GBR, for both MTWV and MAP. Specifically, column 3 outperforms column 2, while column 5 outperforms column 4. The improvement is consistent among word, morpheme and grapheme systems, and both IV and OOV keyword types. Overall, RAE-SF outperforms PRF by 1.4% absolute MTWV and 2.5% MAP; while RAE-SG outperforms its baseline GBR by 1.8% absolute MTWV and 2.0% MAP respectively. Therefore, we can conclude that the proposed RAE framework produces better performance than the PRF and GBR methods due to its use of keyword exemplars.

Column no.			1	2	3	4	5
Methods			Without rescoring	PRF	RAE-SF	GBR	RAE-SG
Used exemplars			No	No	Yes	No	Yes
IV	Word	MTWV	0.5377	0.5482	0.5546	0.5521	0.5652
		MAP	0.5999	0.6042	0.6214	0.6057	0.6177
	Morpheme	MTWV	0.4291	0.4538	0.4652	0.4564	0.4750
		MAP	0.4901	0.4954	0.5182	0.4986	0.5195
	Grapheme	MTWV	0.3169	0.3415	0.3605	0.3422	0.3646
		MAP	0.3664	0.3727	0.3940	0.3786	0.3933
OOV	Morpheme	MTWV	0.4048	0.4235	0.4402	0.4253	0.4440
		MAP	0.3584	0.3650	0.4057	0.3727	0.4065
	Grapheme	MTWV	0.3315	0.3459	0.3632	0.3499	0.3666
		MAP	0.3159	0.3364	0.3581	0.3390	0.3547
Average over all rows		MTWV	0.4040	0.4226	0.4367	0.4252	0.4431
		MAP	0.4261	0.4347	0.4595	0.4389	0.4583

Table 4.4: KWS performance of the proposed re-ranking methods, i.e. RAE by score fusion (RAE-SF) and RAE by similarity graph (RAE-SG), on NIST OpenKWS Swahili language pack (evalpart1). “Word”, “morpheme” and “grapheme” refer to results obtained by the word-based, morpheme-based and grapheme-based KWS systems respectively. “MTWV” denotes Maximum Term-Weighted Value, while “MAP” denotes Mean Average Precision. PRF and GBR denote Pseudo Relevance Feedback and Graph-based Re-ranking which are the corresponding baselines of RAE-SF and RAE-SG.

4.7.3 Analysis of RAE methods

In this section, we compare the two proposed re-ranking algorithms, namely RAE-SF and RAE-SG, and provide analysis on RAE-SG. From the results in Table 4.4, RAE-SG (column 5) slightly outperforms RAE-SF (column 3) in most test cases. This could be due to that the RAE-SF method processes each detection individually while the RAE-SG method performs re-ranking globally. Note that in the RAE-SG, we need to compute the acoustic similarity between each detection and keyword exemplar, hence the computational complexity of RAE-SF is $O(MN)$ which is lower than that of RAE-SG method. Table 4.5 demonstrates a comparison between RAE-SF and RAE-SG methods.

Methods	Iterative?	Complexity	Performance
RAE-SF	No	$O(MN)$	Low
RAE-SG	Yes	$O(T(M + N)^2)$	High

Table 4.5: A comparison between the two proposed methods RAE-SF and RAE-SG. M , N , T are number of detections, keyword exemplars and iterations respectively.

Figure 4.5: The comparison between RAE-SG and the GBR on rare, normal and frequent keywords.

We also analyse how the RAE-SG method improves the KWS performance. In the first analysis, we investigate how the number of keyword exemplars affects the performance of the RAE-SG for IV keywords. We hypothesize that the more keyword exemplars we have for re-ranking, the more improvement we can obtain. In Figure 4.5, the performance of re-ranking is shown separately for three categories of keywords, i.e. “rare keywords” (less than 4 exemplars), “normal keywords” (4 to 10 exemplars), and “frequent keywords”

(more than 10 exemplars). Results of word, morpheme and grapheme systems are shown. From the results, we can observe that the improvements on frequent keywords are usually larger than on rare keywords, especially for subword systems. This could be due to that the more exemplars we use for re-ranking, the more they can affect the detection scores, and hence potentially improve the KWS performance.

Figure 4.6: The comparison between RAE-SG and the GBR on short, medium and long keywords for IV keywords (top figure) and OOV keywords (bottom figure).

We also analyze the performance of RAE-SG on keywords of different keyword length: short keywords (average duration on exemplars is less than 0.45s), medium keywords (average duration is from 0.45s to 0.7s), and long keywords (the rest). Figure 4.6 shows the comparison results for both IV and OOV keywords. It is observed that the RAE-SG generally produces more gains for short keywords than for long keywords. This might be because that the short keywords have much poorer performance than the long keywords due to shorter acoustic evidence.

From Equation 4.16 and 4.17, performance of the RAE-SG method depends on the three hyper-parameters, i.e. α , γ and δ . Therefore, it is worth investigating the sensitivity of the RAE-SG method with respect to these parameters. We consider the MTWV score

of IV keywords produced by the morpheme-based system, i.e. $MTWV = 0.4750$. This result is obtained at $\alpha = 0.2$, $\gamma = 0.4$ and $\delta = 0.2$ (which is achieved by tuning these parameters on the development data). To test the effect of each parameter, we vary it and fix the other parameters. The results are shown in Figure 4.7. It can be seen that the performance difference between the highest and lowest values is less than 0.04 absolute, or less than 10% relative, $MTWV$ on all cases. Therefore, the RAE-SG method is not highly sensitive with the parameter changing.

Figure 4.7: The $MTWV$ score of the RAE-SG on IV keywords when varying each of three parameters, α , γ and δ , and fixing the others.

4.8 Summary

We studied a novel re-ranking framework, called RAE, that exploits the information of keyword exemplars to complement the conventional ASR-based KWS systems. Exemplars are true instances of the keyword generated from a labelled data, i.e. the training data of the ASR system, hence avoid making any assumption as some existing techniques such as PRF and GBR. The proposed RAE framework is implemented in three steps, i.e. keyword exemplar generation, acoustic similarity estimation and re-ranking with keyword exemplars. Performance evaluation on the NIST OpenKWS 2015 benchmarking task showed that the proposed RAE framework significantly outperforms previous re-ranking methods that do not make use of keyword exemplars. Further analysis showed that the improvement for frequent keywords is usually larger than that for rare keywords. This is straightforward since the more exemplars we have, the more they affect the detection scores, and hence potentially improve the STD performance. Additionally, we observed that the proposed framework is generally more effective for short keywords than for long

keywords. In conclusion, our results showed that the acoustic exemplars of keywords provide complementary information that could be used to improve the ASR-based STD systems.

Chapter 5

Re-scoring Hypothesized Detections Using Lattice-based Competing Hypotheses

5.1 Introduction

In the literature, a large portion of re-scoring methods for STD aimed to study features which are informative to distinguish correct detections from false alarms. Common features include lexical features [49–51], utterance/conversation features [49, 54] and prosody features [50, 51].

In this work, we investigate novel lattice-based features derived from competing hypotheses of each detection to perform re-scoring. In particular, for each detection, its competing hypotheses are hypotheses in the detection’s lattice that have similar time information to the detection. From those hypotheses, novel features, that characterize the relative confidence of the detection hypothesis as compared to other hypotheses, are computed. These features exhibit complementary information to the standard detection score, i.e. lattice-based posterior probability. As a result, these are informative and potential for the re-scoring task.

To integrate the different features, the common way is to formulate the re-scoring task as a binary classification problem [40, 50]. In particular, a development keyword list is searched on a development dataset to obtain the list of detections. Then for each detection, features, such as those mentioned above, are extracted together with the true/false label to train the classifier. At the test stage, the posterior probabilities of

the classifier can be considered as new confidence scores. However, optimizing the re-scoring model with a standard cost function, i.e. cross entropy, might not lead to the best STD performance since the main STD metric, i.e. TWV, has different costs for missing and false alarm detections. Therefore, a two-stage process is needed to estimate new confidence scores. Specifically, at the first stage, a binary classifier is applied to integrate features into a single score which leads to more informed “hit/false alarm” decisions. Then, in the second stage, a score normalization method such as KST or STO is applied on top of the classifier results to produce final confidence scores.

This chapter is organized as follows. In Section 5.2, we first introduce the competing hypotheses of each detection and novel features derived from such the collection. We then describe the two-stage process to incorporate these novel features into a single confidence score in Section 5.3. Section 5.4 describes the experimental settings. Results and discussions are presented in Section 5.5 and 5.6 respectively. Finally, we conclude in Section 5.7.

5.2 Novel features derived from competing hypotheses set of each detection

Let’s consider the simple case, where the keyword K to be searched contains only a single word. Each hypothesized detection d of the keyword K is presented as a tuple (d_L, d_S, d_E, d_P) which are the lattice, start time, end time and posterior probability of d . We define the competing hypotheses $T(d)$ as the set of arcs in the lattice d_L that overlap with d . Each hypothesis $h \in T(d)$ is presented by a tuple $(h_I, h_L, h_S, h_E, h_P)$ which are the label id, lattice, start time, end time and posterior probability of h . Figure 5.1 demonstrates the competing hypotheses of a detection d on a lattice. Specifically, in Figure 5.1 the detection d is the red color arc which is presented by a tuple $(\text{“veal”}, L, 3.3, 4.0, 0.4)$ while the set $T(d)$ consists of 4 arcs, i.e.

$$T(d) = \{(\text{‘veal’}, L, 3.3, 4.0, 0.4), (\text{‘very’}, L, 3.3, 3.9, 0.25), (\text{‘villa’}, L, 3.4, 3.9, 0.15), (\text{‘velum’}, L, 3.4, 4.0, 0.2)\}.$$

From the set $T(d)$, the following features are computed:

ranking_score: This feature is the rank of the posterior probability of d compared to posterior probabilities of all items in $T(d)$. It is clear that for the same posterior

Figure 5.1: A detection hypothesis d and its competing hypotheses $T(d)$.

probability, detections with low *ranking_score* are more likely to be true hits. Thus *ranking_score* can provide additional information to determine if a hypothesized detection should be a correct detection or a false alarm.

relative_to_max: This feature is defined as the relative score of the detection d compared to the best score of items in $T(d)$, i.e.,

$$relative_to_max(d) = \frac{d_P}{\max_{h \in T(d)}(h_P)} \quad (5.1)$$

This information indicates how much the best hypothesis in $T(d)$ dominates the detection hypothesis d . The low *relative_to_max* value means the detection hypothesis is highly dominated by another best hypothesis, which means it is less likely to be a correct detection even it might have good *ranking_score*. This *relative_to_max* feature can provide extra information to *ranking_score* and posterior probability to improve hit/false alarm decisions.

number_of_competing_hypotheses: This feature indicates the number of competing hypotheses in $T(d)$. It is straightforward to see that many more competing hypotheses indicate that d is less confident, thus less likely to be a hit.

max_sim_hypotheses: Let d_s be the speech segment at the detected location of d . This feature is computed as the maximum acoustic similarity between d_s and any hypothesis $h \in T(d)$, i.e.

$$\text{max_sim_hypotheses}(d) = \max_{h \in T(d)}(S(h, d_s)) \quad (5.2)$$

where $S(h, d_s)$ is defined as the similarity between d_s and the hypothesis h , which is computed as the average acoustic similarity between d_s and all exemplars of h_I (see the Equation 4.12 in Section 4). This feature describes the acoustic similarity between the signal at the detected location of d and other hypotheses in $T(d)$. The high *max_sim_hypotheses* indicates that the signal at the detected location is acoustically similar to examples of another hypothesis, which implies that d is less confident.

Now we consider the case that the keyword K consists of more than one word, i.e. $K = K_1 K_2 \dots K_n$. A detection d of K is confident as long as all $K_1, K_2, \dots, K_n$ are detected confidently. Therefore, we infer the value of above features of a detection d by the worst value throughout $K_i, i = 1 \dots n$, i.e.

$$\text{ranking_score}(d) = \max_{i=1 \dots n}(\text{ranking_score}(K_i)) \quad (5.3)$$

$$\text{relative_to_max}(d) = \min_{i=1 \dots n}(\text{relative_to_max}(K_i)) \quad (5.4)$$

$$\text{number_of_competing_hypotheses}(d) = \max_{i=1 \dots n}(\text{number_of_competing_hypotheses}(K_i)) \quad (5.5)$$

$$\text{max_sim_hypotheses}(d) = \max_{i=1 \dots n}(\text{max_sim_hypotheses}(K_i)) \quad (5.6)$$

5.3 Two-stage confidence estimation

To integrate different features, the confidence estimation has been formulated as a binary classification problem [40, 49, 50, 69, 71, 72]. Each hypothesized detection is associated with features, such as those mentioned above. Then a classifier can be trained to classify a detection to be true or not based on its feature if a set of training keywords and their associated true/false labels are available.

Generally, the classifier is trained using the cross entropy loss function to maximize the classification accuracy. In particular, let Q be the set of keywords to be considered;

and for a keyword $K \in Q$, let $d_1, d_2, \dots$ be its list of detections. For each detection d_i , let y_i be the binary true label, i.e. $y_i \in \{0, 1\}$, obtained by comparing d_i with the ground truth; and $\hat{y}_i$ be the predicted confidence score produced by the binary classifier. Then, the cross-entropy objective [167] function is defined as

$$CE = - \sum_{K \in Q} \sum_{d_i \in K} (y_i \log \hat{y}_i + (1 - y_i) \log(1 - \hat{y}_i)) \quad (5.7)$$

The above discriminative classifier provides scores that potentially lead to a more in-formed “hit/FA” decision, i.e. to maximize the classification accuracy. However, maximizing the classification accuracy does not guarantee a better TWV performance. This is because the TWV metric has different costs for a false positive (i.e. a false alarm) and a false negative (i.e. a miss). Specifically, from Equations 2.7 and 2.8, missing a correct detection costs $\frac{1}{N_K}$ (where N_K is the number of true keyword occurrences), which is usually higher than the cost of introducing a false alarm $\frac{\beta}{T - N_K}$ (where T is the corpus length). Since the primary metric for STD is TWV, it is necessary to apply a score normalization technique that is designed for the TWV such as KST normalization [41, 131]. Specifically, the binary classifier is first applied on each detection to generate a new classifier-based score. Then, the KST normalization is applied on the list of detections with classifier-based scores to generate final confidence scores for the list of detections. This two-stage scheme is shown in Figure 5.2.

Figure 5.2: The two-stage process to estimate the new confidence score. The binary classifier produces discriminative confidence scores, while a metric-based score normalization outputs a new set of scores that lead to optimum TWV metric.

5.4 Experimental setup

The experiments are conducted in the NIST OpenKWS15, i.e. Swahili data. In this work, we report the experimental results for IV keywords on the word-based system.

The detail about training, development and evaluation data, as well as keyword list and the STD system are described in Section 3.3 and 4.6.3.

The re-scoring model can be any binary classifier such as Logistic Regression, Support Vector Machine, Neural Network (NN). In this work, we choose NN as the classifier since it produces slightly better results than others [40, 50]. Table 5.1 shows the configuration to train the neural network. To obtain training data for the binary classifier, we search the development keyword list on the development set to generate a list of hypothesized detections. Then, for each hypothesized detection, various features (such as those mentioned in Section 5.2) were extracted. The true/false label of each detection was also obtained by aligning the detection with the reference transcription. This process generates over 121,000 examples for the training of the NN. Note that the obtained data is very unbalanced: less than 2% samples have true labels (correct detections).

Configuration	Value
Optimizer	Stochastic Gradient Descent
Number of hidden layers	2
Size of each hidden layer	50
Learning rate	0.01
Batch-size	256
L2 norm weight	3e-5
Max epochs	100

Table 5.1: The configuration to train the neural network based binary classifier.

In addition to lattice-based posterior probability, in this work, we also study the combination of proposed features with some existing features reported in the literature. Those existing features include:

- Lexical features: In [49–51], researchers have shown that lexical properties of the keyword impact the reliability of detections. Therefore, various lexical features of the keyword such as the number of words, phones, vowels, consonants, syllables of the keyword, are extracted for the confidence estimation.
- Posterior probability based features: For each detection, various features are derived from the lattice-based posterior probability. Those include results of score normalizations methods, such as KST and STO normalization (see Section 2.3.2.1), and the ranking of the detection as compared to other detections in the potting list.

- Speaking rate [168]: It is estimated as detection duration divided by the number of phones of the keyword. This feature is informative since detection with an abnormal speaking rate is more likely to be a false alarm.
- Position: It describes the position of the detection, i.e. at the starting, ending or middle of the corresponding utterance. This feature is motivated by the fact that words at the beginning or end of an utterance are more likely to be false alarms [132].
- Scores generated by the graph-based re-scoring method in Chapter 4.5.2

5.5 Experimental results

This section presents the experimental results of the proposed re-scoring scheme. In particular, we first demonstrate that the proposed features in Section 5.2 are effective for rescore KWS detections. Specifically, by using the Mann-Whitney U-test [169], we show that the new confidence scores generated using the proposed features help to effectively separate correct detections out of false alarm ones. Then, the KWS performance (MTWV and MAP scores) are reported on the *evalpart1* dataset.

5.5.1 Effect Size of Mann-Whitney U-test

In this section, we measure the separation between two distributions, i.e. correct detections and false alarms, with and without proposed features to show that these features are effective. The separation can be estimated using the Mann-Whitney U-test, which is a non-parametric technique. Such technique is described as follows.

Let D_{corr} and D_{fa} be two sets of score of correct and false alarm detections respectively; and $D = D_{corr} \cup D_{fa}$ is the total population. We define a non-parametric score ρ [170] as measurement of the overlapping between two populations, i.e. D_{corr} and D_{fa} . ρ is estimated as follows :

$$R_{corr} = \sum_{d \in D_{corr}} rank(d) \quad (5.8)$$

$$U_{corr} = R_{corr} - \frac{\|D_{corr}\| (\|D_{corr}\| + 1)}{2} \quad (5.9)$$

$$\rho = \left| \frac{U_{corr}}{\|D_{corr}\| \times \|D_{fa}\|} - 0.5 \right| \times 2 \quad (5.10)$$

where $rank(d)$ is the rank of d in D , and $\|\cdot\|$ is the cardinal of the set. The ρ value is between 0 and 1, and higher ρ indicates a better separation of D_{corr} and D_{fa} .

We will show that the new detection scores generated by the binary classifier using proposed features result in a higher ρ value, which indicates that proposed features help to better separate correct detections out of false alarms. Table 5.2 shows the ρ values corresponding to two sets of detection scores: the first set are original detection scores, while the second set are the new confidence scores generated by the NN classifier trained using the ASR scores plus proposed features. It can be seen that incorporating proposed features results in 3.0% absolute ρ improvement. In other words, the proposed features are potential for re-scoring KWS detections.

Original detection scores	New confidence scores
0.5291	0.5593

Table 5.2: The ρ values of original ASR scores and the new confidence scores.

5.5.2 The efficiency of novel features for re-scoring STD results

This section compares the STD results of new detection scores generated by the NN classifier with and without the novel features. We first demonstrate that the novel features are effective when being combined with the original detection scores, i.e lattice-based posterior probabilities. Specifically, we compare the STD results of original detection scores (denoted as SYSTEM-1) to the results of the re-scoring model that is trained with the original score plus novel features (which is denoted as SYSTEM-2). We then show that when combining with other features described in Section 5.4, the proposed features still produce additive gains which demonstrates that the proposed features are complementary to existing features. We denote the NN model trained with existing features as SYSTEM-3 and the model trained with existing features plus novel features as SYSTEM-4. The MTWV and MAP of above systems are presented in Table 5.3 and their DET curves are shown in Figure 5.3.

Methods	Descriptions	MTWV	MAP
SYSTEM-1	Original detection score	0.5377	0.5999
SYSTEM-2	NN model trained with original score plus novel features	0.5696	0.6223
SYSTEM-3	NN model trained with original score plus features in Section 5.4	0.5782	0.6287
SYSTEM-4	NN model trained with original score, features in Section 5.4 and novel features	0.5974	0.6429

Table 5.3: The MTWV and MAP metrics of different models trained with and without novel features.

From Table 5.3, it is observed that the SYSTEM-2 produces significant improvement, i.e. 3.2% and 2.2% absolute MTWV and MAP gain, over the baseline SYSTEM-1. This indicates that the novel features are very effective when combining with the original detection scores. Additionally, SYSTEM-4 outperforms SYSTEM-3 by 1.9% and

Figure 5.3: DET curves of NN-based re-scoring models trained with and without novel features.

1.4% absolute MTWV and MAP. It demonstrates that the proposed features are complementary to other existing features. The corresponding DET curves in Figure 5.3 also demonstrate the same trend.

Now we analyse the complexity of four systems described in Table 5.3. Without loss of generality, we consider SYSTEM-2. For the first stage, i.e. binary classifier, given the number of input features 5 (lattice-based posterior probability plus four novel features) and the network configuration in Table 5.1 (i.e. two hidden layers, each has 50 nodes), the number of parameters (without bias terms) is $5 \times 50 + 50 \times 50 + 50 \times 2 = 2850$. With such a small number of parameters, training the binary classifier is fast and inference can be done in real-time. For the second stage, i.e. KST score normalization (See Section 2.3.2.1), it only needs to loop through all detections two times: the first loop is used to estimate the specific threshold using Equation 2.18, and the second loop transforms scores using Equation 2.19. Therefore, the second stage generally can also be done in real-time. As a result, SYSTEM-2 can perform in real-time.

Effect of the second stage (score normalization) on the STD performance

We now analyze the effect of the second stage, i.e. TWV-based score normalization, on the STD performance. Specifically, we compare the STD performance (MTWV and MAP) of SYSTEM 2 (described in Table 5.3) with its alternative (denoted as SYSTEM 2 W/O SCORE NORM) where no score normalization is applied. The experimental comparison is present in Table 5.4. It can be seen that the second stage helps to improve the MTWV substantially, from 0.4782 to 0.5696. Note that score normalization does not affect the ranking for each keyword hence the MAP scores with and without the second stage are the same.

Methods	Descriptions	MTWV	MAP
SYSTEM-2	The two-stage confidence estimation (see Section 5.3)	0.5696	0.6223
SYSTEM-2 W/O SCORE NORM	Confidence estimation without the second stage	0.4782	0.6223

Table 5.4: The MTWV and MAP metrics with and without the second stage, namely TWV-based score normalization.

5.6 Discussion

The two-stage process in Section 5.3 separately optimizes the classification accuracy and STD performance, i.e. the TWV metric. An alternative approach is to train the classifier so that it is directly optimized for the TWV. To achieve that goal, we need to replace the standard cross entropy by a new cost/objective function that reflects the TWV. This can be done using TWV-scaled cross entropy [49] or smooth-ATWV [69] techniques which are described as follows.

TWV-scaled cross entropy objective function

For TWV-scaled cross entropy, the general idea is to apply different weights for positive and negative samples during training the binary classifier since the cost of missing a true detection is different from the cost of introducing a false alarm. In particular, from Equations 2.7 and 2.8, missing a correct detection costs $\frac{1}{N_K}$, which is usually higher than the cost of introducing a false alarm, i.e. $\frac{\beta}{T-N_K}$. This is the motivation of the TWV-scaled cross-entropy objective function proposed in [49]. Before introducing the TWV-scaled cross-entropy, let Q be the set of keywords to be considered; and for a keyword $K \in Q$, let $d_i, i = 1, 2, \dots$ be the keyword's detections. For each detection d_i , let y_i be the binary true label, i.e. $y_i \in \{0, 1\}$; and $\hat{y}_i$ be the predicted confidence score produced by the binary classifier. Then the TWV-scaled cross-entropy objective function is defined as:

$$\text{TWV_scaled_CE} = -\frac{1}{|Q|} \sum_{K \in Q} \sum_{d_i \in K} W_i (y_i \log \hat{y}_i + (1 - y_i) \log(1 - \hat{y}_i)) \quad (5.11)$$

where W_i is the TWV-based scale defined as

$$W_i = \begin{cases} \frac{1}{N_K} & , y_i = 1 \\ \frac{\beta}{T-N_K} & , y_i = 0 \end{cases} \quad (5.12)$$

In Equation 5.11, the outer loop goes through the list of keywords Q while the inner loop goes through list of detections for a keyword $K \in Q$. Positive and negative detections will have different weight W_i as indicated in Equation 5.12.

Smooth-ATWV objective function

Smooth-ATWV [69] is another solution toward training the binary classifier with an ATWV-based objective function. It would be optimum if we can use ATWV metric as

an objective function to train the binary classifier. However, as can be seen later, ATWV metric is not differentiable, hence cannot be used to train the classifier. The idea of [69] is to approximate ATWV by a differentiable function so that it can be used to train the binary classifier.

To begin with, let's re-write the formula to calculate the ATWV as follow

$$\text{ATWV}(\theta) = \frac{1}{|Q|} \sum_{K \in Q} \sum_{d_i \in K} \mathbf{I}_{[\hat{y}_i > \theta]} \left(\frac{1}{N_K} \mathbf{I}_{[d_i \text{ is correct}]} - \frac{\beta}{T - N_K} \mathbf{I}_{[d_i \text{ is a FA}]} \right) \quad (5.13)$$

where $\hat{y}_i$ is the predicted score of the detection d_i and $\mathbf{I}_{[condition]}$ is the indicator function which equals 1 when *condition* is true. It is natural to use the above equation as the objective function of the classifier. However, the difficulty of this objective function is that the indicator function $\mathbf{I}_{[\hat{y}_i > \theta]}$ cannot be differentiated. Vinyals et.al [69] proposed a smooth version of Equation 5.13, which is denoted as *SmoothATWV*:

$$\text{SmoothATWV}(\theta) = \frac{1}{|Q|} \sum_{K \in Q} \sum_{d_i \in K} \sigma(S(\hat{y}_i - \theta)) \left(\frac{1}{N_K} \mathbf{I}_{[d_i \text{ is correct}]} - \frac{\beta}{T - N_K} \mathbf{I}_{[d_i \text{ is a FA}]} \right) \quad (5.14)$$

where σ denotes the sigmoid function $\sigma(x) = \frac{1}{1+e^{-x}}$, S is a constant. The key idea of the *SmoothATWV* is that the sigmoid function $\sigma(S(\hat{y}_i - \theta))$ is used as an approximation for the indicator function $\mathbf{I}_{[\hat{y}_i > \theta]}$. As S goes to infinity, the sigmoid function converges to the indicator function (shown in Figure 5.4).

Figure 5.4: Approximation of indicator function.

To compare the two cost functions, TWV-scaled and Smooth-ATWV, to the two-stage confidence estimation (presented in Section 5.3), we train the binary classifier with all cost functions using the same set of features as SYSTEM-2 in Table 5.3. The experimental results are shown in Table 5.5. From the experimental results, all three cost functions produced similar TWV scores since all methods are designed to optimize TWV metric. However, the MAP scores of the TWV-scaled and Smooth-ATWV are inferior to that of the two-stage process in Section 5.3. This might be because that in the first-stage in Section 5.3, the classifier is trained with the standard cross-entropy which leads to optimum YES/NO decisions and hence results in improved ranking performance (i.e. MAP score). The other methods, on the other hand, are designed to optimize the TWV metric, which might not guarantee the optimum MAP. Another reason for the poor performance of Smooth-ATWV is the trade-off between the degree of approximation and the difficulty of training the binary classifier. Specifically, a higher value for S in Equation 5.14 implies that the sigmoid function is closer to the indicator function at the cost of making the gradient close to 0, which results in difficulty in training the classifier.

Methods	MTWV	MAP
Two-stage confidence estimation (i.e. SYSTEM-2)	0.5696	0.6223
TWV-scaled objective function	0.5698	0.6211
Smooth-ATWV objective function	0.5689	0.6174

Table 5.5: The MTWV and MAP performance of TWV-scaled and Smooth-ATWV objective functions compared to that of the two-stage framework.

5.7 Summary

In this chapter, we proposed to use the lattice-based competing hypotheses of each detection to perform re-scoring. Competing hypotheses of a detection are hypotheses with the same lattice and similar time spans as the detection. From the list of competing hypotheses, various novel features are extracted and integrated into a discriminative classifier to estimate more accurate detection scores. We hypothesized that the features exhibit complementary information to the standard detection score, i.e. the lattice-based posterior probability, hence these are informative and potential for the re-scoring task.

To further optimize for the TWV metric, a TWV-based score normalization method is applied on top of the confidence scores produced by the classifier. Experimental results showed that these features are effective in separating correct detections from false alarms. Additionally, these novel features lead to better STD performance when combining with other existing features. Experimental results also showed that the TWV-based score normalization results in better STD performance which indicate that this step is necessary.

Chapter 6

Conclusion and Future Work

This thesis has focused on the topic of STD, or KWS, where the aim is to detect the presence of a keyword in a speech corpus. The most common approach for STD is to rely on automatic speech recognition (ASR) to transcribe the speech content into the lattice representation then apply indexing/retrieval techniques on the lattice. The major challenge of STD systems is the high miss and false alarm rate due to the imperfect modelling of the underlying ASR component. This thesis has developed novel techniques to compensate ASR errors to improve the STD performance. To conclude the work, Section 6.1 first summarises the contributions proposed in this thesis. Then, Section 6.2 discusses some potential directions for future work.

6.1 Contributions

In this section, we first summarize our three contributions from Section 6.1.1 to Section 6.1.3. We then discuss the combination of proposed approaches in Section 6.1.4 to improve the STD performance.

6.1.1 A partial search approach for subword-based STD systems

The first contribution of this thesis is a partial search framework to reduce the miss rate of subword-based STD systems which was published in the SoICT 2017 conference [171]. Typically, in a subword-based STD system, the keyword is converted into a subword sequence then the sequence is searched on subword lattices. However, due to the high

error rate of subword ASR systems, it is difficult to find the entire subword sequence in a lattice, which results in a high miss rate. The main idea of the proposed approach is to split the subword sequence of the keyword into several sub-sequences, then search these sub-sequences on the subword index to produce intermediate result lists which are later merged to generate a final detections list. The proposed approach reduces the miss rate since matching short sequences is easier than matching the original long sequence. Additionally, to further reduce the miss rate, the proposed approach accepts detections that only match with a certain portion of the keyword sub-sequences. Although this can overcome the issue of high miss rate, it normally produces many more false alarms. To address the issue of high false alarm rate, we proposed two pruning methods, namely Global Threshold Pruning (GTP) and Keyword-Specific Pruning (KSP). The first method empirically uses a single threshold for all keywords while the second method gradually adjusts a keyword-specific threshold till a detection is found.

Experimental results show that the proposed approach significantly outperforms its baselines: the first one is the conventional full search method that uses the entire keyword sequence to search; and the second one is the query expansion technique which reduces the miss rate by performing approximate matching. For example, on OOV keyword list of the grapheme system, partial search with KSP outperforms the full search baseline by 6.9% and 7.4% absolute MTWV and STWV. Additionally, the proposed approach does not require any confusion model and is also much faster than the query expansion baseline.

6.1.2 A re-ranking framework that uses keyword exemplars

The second contribution of this thesis is a re-ranking framework, named Re-ranking by Acoustic Exemplar (RAE), which makes use of keyword exemplars extracted from ASR training data. This work was published in two conference and one journal papers, namely ICASSP 2016 [172], Interspeech 2016 [173] and Speech Communication [174]. Typically, the detection scores are the posterior probabilities of the keyword at the hypothesized locations and computed from the decoding lattices. These scores are derived from imperfect acoustic and language models which might result in poor STD performance. The basic idea of the proposed approach is to directly estimate the acoustic similarity between

the speech signal of each hypothesized detection and that of keyword exemplars through template matching, i.e. Dynamic Time Warping, then use the similarity to perform re-scoring. The motivation of the proposed approach is that template matching does not suffer from modelling errors and provides complementary information to the lattice-based posterior probability. Given the acoustic similarity, two re-scoring methods, RAE-SF and RAE-SG, have been proposed. The RAE-SF method estimates the average similarity between each hypothesized detection and all keywords exemplars, then interpolates such measurement with the original lattice-based posterior probabilities. The graph-based method, on the other hand, builds a fully connected graph, where each node represents a hypothesized detection or an exemplar, and then applies the random walk algorithm to estimate the new scores.

Experiments were carried out to compare the KWS performance of the proposed re-scoring framework with that of the ranking based only on ASR confidence scores as well as other re-ranking methods without using keyword exemplars. Experimental results showed that the proposed methods significantly outperform the ranking based only on ASR confidence scores. For instance, RAE-SG outperforms the baseline by 3.9% absolute MTWV and 3.2% MAP. It was also observed that the proposed methods outperform others re-scoring methods, i.e. PRF and GBR, that do not use keyword exemplars. The improvement is consistent among word, morpheme and grapheme systems, and both IV and OOV keyword types. Therefore, we concluded that the proposed RAE framework produces better performance than its baselines due to its use of keyword exemplars.

Detailed analysis was also carried out to demonstrate what types of keyword benefit more from the proposed framework. It was observed that the improvement for frequent keywords is usually larger than that for rare keywords. This could be due to that the more exemplars we use for re-ranking, the more they can affect the detection scores, and hence potentially improve the STD performance. We also analysed re-scoring performance with different keyword length and observed that the proposed framework is generally more effective for short keywords than for long keywords. This could be due to that the short keywords have much poorer performance than the long keywords due to shorter acoustic evidence.

6.1.3 A re-scoring framework that uses lattice-based competing hypotheses

The third contribution of this thesis is a re-scoring technique that utilizes the lattice-based competing hypotheses of each detection to perform re-scoring, with the work published in the ICASSP 2014 conference [71]. In particular, competing hypotheses of a detection are those at the similar time spans as the detection. From this set, novel features, that characterize the relative confidence of the detection hypothesis compared to other hypotheses, are computed and used for re-scoring. We hypothesize that those features exhibit complementary information to the standard detection score, i.e. the lattice-based posterior probability, hence these are informative and potential for the re-scoring task.

Experiments were carried out to compare the KWS performance with and without incorporating the novel features. It was observed that when combining with the original detection scores, the novel features resulted in 3.2% and 2.2% absolute MTWV and MAP improvement. It is also observed that when combining with various existing features, the proposed features still produced additive gains which demonstrated that the proposed features are complementary to the existing features.

6.1.4 Combining proposed approaches to improve STD performance

As presented from Section 6.1.1 to 6.1.3, the first proposed approach aims to reduce the miss rate while the two other approaches aim to reduce the false alarm rate for STD systems. Therefore, one interesting question is how complementary are the three approaches. In this section, we discuss this perspective and highlight the total performance improvement obtained by combining proposed approaches. The ability of combining the proposed approaches is summarized in Table 6.1 and detailed explanations are presented as below.

The second approach uses keyword exemplars to perform re-ranking on a list of hypothesized detections produced by any STD system. Therefore, it can be applied on top of the results produced by the first proposed approach, i.e. the partial search framework. In this thesis, it can be seen from Table 3.3 and 4.4 that applying the RAE-SG on top of results produced by the partial search with KSP significantly improves the MTWV

Pair of proposed approaches	Can be combined?
Approaches 1 and 2	Yes
Approaches 2 and 3	Yes
Approaches 1 and 3	No

Table 6.1: The ability of combining proposed approaches.

over the baseline full search in Chapter 3 for both morpheme and grapheme systems. For example, with the morpheme system, the MTWV on IV keywords improved from 0.4187 to 0.4750 while the MTWV for OOV keywords improved from 0.3569 to 0.4440.

The second and the third approaches use different types of information, hence it would be beneficial to combine them together. One straightforward strategy, which is used in the thesis, is to use the re-ranking score produced by the second approach, i.e. the RAE-SG method, as one of the feature for the binary classifier described in the third proposed approach. This strategy was realized in the SYSTEM 4 in Table 5.3 and it significantly outperforms the baseline system which does not use any re-ranking method (i.e. SYSTEM 1 in Table 5.3). On the word-based system, the MTWV performance improved from 0.5377 to 0.5974, and the MAP score improved from 0.5999 to 0.6429.

We note that it is difficult to apply the third proposed approach on top of the results of the first approach. Specifically, in the first proposed approach, we split the subword sequence into several sub-sequences, then allow some sub-sequences of the keyword to be miss detected. As a result, the timing information for each subword unit of the query is missing. Such the timing information is crucial for the third proposed approach to extract the list of competing hypotheses. Therefore, it is not straightforward to combine the two approaches.

6.2 Future work

One of the major issues of the re-scoring framework in Chapter 4 is how to generate high quality exemplars for unseen keywords. Although the concatenation technique (see Section 4.3) has produced effective samples for re-scoring, it might not be optimum due to its inherent limitations, e.g. data sparsity. One potential direction is to examine deep learning techniques that have been successfully applied to Text-to-Speech research. For

instance, recently, Generative Adversary Network [175] has been studied on various tasks to generate synthetic samples that are very similar to natural ones. Another research direction is the joint optimization of acoustic similarity and re-ranking algorithms for the KWS performance measure. Currently, these are separately optimized from the final STD performance measure such as MTWV and MAP. It could be beneficial to optimize the similarity together with the re-ranking algorithms themselves to maximize the MTWV and MAP. This is similar to feature learning for graph-based algorithms which has become popular in the machine learning community [176].

As mentioned in Chapter 5, numerous features have been examined to train a re-scoring model. However, this strategy have several issues:

Figure 6.1: A block diagram of the end-to-end re-scoring model.

- Extracting engineering features is expensive and might require expert knowledge.
- Features might not be independent from each others, which result in less efficiency when combining them together.
- It might not be adequate to extract features from ASR output since some potentially useful information, such as the temporal structure of the signal, might be lost when speech signals are decoded using a standard ASR [48].

As a result, it is desirable to examine end-to-end approaches for re-scoring hypothesized detections. An end-to-end re-scoring model could directly take the textual keyword and speech signal at the detected location as the inputs and produce the decision whether the keyword appears at the detected location. It can be demonstrated in Figure 6.1. Specifically, we might use a Long Short Term Memory (LSTM) [177] network to encode the speech signal at detected location into a vector representation, and use a Convolutional Neural Network (CNN) [178] to encode the character sequence of the text query into another vector; finally, these two vectors are treated as the input to a feedforward network to make “YES/NO” decision.

List of Publications

- (i) **V. T. Pham**, H. Xu, X. Xiao, N. F. Chen, E. S. Chng, H. Li, “Re-ranking Spoken Term Detection with Acoustic Exemplars of Keywords”, in *Speech Communication*, Vol 104, November 2018, pp. 12-23.
- (ii) **V.T. Pham**, H. Xu, N. F. Chen, S. Sivadas, B. P. Lim, E. S. Chng, H. Li, “Discriminative score normalization for keyword search decision”, in *Proc. of ICASSP 2014*, pp. 7128-7132.
- (iii) **V.T. Pham**, N. F. Chen, S. Sivadas, H. Xu, I. F. Chen, C. Ni, E. S. Chng, H. Li, “System and keyword dependent fusion for spoken term detection”, in *Proc. of SLT 2014*, pp. 430-435.
- (iv) **V. T. Pham**, H. Xu, T. Y. Chong, X. Xiao, E. S. Chng, and H. Li, “On the study of very low-resource language keyword search,” in *Proc. of APSIPA 2015*, pp. 358-364.
- (v) **V. T. Pham**, H. Xu, X. Xiao, N. F. Chen, E. S. Chng, H. Li, “Keyword search using query expansion for graph-based rescoring of hypothesized detections”, in *Proc. of ICASSP, 2016*, pp. 6035 - 6039.
- (vi) **V. T. Pham**, H. Xu, X. Xiao, N. F. Chen, E. S. Chng, H. Li, “Rescoring hypothesized detections of Out-of-Vocabulary keywords using subword samples”, in *Proc. Interspeech 2016*, pp. 933-937.
- (vii) **V. T. Pham**, H. Xu, X. Xiao, N. F. Chen, E. S. Chng “Pruning strategies for partial search in Spoken Term Detection”, In *Proc. of SoICT 2017*, pp. 114-119.

References

- [1] C. Chelba, T. J. Hazen, and M. Saraclar, “Retrieval and browsing of spoken content,” *IEEE Signal Processing Magazine*, vol. 25, no. 3, pp. 39–49, 2008.
- [2] P. Yu and F. Seide, “A hybrid word/phoneme-based approach for improved vocabulary-independent search in spontaneous speech,” in *Proc. of Interspeech*, pp. 293–296, 2004.
- [3] G. Chen, O. Yilmaz, J. Trmal, D. Povey, and S. Khudanpur, “Using proxies for OOV keywords in the Keyword Search task,” in *Proc. of ASRU*, pp. 416–421, 2013.
- [4] I. Szke, M. Fapo, L. Burget, and J. Cernock, “Hybrid word-subword decoding for Spoken Term Detection,” in *Proc. of SIGIR*, pp. 42–48, 2008.
- [5] J. Rohlicek, W. Russell, S. Roukos, and H. Gish, “Continuous Hidden Markov modeling for speaker-independent word spotting,” in *Proc. of ICASSP*, pp. 627–630, 1989.
- [6] J. Garofolo, G. Auzanne, and E. Voorhees, “The TREC Spoken Document Retrieval track: A success story,” in *Proc. of TREC-9*, pp. 416–420, 2000.
- [7] NIST, <https://catalog.ldc.upenn.edu/docs/LDC2011S02/std06-evalplan-v10.pdf>, *The Spoken Term Detection (STD) 2006 Evaluation Plan*, 2006.
- [8] R. Rose, *Word Spotting from Continuous Speech Utterances*, ch. Automatic Speech and Speaker Recognition: Advanced Topics. Kluwer Academic, 1995.
- [9] R. Rose and D. Paul, “A hidden markov model based keyword recognition system,” in *Proc. of ICASSP*, pp. 129–132, 1990.

REFERENCES

- [10] I. Szke, P. Schwarz, P. Matjka, and M. Karafit, “Comparison of keyword spotting approaches for informal continuous speech,” in *Prof. of Eurospeech*, pp. 633 – 636, 2005.
- [11] R. A. Sukkar, A. R. Setlur, M. G. Rahim, and C.-H. Lee, “Utterance verification of keyword strings using word-based minimum verification error (wb-mve) training,” in *Proc. of ICASSP*, pp. 518–521, 1996.
- [12] E. D. Sandness and I. Lee Hetherington, “Keyword-based discriminative training of acoustic models,” in *Proc. of ICSLP*, pp. 135–138, 2000.
- [13] J. Keshet, D. Grangier, and S. Bengio, “Discriminative Keyword Spotting,” *Speech Communication*, vol. 51, no. 4, pp. 317–329, 2009.
- [14] A. Jansen and P. Niyogi, “Point process models for spotting keywords in continuous speech,” *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 17, pp. 1457–1470, Nov 2009.
- [15] M. Lehtonen, “Hierarchical approach for spotting keywords,” Idiap-RR Idiap-RR-41-2005, IDIAP, 2005.
- [16] K. Kintzley, A. Jansen, and H. Hermansky, “Event selection from phone posteriorgrams using matched filters,” in *Proc. of Interspeech*, 1905.
- [17] P. Schuble and M. Wechsler, “First experiences with a system for content based retrieval of information from speech recordings,” in *IJCAI-95 Workshop on Intelligent Multimedia Information Retrieval*, pp. 59–69, 1995.
- [18] C. Tee, S. Khe, H. Li, and H. Ng, “Statistical Lattice-Based Spoken Document Retrieval,” *ACM Transactions on Information Systems*, vol. 28, no. 2, pp. 1–30, 2010.
- [19] J. Mamou, D. Carmel, and R. Hoory, “Spoken Document Retrieval from call-center conversation,” in *Prof. of SIGIR*, pp. 51–58, 2006.
- [20] T. Hazen, W. Shen, and C. White, “Query-By-Example Spoken Term Detection using phonetic posteriorgram templates,” in *Proc. of ASRU*, pp. 421–426, 2009.

- [21] Y. Zhang, *Unsupervised Speech Processing with Applications to Query-by-Example Spoken Term Detection*. PhD thesis, Massachusetts Institute of Technology, 2013.
- [22] A. Muscariello, G. Gravier, and F. Bimbot, “Zero-resource audio-only Spoken Term Detection based on a combination of template matching techniques,” in *Proc. of Interspeech*, 2011.
- [23] A. Jansen and B. Durme, “Indexing raw acoustic features for scalable zero resource search,” in *Proc. of Interspeech*, pp. 2466–2469, 2012.
- [24] C. Parada, A. Sethy, and B. Ramabhadran, “Query-by-example spoken term detection for oov terms,” in *Proc. of ASRU*, pp. 404–409, 2009.
- [25] W. Shen, C. M. White, and T. J. Hazen, “A comparison of query-by-example methods for spoken term detection,” in *Proc. of Interspeech*, 2009.
- [26] J. S. Bridle, “An efficient elastic-template method for detecting given words in running speech,” in *Proc. of the British Acoustic Society Meeting*, pp. 1–4, 1973.
- [27] P. Mermelstein, “Distance measures for speech recognition—psychological and instrumental,” in *Joint Workshop on Pattern Recognition and Artificial Intelligence*, pp. 374–388, 1976.
- [28] S. Davis and P. Mermelstein, “Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences,” *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 28, pp. 357–366, August 1980.
- [29] J. S. Bridle and M. D. Brown, “An experimental automatic word recognition system,” tech. rep., Joint Speech Research Unit, Ruislip, England, 1974.
- [30] Y. Zhang and J. R. Glass, “Unsupervised spoken keyword spotting via segmental dtw on gaussian posteriorgrams,” in *Proc. of ASRU*, 2009.
- [31] H. Wang, T. Lee, and C. Leung, “Unsupervised spoken term detection with acoustic segment model,” in *Proc. of Speech Database and Assessments*, pp. 106–111, 2011.

REFERENCES

- [32] T. Ezzat and T. Poggio, “Discriminative word-spotting using ordered spectro-temporal patch features,” in *Proc. of SAPA Workshop*, pp. 35–40, 2008.
- [33] S. Barnwal, K. Sahni, R. Singh, and B. Raj, “Spectrographic seam patterns for discriminative word spotting,” in *Proc. of ICASSP*, pp. 4725–4728, 2012.
- [34] F. Grézl, M. Karafiát, S. Kontar, and J. Cernocký, “Probabilistic and bottle-neck features for LVCSR of meetings,” in *Proc. of ICASSP*, pp. 757–760, 2007.
- [35] F. Grezl and P. Fousek, “Optimizing bottle-neck features for LVCSR,” in *Proc. of ICASSP*, pp. 4729–4732, 2008.
- [36] G. Evermann and P. Woodland, “Large vocabulary decoding and confidence estimation using word posterior probabilities,” in *Proc. of ICASSP*, pp. 1655–1658, 2000.
- [37] F. Wessel, R. Schluter, K. Macherey, and H. Ney, “Confidence measures for large vocabulary continuous speech recognition,” *IEEE Transactions on Speech and Audio Processing*, vol. 9, no. 3, pp. 288–298, 2001.
- [38] M. S. Seigel, *Confidence estimation for Automatic Speech Recognition Hypotheses*. PhD thesis, University of Cambridge, 2013.
- [39] Z. Lv, M. Cai, and J. L. W.-Q. Zhang, “A novel discriminative score calibration method for Keyword Search,” in *Proc. of Interspeech*, pp. 745–749, 2016.
- [40] D. Wang, J. Tejedor, S. King, and J. Frankel, “Term-dependent confidence normalisation for Out-of-Vocabulary Spoken Term Detection,” *Journal of Computer Science and Technology*, vol. 27, no. 2, pp. 358–375, 2012.
- [41] D. Miller, M. Kleber, C. Kao, O. Kimball, T. Colthurst, S. Lowe, R. Schwartz, and H. Gish, “Rapid and accurate Spoken Term Detection,” in *Proc. of Interspeech*, pp. 314–317, 2007.
- [42] D. Karakos and R. Schwartz, “Score normalization,” tech. rep., Babel PI meeting, 2013.

- [43] J. Mamou, J. Cui, X. Cui, M. Gales, B. Kingsbury, K. Knill, L. Mangu, D. Nolden, M. Picheny, B. Ramabhadran, R. Schluter, A. Sethy, and P. Woodland, “System combination and score normalization for Spoken Term Detection,” in *Proc. of ICASSP*, pp. 8272–8276, 2013.
- [44] Y.-N. Chen, C.-P. Chen, H.-Y. Lee, C.-A. Chan, and L.-s. Lee, “Improved spoken term detection with graph-based re-ranking in feature space,” in *Proc. of ICASSP*, pp. 5644–5647, 2011.
- [45] 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. of ICASSP*, pp. 2479–2483, 2013.
- [46] A. Norouzi, R. C. Rose, S. H. Ghahghah, and A. Jansen, “Zero resource graph-based confidence estimation for open vocabulary Spoken Term Detection,” in *Proc. of ICASSP*, pp. 8292–8296, 2013.
- [47] H.-Y. Lee, P.-W. Chu, and L.-s. Lee, “Improved open-vocabulary spoken content retrieval with word and subword lattices using acoustic feature similarity,” *Computer Speech and Language*, vol. 28, no. 5, pp. 1045–1065, 2014.
- [48] L.-s. Lee, J. Glass, H.-Y. Lee, and C.-A. Chan, “Spoken content retrieval: Beyond cascading speech recognition with text retrieval,” *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 23, no. 9, pp. 1389–1420, 2015.
- [49] 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. of Interspeech*, pp. 2464–2468, 2014.
- [50] J. Tejedor, D. T. Toledano, D. Wang, S. King, and J. Colas, “Feature analysis for discriminative confidence estimation in Spoken Term Detection,” *Speech Communication*, vol. 28, no. 5, pp. 1083–1114, 2014.
- [51] J. Tejedor, D. T. Toledano, M. Bautista, S. King, D. Wang, and J. Colás, “Augmented set of features for confidence estimation in Spoken Term Detection,” in *Proc. of Interspeech*, pp. 701–704, 2010.

REFERENCES

- [52] D. Schneider and T. Mertens, “Contextual verification for open vocabulary Spoken Term Detection,” in *Proc. of Interspeech*, pp. 697–700, 2010.
- [53] H.-Y. Lee, T.-W. Tu, C.-P. Chen, C.-Y. Huang, and L.-s. Lee, “Improved Spoken Term Detection using Support Vector Machines based on lattice context consistency,” in *Proc. of ICASSP*, pp. 5648–5651, 2011.
- [54] M. Seigel, P. Woodland, and M. Gales, “A confidence-based approach for improving keyword hypothesis scores,” in *Proc. of ICASSP*, pp. 8565–8569, 2013.
- [55] C. Parada, M. Dredze, D. Filimonov, and F. Jelinek, “Contextual information improves OOV detection in speech,” in *Proc. of HLT-NAAC*, pp. 216–224, 2010.
- [56] I. Szke, M. Fapso, M. Karafiat, L. Burget, F. Grezl, P. Schwarz, P. Glembek, O. Matejka, S. Kontar, and J. Cernocky, “BUT system for NIST STD 2006 - English,” in *Proc. of STD*, 2006.
- [57] K. Thambiratnam and S. Sridharan, “Rapid yet accurate speech indexing using Dynamic Match Lattice Spotting,” *IEEE Trans. Speech Audio Process*, vol. 15, no. 1, pp. 346–357, 2007.
- [58] R. Wallace, R. Vogt, and S. Sridharan, “A Phonetic Search approach to the 2006 NIST Spoken Term Detection Evaluation,” in *Proc. of Interspeech*, pp. 2385–2388, 2007.
- [59] R. Wallace, R. Vogt, and S. Sridharan, “Spoken Term Detection using fast phonetic decoding,” in *Proc. of ICASSP*, pp. 4881–4884, 2009.
- [60] J. Mamou, B. Ramabhadran, and O. Siohan, “Vocabulary independent Spoken Term Detection,” in *Proc. of SIGIR*, pp. 615–622, 2007.
- [61] A. Nozouzian and R. Rose, “An efficient approach for two-stage open vocabulary Spoken Term Detection,” in *Proc. of SLT*, pp. 50–62, 2010.
- [62] Y.-C. Pan and L.-s. Lee, “Performance analysis for lattice-based speech indexing approaches using words and subword units,” in *IEEE Transactions on Audio, Speech, and Language Processing*, pp. 1562–1574, IEEE, 2010.

- [63] T. Mertens and D. Schneider, “Efficient subword lattice retrieval for German Spoken Term Detection,” in *Proc. of ICASSP*, pp. 4885–4888, 2009.
- [64] K. Iwami, Y. Fujii, K. Yamamoto, and S. Nakagawa, “Efficient Out-Of-Vocabulary term detection by n-gram array indices with distance from a syllable lattice,” in *Proc. of ICASSP*, pp. 5664–5667, 2011.
- [65] V. Turunen and M. Kurimo, “Indexing Confusion Networks for Morph-based Spoken Document Retrieval,” in *Proc. of SIGIR*, pp. 631–638, 2007.
- [66] P. Karanasou, L. Burget, D. Vergyri, M. Akbacak, and A. M., “Discriminatively trained phoneme confusion model for Keyword Spotting,” in *Proc. of Interspeech*, pp. 2434–2437, 2012.
- [67] M. Saraclar, A. Sethy, B. Ramabhadran, L. Mangu, J. Cui, X. Cui, B. Kingsbury, and J. Mamou, “An empirical study of confusion modeling in Keyword Search for low resource languages,” in *Proc. of ASRU*, pp. 464–469, 2013.
- [68] U. Chaudhari and M. Picheny, “Improvements in phone based audio search via constrained match with high order confusion estimates,” in *Proc. of ASRU*, pp. 665–670, 2007.
- [69] O. Vinyals and S. Wegmann, “Chasing the metric: Smoothing learning algorithms for keyword detection,” in *Proc. of ICASSP*, pp. 3301–3305, 2014.
- [70] C. Benitez, A. J. Rubio, P. Garcia, and A. de la Torre, “Different confidence measures for word verification in speech recognition,” *Speech Communication*, vol. 32, pp. 79–94, 2000.
- [71] V. T. Pham, H. Xu, N. F. Chen, S. Sivadas, B. P. Lim, E. S. Chng, and H. Li, “Discriminative score normalization for keyword search decision,” in *Proc. of ICASSP*, pp. 7078–7082, 2014.
- [72] H.-Y. Lee and L.-s. Lee, “Enhanced Spoken Term Detection using Support Vector Machines and weighted pseudo examples,” *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 21, no. 6, pp. 1272–1284, 2013.

REFERENCES

- [73] NIST, “The Open Keyword Search (KWS) 2014 Evaluation,” in <http://www.nist.gov/itl/iad/mig/upload/KWS14-evalplan-v11.pdf>, 2014.
- [74] S. Davis and P. Mermelstein, “Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences,” *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 28, no. 2, pp. 357–366, 1980.
- [75] H. Hermansky, “Perceptual linear predictive (PLP) analysis of speech,” *Journal of the Acoustical Society of America*, vol. 87, no. 4, pp. 1738–52, 1990.
- [76] C. M. Bishop, *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 2006.
- [77] A. Mohamed, G. Dahl, and G. Hinton, “Deep Belief Networks for phone recognition,” in *Proc. of NIPS*, 2009.
- [78] G. E. Dahl, M. Ranzato, A. rahman Mohamed, and G. E. Hinton, “Phone recognition with the Mean-Covariance Restricted Boltzmann Machine,” in *Proc. of NIPS*, pp. 469–477, 2010.
- [79] G. Hinton, L. Deng, D. Yu, G. Dahl, A.-r. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. Sainath, and B. Kingsbury, “Deep Neural Networks for acoustic modeling in speech recognition: The shared views of four research groups,” *IEEE Signal Processing Magazine*, vol. 29, no. 6, pp. 82–97, 2012.
- [80] R. Rosenfeld, “Two decades of statistical language modeling: where do we go from here?,” *Proceedings of the IEEE*, vol. 88, pp. 1270–1278, Aug. 2000.
- [81] G. D. Forney, “The Viterbi algorithm,” in *Proceedings of the IEEE*, pp. 268–278, 1973.
- [82] L. R. Rabiner, “Readings in speech recognition,” pp. 267–296, Morgan Kaufmann Publishers Inc., 1990.
- [83] S. Young, N. Russell, and J. Thornton, “Token passing: a simple conceptual model for connected speech recognition systems,” tech. rep., 1989.

- [84] C. Chelba and J. Silva, “Soft indexing of speech content for search in spoken documents,” *Computer Speech and Language*, vol. 21, no. 3, pp. 458–478, 2007.
- [85] F. Seide, P. Yu, and Y. Shi, “Towards Spoken Document Retrieval for the enterprise: Approximate word-lattice indexing with text indexers,” in *Proc. of ASRU*, pp. 629–634, 2007.
- [86] P. Yu, K. Thambiratnam, and F. Seide, “Word-lattice based spoken-document indexing with standard text indexers,” in *Proc. of SLT*, pp. 293–296, 2008.
- [87] L. Mangu, E. Brill, and A. Stolcke, “Finding consensus in speech recognition: Word error minimization and other applications of confusion networks,” *Computer Speech and Language*, vol. 14, pp. 373–400, 10 2000.
- [88] D. Hakkani-Tur and G. Riccardi, “A general algorithm for word graph matrix decomposition,” in *Proc. of ICASSP*, pp. I–I, 2003.
- [89] K. Katsurada, S. Teshima, and T. Nitta, “Fast keyword detection using suffix array,” in *Proc. of Interspeech*, pp. 909–912, 2009.
- [90] U. Chaudhari and M. Picheny, “Improved vocabulary independent search with approximate match based on Conditional Random Fields,” in *Proc. of ASRU*, pp. 416–420, 2009.
- [91] J. Mamou and B. Ramabhadran, “Phonetic query expansion for Spoken Document Retrieval,” in *Proc. of Interspeech*, pp. 2106–2109, 2008.
- [92] M. Saraclar and R. Sproat, “Lattice-based search for Spoken Utterance Retrieval,” in *Proc. of HLT-NAACL*, pp. 129–136, 2004.
- [93] J. Cernocky, I. Szoke, M. Fapso, M. Karafiat, L. Burget, J. Kopecky, F. Grezl, P. Schwarz, O. Glembek, I. Oparin, P. Smrz, and P. Matejka, “Search in speech for public security and defense,” in *Proc. of IEEE Workshop on Signal Processing Applications for Public Security and Forensics*, pp. 1–7, 2007.
- [94] I. Szoke, J. Cernock, L. Burget, and M. Fapo, “Sub-word modeling of out of vocabulary words in Spoken Term Detection,” in *Proc. of SLT*, pp. 273–276, 2008.

REFERENCES

- [95] J. Scott Olsson, J. Wintrobe, and M. Lee, “Fast unconstrained audio search in numerous human languages,” in *Proc. of ICASSP*, pp. 77–80, 2007.
- [96] O. Siohan and M. Bacchiani, “Fast vocabulary-independent audio search using path-based graph indexing,” in *Proc. of Interspeech*, pp. 53–56, 2005.
- [97] D. Can and M. Saraclar, “Lattice indexing for Spoken Term Detection,” *IEEE Trans. Speech Audio Process*, vol. 19, no. 8, pp. 2338–2347, 2011.
- [98] C. Allauzen, M. Mohri, and M. Saraclar, “General indexation of weighted automata - Application to spoken utterance retrieval,” in *Proc. of HLT-NAACL*, pp. 33–40, 2004.
- [99] S. Parlak and M. Saraclar, “Spoken Term Detection for Turkish broadcast news,” in *Proc. of ICASSP*, pp. 5244–5247, 2008.
- [100] D. Can, E. Cooper, A. Sethy, C. White, B. Ramabhadran, and M. Saraclar, “Effect of pronunciations on OOV queries in Spoken Term Detection,” in *Proc. of ICASSP*, pp. 3957–3960, 2009.
- [101] B. Logan, J. Van Thong, and P. Moreno, “Approaches to reduce the effects of OOV queries on indexed spoken audio,” *IEEE Trans. Speech Audio Process*, vol. 7, no. 5, pp. 899–906, 2005.
- [102] M. Saraclar, A. Sethy, B. Ramabhadran, L. Mangu, J. Cui, X. Cui, B. Kingsbury, and J. Mamou, “An empirical study of confusion modeling in Keyword Search for low resource languages,” in *Proc. of ASRU*, pp. 464–469, 2002.
- [103] D. Povey, A. Ghoshal, G. Boulianne, L. Burget, O. Glembek, N. Goel, M. Hannemann, P. Motlicek, Y. Qian, P. Schwarz, J. Silovsky, G. Stemmer, and K. Vesely, “The Kaldi speech recognition toolkit,” in *Proc. of ASRU*, pp. 1–4, 2011.
- [104] D. Wang, *Fast and Accurate Phonetic Spoken Term Detection*. PhD thesis, University of Edinburgh, 2010.
- [105] S. Dharanipragada and S. Roukos, “A multistage algorithm for spotting new words in speech,” *IEEE Trans. Speech Audio Process*, vol. 10, no. 8, pp. 542–555, 2002.

REFERENCES

- [106] D. Wang, S. King, J. Frankel, and P. Bell, “Term-Dependent Confidence for Out-of-Vocabulary Term Detection,” in *Proc. of Interspeech*, pp. 2139–2142, 2009.
- [107] M. Akbacak, D. Vergyri, and A. Stolcke, “Open-vocabulary Spoken Term Detection using grapheme-based hybrid recognition systems,” in *Proc. of ICASSP*, pp. 5240–5243, 2008.
- [108] A. Rastrow, A. Sethy, and B. Ramabhadran, “A new method for OOV detection using hybrid word/fragment system,” in *Proc. of ICASSP*, pp. 3953–3956, 2009.
- [109] B. Ramabhadran, A. Sethy, J. Mamou, B. Kingsbury, and U. Chaudhari, “Fast decoding for open vocabulary Spoken Term Detection,” in *Proc. of NAACL-HLT*, pp. 277–280, 2009.
- [110] T. Hori, I. Hetherington, T. Hazen, and J. Glass, “Open-vocabulary Spoken Utterance Retrieval using Confusion Networks,” in *Proc. of ICASSP*, pp. 73–76, 2007.
- [111] M. Akbacak, L. Burget, W. Wang, and J. Hout, “Rich system combination for Keyword Spotting in noisy and acoustically heterogeneous audio stream,” in *Proc. of ICASSP*, pp. 8267–8271, 2013.
- [112] L. Mangu, H. Soltau, and H. Kuo, “Exploiting diversity for Spoken Term Detection,” in *Proc. of ICASSP*, pp. 8282–8286, 2013.
- [113] P. Motlicek, F. Valente, and I. Szoke, “Improve acoustic based Keyword Spotting using LVCSR lattices,” in *Proc. of ICASSP*, pp. 4413–4416, 2012.
- [114] A. Mandal, K. R. Prasanna Kumar, and P. Mitra, “Recent developments in spoken term detection: A survey,” *Int. J. Speech Technol.*, vol. 17, pp. 183–198, June 2014.
- [115] The Guardian, <https://www.theguardian.com/media/mind-your-language/2016/feb/04/english-neologisms-new-words>, 2016.
- [116] K. Ng and V. W. Zue, “Subword-based approaches for spoken document retrieval,” *Speech Communication*, vol. 32, no. 3, pp. 157 – 186, 2000.

REFERENCES

- [117] F. Seide and P. Yu, “Vocabulary-independent search in spontaneous speech,” in *Proc. of ICASSP*, pp. 253–256, 2004.
- [118] S. Deligne and F. Bimbot, “Language modeling by variable length sequences: theoretical formulation and evaluation of multigrams,” in *Proc. of ICASSP*, pp. 169–172, 1995.
- [119] G. F. Choueiter, *Linguistically-Motivated Sub-word Modeling with Applications to Speech Recognition*. PhD thesis, Massachusetts Institute of Technology, 2010.
- [120] K. Ng, *Subword-based approaches for Spoken Document Retrieval*. PhD thesis, MIT, 2000.
- [121] J. Mamou, B. Ramabhadran, Y. Mass, and B. Sznajder, “Combination of multiple speech transcription methods for vocabulary independent search,” in *Proc. of SIGIR*, pp. 20–27, 2008.
- [122] N. Chen, C. Ni, S. Sivadas, V. T. Pham, H. Xu, X. Xiao, T. S. Lau, S. J. Leow, B. P. Lim, C. C. Leung, L. Wang, C. H. Lee, A. Goh, E. S. Chng, B. Ma, and H. Li, “Low-resource Keyword Search strategies for Tamil,” in *Proc. of ICASSP*, pp. 5366–5370, 2015.
- [123] M. Creutz and K. Lagus, “Unsupervised discovery of morphemes,” in *Proc. of the Workshop on Morphological and Phonological Learning of ACL-02*, pp. 21–30, 2002.
- [124] W. Hartmann, V.-B. Le, A. Messaoudi, L. Lamel, and J. luc Gauvain, “Comparing decoding strategies for subword-based keyword spotting in low-resourced languages,” in *Proc. of Interspeech*, pp. 2764 – 2768, 2014.
- [125] K. Audhkhasi and A. Verma, “Keyword search using modified minimum edit distance measure,” in *Proc. of ICASSP*, pp. 929 – 932, 2007.
- [126] Y. Itoh, T. Otake, K. Iwata, K. Kojima, M. Ishigame, K. Tanaka, and S. Lee, “Two-stage vocabulary-free Spoken Document Retrieval - subword identification and re-recognition of the identified sections,” in *Prof. of Interspeech*, pp. 1161–1164, 2006.

- [127] K. Iwata, K. Shinoda, and S. Furui, “Robust Spoken Term Detection using combination of phone-based and word-based recognition,” in *Proc. of Interspeech*, pp. 2195 – 2198, 2008.
- [128] C. Parada, A. Sethy, and B. Ramabhadran, “Balancing false alarms and hits in Spoken Term Detection,” in *Proc. of ICASSP*, pp. 5286–5289, 2010.
- [129] T. Mertens, D. Schneider, and J. Köhler, “Merging search spaces for subword Spoken Term Detection,” in *Proc. of Interspeech*, pp. 2127–2130, 2009.
- [130] L. E. Baum and J. A. Eagon, “An inequality with applications to statistical estimation for probabilistic functions of Markov processes and to a model for ecology,” *Bull. Am. Math. Soc.*, vol. 73, no. 3, pp. 360–363, 1967.
- [131] D. K. et.al., “Score normalization and system combination for improved keyword spotting,” in *Proc. of ASRU*, pp. 210–215, IEEE, 2013.
- [132] S. Goldwater, D. Jurafsky, and C. D. Manning, “Which words are hard to recognize? prosodic, lexical, and disfluency factors that increase speech recognition error rates,” *Speech Communication*, vol. 52, no. 3, pp. 181–200, 2010.
- [133] N. Kanda, R. Takeda, and Y. Obuchi, “Using rhythmic features for Japanese Spoken Term Detection,” in *Proc. of SLT*, pp. 170–175, 2012.
- [134] H. Li and J. Han, “A novel confidence measure based on context consistency for Spoken Term Detection,” in *Proc. of Interspeech*, pp. 2430–2433, 2012.
- [135] J. Richards, M. Ma, and A. Rosenberg, “Using word burst analysis to rescore Keyword Search candidates on low-resource languages,” in *Proc. of ICASSP*, pp. 7824–7828, 2014.
- [136] 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. of Interspeech*, pp. 2769–2773, 2014.

REFERENCES

- [137] D. Vergyrim, I. Shafran, A. Stolcke, R. Gadde, M. Akbacak, B. Roark, and W. Wang, “The SRI/OGI 2006 Spoken Term Detection system,” in *Proc. of Interspeech*, pp. 2393–2396, 2007.
- [138] H. Su, V. T. Pham, Y. He, and J. Hieronymus, “Improvements on transducing syllable lattices to word lattice for Keyword Search,” in *Proc. of ICASSP*, pp. 4729–4733, 2015.
- [139] H. Xu, J. Hou, X. Xiao, V. T. Pham, C. C. Leung, L. Wang, V. H. Do, H. Lv, L. Xie, B. Ma, E. S. Chng, and H. Li, “Approximate search of audio queries by using dtw with phone time boundary and data augmentation,” in *Proc. of ICASSP*, pp. 6030–6034, 2016.
- [140] J. Hou *et al.*, “The NNI Query-by-Example System for MediaEval 2015,” in *MediaEval 2015 Workshop, Wurzen, Germany*, 2015.
- [141] L. Wang, E. S. Chng, and H. Li, “Efficient sparse self-similarity matrix construction for repeating sequence detection,” in *Proc. of IEEE International Conference on Multimedia and Expo*, pp. 458–461, 2009.
- [142] C. Liu and D. Wang, “N-gram FST indexing for Spoken Term Detection,” in *Proc. of Interspeech*, pp. 2093–2096, 2012.
- [143] J. Cernocky, L. Burget, P. Schwarz, P. Matejka, M. Karafiat, O. Glembek, J. Kopecky, I. Szoke, M. Fapso, F. Grezl, V. Hubeika, and I. Oparin, “Search in speech, language identification and speaker recognition in Speech@FIT,” in *Proc. of 17th International Conference Radio elektronika*, pp. 1–6, 2007.
- [144] N. F. Chen, S. Sivadas, B. P. Lim, H. G. Ngo, H. Xu, V. T. Pham, B. Ma, and H. Li, “Strategies for Vietnamese Keyword Search,” in *Proc. of ICASSP*, pp. 4121–4125, 2014.
- [145] A. Stolcke, “SRILM -An extensible language modeling toolkit,” in *Proc. of Interspeech*, pp. 901–904, 2002.

REFERENCES

- [146] C. Manning, P. Raghavan, and H. Schtze, *An introduction to Information Retrieval*. Cambridge University Press., 2008.
- [147] A. Viterbi, “Error bounds for convolutional codes and an asymptotically optimum decoding algorithm,” *IEEE Transactions on Information Theory*, vol. 13, no. 2, pp. 260–269, 1967.
- [148] J.-h. Yang, Z.-w. Zhao, Y. Jiang, G.-p. Hu, and X.-r. Wu, “Multi-tier non-uniform unit selection for corpus-based speech synthesis,” in *Proc. of Blizzard Challenge*, 2006.
- [149] L. Latacz, Y. O. Kong, and W. Verhelst, “Unit selection synthesis using long non-uniform units and phoneme identity matching,” in *Proc. of the 6th ISCA Workshop on Speech Synthesis*, pp. 270–275, 2007.
- [150] A. J. Hunt and A. W. Black, “Unit selection in a concatenative speech synthesis system using a large speech database,” in *Proc. of ICASSP*, vol. 1, pp. 373–376, 1996.
- [151] Y. Takefuta, E. G. Jancosek, and M. Brunt, “A statistical analysis of melody curves in the intonation of american english,” in *Proc. of the 7th International Congress of Phonetic Sciences*, pp. 233–241.
- [152] M. Gordon and P. Ladefoged, “Phonation types: a cross-linguistic overview,” *Journal of Phonetics*, vol. 29, no. 4, pp. 383 – 406, 2001.
- [153] J. Han, M. Kamber, and J. Pei, *Data Mining: Concepts and Techniques*. New York, NY, USA: CMorgan Kaufmann, 2011.
- [154] S. Pigeon, P. Druytza, and P. Verlinde, “Applying Logistic Regression to the fusion of the NIST’99 1-speaker submissions,” *Digital Signal Processing*, vol. 10, pp. 237–248, Jan 2000.
- [155] G. S. Morrison, “Tutorial on Logistic Regression calibration and fusion:converting a score to a likelihood ratio,” *Australian Journal of Forensic Sciences*, vol. 45, pp. 173–197, June 2013.

REFERENCES

- [156] A. Park and J. R. Glass, “Towards unsupervised pattern discovery in speech,” in *Proc. of ASRU*, pp. 53–58, 2005.
- [157] H.-Y. Lee and L.-s. Lee, “Improved semantic retrieval of spoken content by document/query expansion with random walk over acoustic similarity graphs,” in *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 22, pp. 80–94, 2013.
- [158] A. N. Langville and C. D. Meyer, “A survey of eigenvector methods for web information retrieval,” *SIAM Rev.*, vol. 47, no. 1, pp. 135–161, 2005.
- [159] Y. Stylianou and A. K. Syrdal, “Perceptual and objective detection of discontinuities in concatenative speech synthesis,” in *Proc. of ICASSP*, pp. 837–840, 2001.
- [160] A. Lozano-Diez, A. Silnova, P. Matejka, O. Glembek, O. Plchot, J. Pesan, L. Burget, and J. Gonzalez-Rodriguez, “Analysis and optimization of bottleneck features for speaker recognition,” in *Proc. of Odyssey*, pp. 352–357, 2016.
- [161] Y. Liu, T. Fu, Y. Fan, Y. Qian, and K. Yu, “Speaker verification with deep features,” in *Proc. of IJCNN*, pp. 747–753, 2014.
- [162] Z. Zhang, L. Wang, and A. Kai, “Distant-talking speaker identification by generalized spectral subtraction-based dereverberation and its efficient computation,” *EURASIP Journal on Audio, Speech, and Music Processing*, vol. 15, no. 1, pp. 1–12, 2014.
- [163] V. Gupta, J. Ajmera, A. Kumar, and A. Verma, “A language independent approach to audio search,” in *Proc. of Interspeech*, pp. 1125–1128, August 2011.
- [164] Y. Yuan, C. Leung, L. Xie, H. Chen, B. Ma, and H. Li, “Pairwise learning using multi-lingual bottleneck features for low-resource Query-by-Example Spoken Term Detection,” in *Proceeding of ICASSP*, pp. 5645–5649, 2017.
- [165] I. Szoke, M. Skacel, L. Burget, and J. Cernocky, “Coping with channel mismatch in Query-by-Example - BUT QUESST 2014,” in *Proc. of ICASSP*, pp. 5838–5842, 2015.

REFERENCES

- [166] V. T. Pham, H. Xu, T. Y. Chong, X. Xiao, E. S. Chng, , and H. Li, “On the study of very low-resource language Keyword Search,” in *Proc. of APSIPA*, pp. 358–364, 2015.
- [167] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016.
- [168] J. Ajmera and F. Metze, “Keyword spotting using durational entropy,” in *Proc. of ICASSP*, pp. 973–976, 2007.
- [169] H. Mann and D. Whitney, “On a test of whether one of two random variables is stochastically larger than the other,” *Annals of Mathematical Statistics*, vol. 18, no. 1, pp. 50–60, 1947.
- [170] T. H. Nguyen, E. S. Chng, and H. Li, “T-test distance and clustering criterion for speaker diarization,” in *Proc. of Interspeech*, pp. 36–39, 2008.
- [171] V. T. Pham, H. Xu, X. Xiao, N. F. Chen, and E. S. Chng, “Pruning strategies for partial search in Spoken Term Detection,” in *Proc. of SoICT*, pp. 114–119, 2017.
- [172] V. T. Pham, H. Xu, X. Xiao, N. F. Chen, E. S. Chng, and H. Li, “Keyword Search using query expansion for graph-based rescoring of hypothesized detections,” in *Proc. of CASSP*, pp. 6035–6039, March 2016.
- [173] V. T. Pham, H. Xu, X. Xiao, N. F. Chen, E. S. Chng, and H. Li, “Rescoring hypothesized detections of Out-of-Vocabulary keywords using subword samples,” in *Proc. of Interspeech*, pp. 933–937, 2016.
- [174] V. T. Pham, H. Xu, X. Xiao, N. F. Chen, E. S. Chng, and H. Li, “Re-ranking Spoken Term Detection with acoustic exemplars of keywords,” *Speech Communication*, vol. 104, pp. 12–23, 2018.
- [175] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, “Generative Adversarial Nets,” in *Proc. of NIPS*, pp. 2672–2680, 2014.
- [176] F. Tian, B. Gao, Q. Cui, E. Chen, and T.-Y. Liu, “Learning deep representations for graph clustering,” in *Proceedings of Artificial Intelligence*, pp. 1293–1299, 2014.

REFERENCES

- [177] S. Hochreiter and J. Schmidhuber, “Long short-term memory,” *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.
- [178] X. Zhang, J. Zhao, and Y. LeCun, “Character-level convolutional networks for text classification,” in *Proc. of NIPS*, pp. 649–657, 2015.