PRE-FILTERED DYNAMIC TIME WARPING FOR POSTERIORGRAM BASED KEYWORD SEARCH

by

Gözde Çetinkaya

B.S., Electronics Engineering, Istanbul Kültür University, 2012

Submitted to the Institute for Graduate Studies in
Science and Engineering in partial fulfillment of
the requirements for the degree of
Master of Science

Graduate Program in Electrical and Electronics Engineering
Boğaziçi University

2016

PRE-FILTERED DYNAMIC TIME WARPING FOR POSTERIORGRAM BASED KEYWORD SEARCH

APPROVED BY:

Assoc. Prof. Murat Saraçlar .....
(Thesis Supervisor)

Prof. Levent Arslan .....

Assist. Prof. Arzucan Özgür .....

DATE OF APPROVAL: 11.01.2016

ACKNOWLEDGEMENTS

Foremost, I would like to express the deepest appreciation to my advisor, Assoc. Prof. Murat Saraçlar, for his invaluable guidance, encouragement and excellent advice throughout this thesis. I am delighted that I had the great chance to work with him and look forward to continue working with him in the upcoming years as well. I would like to thank my committee members, Prof. Levent Arslan and Assist. Prof. Arzucan Özgür, for sharing their comments and profound knowledge.

It is a great pleasure to be a member of BUSIM. I would like to thank all BUSIM members. Thanks to Leda for sharing her experiences and kind assistance, Batuhancan for his friendship, support and sincere help, Burcu for providing a fun working environment and keeping up with me.

I also would like to thank my wonderful friends, Berk, Derya, Emre, Engin, Gönenç, Gürkan, İskender, İsmail, Okan, Zahid, for all the delightful moments shared together.

Special thanks to Ozan, for his endless love, patience and generous support in writing this thesis. With his presence, life is more beautiful, troubles are easier, and future is more promising.

Finally, I would like to express my gratitude to my parents, Gül and Safter and to my little brother Orkun, who stand by me with all their hearts in all circumtances. I always feel like the luckiest person in the world to have such a great family. This thesis is dedicated to them.

This study uses the IARPA Babel Program base period language collection release babel105b-v0.4, and is supported by the Intelligence Advanced Research Projects Activity (IARPA) via Department of Defense U.S. Army Research Laboratory (DoD/ARL) contract number W911NF-12-C-0012. The U.S. Government is authorized to repro-

duce and distribute reprints for Governmental purposes notwithstanding any copyright annotation thereon. Disclaimer: The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of IARPA, DoD/ARL, or the U.S. Government.

ABSTRACT

PRE-FILTERED DYNAMIC TIME WARPING FOR POSTERIORGRAM BASED KEYWORD SEARCH

This thesis is motivated by the challange of high computational demand of the dynamic time warping algorithm, which is very prevalent for template based keyword search tasks. The dynamic time warping algorithm suffers from grand calculations of the cost matrix during the search process. To solve this problem, in this thesis, we present a pre-filtering step for the algorithm. We use phonetic posteriorgrams to represent the audio data and generate average posteriorgrams to represent the given text queries. Posteriorgram of the document is divided into segments, or submatrices, and are turned into supervectors, which are used to determine the possible candidates for query matching. Cosine distance is used for the distance measure to find the similarity between the document supervectors and the query supervectors. Filtering is performed according to a threshold value, which specifies the amount of filtering. The other aim is to improve the performance of a large vocabulary continuous speech recognition based keyword search system using the proposed approach for posteriorgram based keyword search. Experimental results indicate that the proposed method reduces the computational complexity of the dynamic time warping algorithm without a significant loss and when combined with the baseline large vocabulary continuous speech recognition based keyword search system, it improves the performance for both the in-vocabulary and the out-of-vocabulary queries.

ÖZET

POSTERİORGRAM TABANLI ANAHTAR SÖZCÜK ARAMA İÇİN ÖN FİLTRELEMELİ DİNAMİK ZAMAN BÜKME ALGORİTMASI

Bu tez, temel anahtar sözcük arama görevlerinde oldukça yaygın kullanılan dinamik zaman bükme algoritmasının bir sorunu olan yüksek hesaplama karmaşıklığıdır. Dinamik zaman bükme, arama sürecinde, toplam uzaklık matrisinin çok fazla hesaplama gerektirir. Tezde, bu sorunu çözmek için bir ön eleme basamağı önerilmiştir. Ses verisinin gösterimi için sesçik posteriorgramı kullanılmış, yazılı sorgunun gösterimi için ortalama posteriorgramlar üretilmiştir. Arama dokümanının posteriorgramı parçalara bölünerek sorgu eşleşmesi için muhtemel adayları saptamakta kullanılacak süper vektörlere dönüştürülmüştür. Doküman süper vektörleri ve sorgu süper vektörleri arasındaki benzerliği bulmak için kosinüs uzaklığı kullanılmıştır. Eleme, filtreleme miktarını belirleyen bir eşik değerine göre yapılır. Diğer amacımız ise önerilen yöntemi kullanarak geniş dağarcıklı sürekli konuşma tanıma tabanlı temel anahtar sözcük arama sisteminin başarımını arttırmaktır. Deney sonuçları önerilen yöntemin dinamik zaman bükme algoritmasının hesaplama karmaşıklığını azalttığını ve sonuçların geniş dağarcıklı sürekli konuşma tanıma tabanlı temel anahtar sözcük arama sistemiyle birleştirildiğinde hem dağarcık içi hem de dağarcık dışı sorgular için başarımı arttırdığını göstermiştir.

TABLE OF CONTENTS

ACKNOWLEDGEMENTS . . . iii
ABSTRACT . . . v
ÖZET . . . vi
LIST OF FIGURES . . . ix
LIST OF TABLES . . . xi
LIST OF SYMBOLS . . . xii
LIST OF ACRONYMS/ABBREVIATIONS . . . xiii
1. INTRODUCTION . . . 1
2. BACKGROUND . . . 4
2.1. Spoken Content Retrieval . . . 4
2.1.1. Spoken Document Retrieval . . . 4
2.1.2. Spoken Utterance Retrieval . . . 5
2.1.3. Spoken Term Detection . . . 5
2.1.4. Evaluation Metrics for Spoken Term Detection . . . 6
2.2. Approaches to Spoken Content Retrieval . . . 9
2.2.1. Subword Based Methods . . . 10
2.2.2. Lattice Based Methods . . . 11
2.2.3. Acoustic Modelling for Spoken Content Retrieval . . . 13
2.2.4. Language Modelling for Spoken Content Retrieval . . . 15
2.2.5. Confusion Modelling for Spoken Content Retrieval . . . 15
2.3. Query by Example Spoken Term Detection . . . 16
2.3.1. Symbol Based Search Methods . . . 16
2.3.2. Template Matching Based Search Methods . . . 17
2.3.3. Query Modelling Methods for Text Based QbE-STD . . . 18
3. METHODOLOGY AND EXPERIMENTS . . . 21
3.1. Dynamic Time Warping . . . 21
3.1.1. Segmental Dynamic Time Warping . . . 23
3.1.2. Subsequence Dynamic Time Warping . . . 25

3.1.3. Speed-up Techniques for Dynamic Time Warping 26
3.2. Pre-filtered Dynamic Time Warping 30
3.2.1. Generation of Supervectors 32
3.2.2. Cosine Similarity and Filtering 32
3.3. Experimental Results 34
3.3.1. Individual Results of the Proposed System 35
3.3.2. Merged Results with the LVCSR Lattice Based System 38
4. CONCLUSIONS AND FUTURE WORK 45
REFERENCES 48

LIST OF FIGURES

Figure 2.1. An example DET curve. 8

Figure 2.2. An example of lattice structure. 12

Figure 2.3. An example of the weighted finite state transducer indexing. . . . 13

Figure 2.4. An example of confusion network structure. 13

Figure 2.5. Scheme of acoustic keyword spotting system. 17

Figure 2.6. General architecure of template matching based search method. . 18

Figure 3.1. A warping between two time series. 21

Figure 3.2. The cost matrix and warping path. 23

Figure 3.3. The segmental dynamic time warping. 24

Figure 3.4. An illustration of the Sakoe-Chiba Band (left) and the Itakura Parallelogram (right). 29

Figure 3.5. An example of the original cost matrix (right) and the lower resolution cost matrix (left). 30

Figure 3.6. Flowchart of the proposed system. 31

Figure 3.7. An illustration of supervector generation. 33

Figure 3.8. MTWV of STO normalized individual system for all, IV and OOV queries. 37

Figure 3.9. MTWV of linear normalized individual system for all, IV and OOV queries. 38

Figure 3.10. MTWV of STO normalized merged system for all, IV and OOV queries. 41

Figure 3.11. MTWV of linear normalized merged system for all, IV and OOV queries. 41

Figure 3.12. DET curve of the baseline and STO normalized merged system. . 42

Figure 3.13. MTWV of all, IV and OOV queries of the merged system for 3 different thresholds. 44

LIST OF TABLES

Table 3.1. Individual MTWV, OTWV and STWV scores of STO normalized experiments with different threshold values. 36

Table 3.2. Individual MTWV, OTWV and STWV scores of linear normalized experiments with different threshold values. 37

Table 3.3. MTWV, OTWV and STWV results for LVCSR lattice based system (L) and merged (L+P) systems with STO normalized experiments. 39

Table 3.4. MTWV, OTWV and STWV results for LVCSR lattice based system (L) and merged (L+P) systems with linear normalized experiments. 40

Table 3.5. Configurations for the experiments. 43

Table 3.6. Merged MTWV, OTWV and STWV results for fixed window approach (L+F), 50% overlap approach (L+O) and base conditions (L+P). 44

LIST OF SYMBOLS

$d(\mathbf{x}_i, \mathbf{y}_j)$	Distance between $\mathbf{x}_i$ and $\mathbf{y}_j$
E	Edit distance
K	Original keyword
K'	Weighted finite state transducer of several in vocabulary words
L_1	Original lexicon
L_2	Lexicon for pronunciation ok K
$\mathcal{L}$	Path length matrix
$\mathcal{L}_o$	Optimal beginning frame matrix
Q	Query posteriorgram
$\mathbf{Q}$	Supervector of the query
$\mathbf{S}_k$	Supervector of the k-th segment
$\mathcal{T}$	Transformation matrix
$\widehat{U}$	Lower-bound approximated posteriorgram
$\mathbf{x}_i$	i-th frame of query posteriorgram
$\bar{x}_i$	i-th element of the reduced sequence
$\mathcal{Y}$	Document posteriorgram
$\mathbf{y}_j$	j-th frame of document posteriorgram
β	TWV parameter
γ	Cumulative distance matrix
δ	Shift amount for submatrices

LIST OF ACRONYMS/ABBREVIATIONS

AKWS	Acoustic Keyword Spotting
ASR	Automatic Speech Recognition
ATWV	Actual Term Weighted Value
BUSIM	Boğaziçi University Signal and Image Processing Lab
CN	Confusion Network
DET	Detection Error Trade-off
DNN	Deep Neural Network
DTW	Dynamic Time Warping
GMM	Gaussian Mixture Model
G2P	Grapheme-to-Phoneme
HMM	Hidden Markov Model
IR	Information Retrieval
IV	In-Vocabulary
LM	Linear Normalization
LSH	Locality Sensitive Hashing
LVCSR	Large Vocabulary Continuous Speech Recognition
MFCC	Mel Frequency Cepstral Coefficients
MAP	Mean Average Precision
MCE	Minimum Classification Error
MPE	Minimum Phone Error
MTWV	Maximum Term Weighted Value
NIST	National Institude of Standards and Technology
OOV	Out-of-Vocabulary
OTWV	Optimal Term Weighted Value
PAA	Piecewise Aggregate Approximation
PSPL	Position Specific Posterior Probability Lattice
QbE-STD	Query By Example Spoken Term Detection
SCR	Spoken Content Retrieval

SDR	Spoken Document Retrieval
sMBR	State Level Minimum Bayes Risk
SR	Speech Rterieval
SSMs	Self Similarity Matrices
STD	Spoken Term Detection
SUR	Spoken Utterance Retrieval
STO	Sum-to-one
STWV	Supremum Term Weighted Value
TREC	Text Retrieval Conferance
TTS	Text-to-Speech
TWV	Term Weighted Value
ROC	Receiver Operating Curve
WER	Word Error Rate
WFST	Weighted Finite State Transducer

1. INTRODUCTION

Today search and retrieval of information have become an inevitable demand as a result of ever increasing multimedia content and ability of cheaper storage. With this perspective, speech retrieval (SR) which combines automatic speech recognition (ASR) and information retrieval (IR) is an important key to easier browsing and retrieving of these multimedia content. In the basic scenario for speech retrieval, the spoken content is transcribed into lattice or text format, then a text retrieval method searches over the ASR output to find requested information. However, in this framework, accuracy and vocabulary size of ASR heavily affect retrieval performance.

Much work has been done to cope with the limitations of the ASR system. Instead of a single word sequence, lattices have been used to represent the spoken content and subword based techniques have been proposed to overcome vocabulary limitations. Unfortunately, it becomes less sufficient when more challenging real-world scenarios are considered such as when the recording conditions are noisy, the large vocabulary size for agglutinative languages, low resource languages where there is not enough training data and when the working environment is multilingual. Thus, studies for SR mainly focus on retrieval approaches to compansate ASR deficiencies.

Speech retrieval refers to searching specific pieces of spoken audio data in a large archive. Once the user provides a query, the system is expected to search over the content and returned a list of rank-ordered documents where the query can be in a form either spoken or textual. There are different types of SR tasks which are spoken document retrieval (SDR), spoken utterance retrieval (SUR) and spoken term detection (STD). The goal of SDR and SUR is to find spoken audio data which is relevant to the query, whereas the aim of STD is to return all occurrences of a query along with their time intervals of starting and ending in the document. However, STD can be challenging when it comes to low resource languages since STD methods require enough resources in the target languages. Therefore, query by example (QbE) spoken term detection is proposed as a solution to this problem.

The task of query by example spoken term detection (QbE-STD) aims to find a spoken query within spoken audio data. The user searches some relevant data within a spoken content or by means of a microphone and uttering the term of interest. The task has been approached in the literature from two different points of view which are symbol based method and template matching based method. Symbol based method transcribes the query with ASR to get its word or sub-word representation and a text-based STD is applied. Template matching method uses features extracted from the query and document and it borrows the idea from dynamic time warping (DTW) based speech recognition.

Template based QbE-STD borrows ideas from DTW based speech recognition, since the two same words can be produced at different speeds with different durations. Thus, the spoken content and spoken query are not aligned at the same rate generally. DTW approach offers to detect alignment between such two sequences by stretching or compressing the time axis when there is time difference between these sequences. Therby, direct alignment of two same words can be possible even if the durations of each word units is different. However, DTW is a computationally expensive procedure and for the applications that have multi dimensional long sequences, this complexity constrains the effectiveness of the approach.

Aside from distributing the workload to multiple processors on multicore servers, many approaches have been studied in order to speed up the algorithm by means of lower-bound estimation, cost matrix approximations, constraining and data abstraction. The lower-bound estimation proposes to compute a lower-bound distance and use this approximated distance to build an index for subsequent search. Cost matrix approximations and constraints depends on reducing the size of cost matrix and limiting the calculations while the data abstraction methods decrease the size of dimensions of the data. However, all these systems are still computationally expensive. The aim of this work is to reduce the complexity of dynamic time warping by using a pre-filtering approach and apply this to posteriorgram based keyword search (KWS).

The main contribution of this thesis is to improve the performance of a large

vocabulary continuous speech recognition (LVCSR) based keyword search using pre-filtered dynamic time warping. The baseline system uses lattice indexing for fast keyword retrieval and uses proxy keywords to cope with the out of vocabulary terms. Unfortunately, for the low resource languages, it is a challenging issue to develope such a system due to the lack of the availability of enough resources in the target languages. These resources include transcribed data for training models, phone sets and pronunciation dictionaries. Thus, our aim is to improve the baseline system with query by example spoken term detection approach using the proposed pre-filtering dynamic time warping algorithm.

Up to this point in Chapter 1, a brief introduction and aims of this work are presented. Chapter 2 gives an overview of spoken content retrieval and the previous work with different methods used in the literature. Also, approaches to spoken content retrieval and query by example spoken term detection method are presented. The overview of classical dynamic time warping approach is given in Chapter 3 and current speed-up techniques for the algorithm are presented in detail. Then, proposed pre-filtered DTW is considered and experimetal results of individual and merged systems are given. Finally in Chapter 4, this thesis is concluded with a brief summary of the results, which is followed by an introduction of the possible future works.

2. BACKGROUND

In this chapter, we define different tasks related to search and retrieval of speech documents. We first outline the different spoken content retrieval approaches and techniques which have been proposed for these approaches. Then evaluation methods are given, which are significant metrics to determine the performance of the system. Query by example spoken term detection is presented finally, and query modelling techniques for text based QbE-STD are given.

2.1. Spoken Content Retrieval

Spoken content retrieval (SCR) is an emerging technology that has become critical with the increase of multimedia documents. Spoken document retrieval, spoken term detection and spoken utterance retrieval are different types of SR tasks which all aim to provide access to large spoken documents. The primary goal of the applications of SCR is to provide the user with the most relevant set of audio documents once the user provides a query. Particularly, SCR refers to the task of retrieving the specific pieces of spoken audio data from a large content that pertain to a query requested by a user [1]. The primary challenge in SCR is to decrease speech recognition deficiencies for low resource languages and out-of-vocabulary (OOV) query terms. This chapter provides a brief introduction of the different SCR tasks, the approaches to SCR, and evaluation metrics for STD. Also, query by example spoken term detection is presented to overcome the challenges generated by low resource languages and OOV query terms.

2.1.1. Spoken Document Retrieval

In SCR, the user enters a query which can be either spoken or textual form and the system is expected to search over the content and returned a list of rank-ordered documents. The system is refered as SDR if the system output to a query is an audio file which contains the query term. Like spoken utterance retrieval, spoken document

retrieval tries to find spoken audio which is relevant to the query however, SDR differs from STD as content is more important than exact matching of query and document in SDR tasks.

The scenario where the search system returns the audio or video files that contain the utterance of the query term is spoken document retrieval. Google's Audio Indexing (Gaudi) technology is providing such a service for various YouTube channels to provide better search and browsing.

2.1.2. Spoken Utterance Retrieval

When the spoken document is long, segmenting this long document into shorter portions is become important to locate the utterances that are most similar to a query term. Thus, SUR is the retrieval of utterance length segments rather than entire long audio documents. MIT Lecture Browser [2] and browsing Turkish Broadcast News and teleconferences [3] are some examples of spoken utterance retrieval.

2.1.3. Spoken Term Detection

Spoken term detection was defined by National Institute of Standards and Technology (NIST) and the aim of STD is to find all of the occurences of a specified term -a sequence of one or more words- given corpus of speech data. STD provides a more particular measure of evaluation with exact term matching. The returned outputs have the location of the term in the speech data. STD also provides a binary decision wheather the detection is correct or false with the score of the return. An Evaluation Plan was organized in 2006 to develop this key technology [4] which three languages are involved: English, Arabic and Mandarin. The system was occured on broadcast news, conversational telephone speech and conferences meetings.

Earlier systems used a fixed set of keywords and they returned spoken instances of these keywords in the speech. This task is referred as keyword spotting [5]. The difference is that a keyword spotting system usually has this fixed keywords while the

STD system has an open vocabulary, which means that the STD system is able to search for any term without re-transcribing the speech. As the field has matured, the restriction of knowing the keywords in advance was removed [6]. Recently, keyword search has attracted with renewed approach which is creating efficient systems capable of handling large amount of spoken content and STD is generally applied for such systems [7]. An other difference is that STD uses evaluation metrics defined by NIST, however, keyword spottting uses conventional metrics as figure of merit (FOM) [8].

2.1.4. Evaluation Metrics for Spoken Term Detection

The evaluation methodology is a significant aspect for spoken content retrieval tasks to evaluate the performance of the system properly. In the SCR scenerio, the achievement can be determined as degree of relevance for SDR, while it can be defined as precise in detection for SUR and STD task. For the retrieval based metrics we label the output as true positive if it is retrieved and relevant, false positive if it is retrieved and irrelevant, false negative if it is not retrieved and relative and finally true negative if it is not retrieved and irrelevant. Besides the evaluation metrics, evaluation data sets can be useful to decide the effectiveness of the systems. It is difficult to compensate the different effects of different system considerations, since the evaluation method should regulate the system's performance and user's goal.

Precision-Recall Rate and the F-Measure are set-based measures for spoken content retrieval which are computed using unordered sets of documents [9]. Precision (Pr) is the fraction of returned documents from the collection that are relevant to the query, and recall (Rc) is the fraction of relevant documents in the collection that are returned. They are computed using the formulae:

$$Pr = \frac{1}{Q}\sum_{q=1}^{Q}\frac{C(q)}{A(q)} \qquad Rc = \frac{1}{Q}\sum_{q=1}^{Q}\frac{C(q)}{R(q)} \qquad F = \frac{2 \times Pr \times Rc}{Pr + Rc} \tag{2.1}$$

where C(q) represents the number of relevant documents which are correctly retrieved while A(q) is the total number of retrieved documents. R(q) denotes the total number

of documents that are relevant to query q given Q queries. For each such set, precision and recall values can be plotted to give a precision-recall curve.

There are also other evaluation metrics such as mean average precision (MAP), R-Precision and Precisıon@N metrics which are derived from precision and recall rates. Map for a set of queries is the mean of the average precision scores for each query. R-Precision and Precision@N are similar since they are the Pr value of the top N documents, but the difference is N varies for each given query and it is set to the number of relevant documents for Precision@N [10].

On the other hand, receiver operating curve (ROC), figure of merit, detection error trade-off (DET) curve and the term weighted value (TWV) metrics are related to STD applications. In spoken term detection the returned outputs are rank-ordered with respect to confidence scores. This scoring scheme was introduced in [4]. A binary decision is returned for the queries as 'YES' and 'NO' according to some given threshold. If the score for a given query is above this threshold the decision is 'YES' whereas if the score is under this threshold the decision is 'NO'. The evaluation system assess the output as 'HIT' when the central point of the query matches with the reference occurence. Otherwise, the output is assessed as 'FA' which means false alarm.

Probability of miss (P_{MISS}), probability of false (P_{FA}) and probability of hit (P_{HIT}) are computed for each query as:

$$P_{HIT} = \frac{N_{HIT}(term, th)}{N_{target}(term)} \tag{2.2}$$

$$P_{MISS}(term, th) = 1 - P_{HIT}(term, th) = 1 - \frac{N_{HIT}(term, th)}{N_{target}(term)} \tag{2.3}$$

$$P_{FA}(term, th) = \frac{N_{FA}(term, th)}{T - N_{target}(term)} \tag{2.4}$$

where the $term$ the term being searched, th the decision threshold, T is the total duration of the document. $N_{target}(term)$ denotes the number of all correct occurences of the term in the dataset. $N_{HIT}(term, th)$ is the number of correctly detection term occurences for the given th value and the $N_{FA}(term, th)$ is the number of false alarms for the term.

The ROC curve plots the true positive rate against the false positive rate. An ROC curve always goes from the bottom left to the top rigth and for a good system curve should be rising [9]. FOM is an upper-bound estimate on STD accuracy averaged over 1 to 10 false alarms per hour. However, the FOM metric does not provide useful information since for a single th for all queries since it is depend on each term's threshold. Thus, the DET curve is introduced which is widely used in the recent STD applications [11]. The DET curve represents the dependency of false alarm probability to the miss probability with all possible values of the threshold. In Figure 2.1 an example of DET curves are shown.

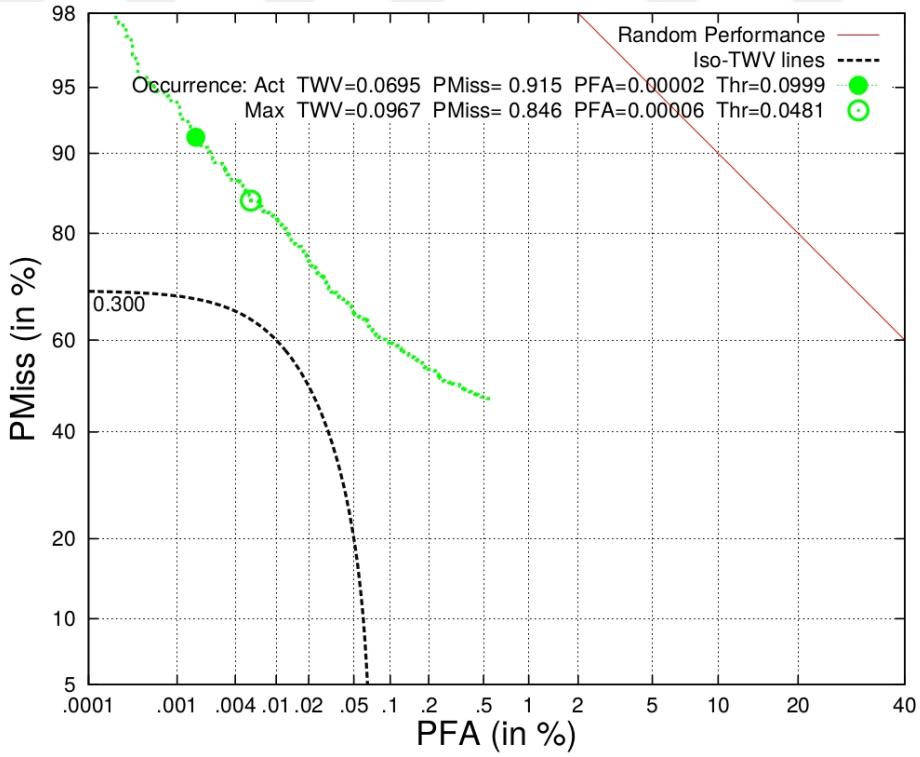

Figure 2.1. An example DET curve.

A novel metric to evaluate STD system's performance TWV is defined in NIST STD 2006 Evalatuation Plan as:

$$TWV(th) = 1 - \frac{1}{Q}\sum_{k=1}^{Q}\{P_{MISS}(Q_k, th) + \beta P_{FA}(Q_k, th)\} \tag{2.5}$$

where β is a user defined parameter to adjust the trade-off between miss and false alarm cost and it is calculated as:

$$\beta = \frac{C}{V}\left(pr_{term}^{-1} - 1\right) \tag{2.6}$$

where C is the cost of false alarm, V is the value of correct detections and pr_{term}^{-1} represents the prior probability term. For NIST STD 2006 evaluations $\frac{C}{V}$ was set to 0,1 and pr_{term}^{-1} was set to 10^{-4} which yields β is constant equal to 999,9. TWV gives a number between -∞ to 1. A sytem that achieves to detect all of the occurences of all term with no false alarms will have a TWV of 1. The TWV can get negative values and for the empty hit list, TWV is 0. TWV is calculated for all possible thresholds and the maximum TWV is called maximum term weighted value (MTWV). Optimal term weighted value (OTWV) is determined using the best threshold value for each query and supremum term weighted value (STWV) is calculated by taking the correct decisions into account. Actual term weighted value (ATWV) is the TWV using the actual decisions and it represents the ability to predict the optimal operating point for the system given the TWV metric.

2.2. Approaches to Spoken Content Retrieval

Research in spoken content retrieval has been continuing for more than twenty years. During this time many techniques have been proposed for different tasks and datasets. In 1990s, large vocabulary continuous speech recognition had matured yet and the task addressed as finding voice messages by using wordspotting. Afterwards, the task is referred as information retrieval [12, 13]. In [6], a wordspotting method based on phonetic lattices made the method is possible to fulfill vocabulary inde-

pendent wordspotting. A considerable contribution is performed using ASR ouuputs and combining the outputs with the information retrieval systems for SCR [14]. A significant change in SDR research was the text retrieval conferance (TREC) evaluations [15]. The application Speechbot [16] was the first SCR system used in web. The ASR results had fewer errors owing to better recording conditions since these applications mainly centered on Broadcast News speech. The word error rates (WER) for the TREC evaluations were better than those applications that use ASR outputs. To tackle the ASR deficiencies, lattice based methods were used in indexing and search instead of the 1-best result of the ASR system. Subword based retrieval systems were developed to overcome the problem of retrieval for OOV query terms.

Today, large vocabulary continuous speech recognition systems are widespread to transcribe audio samples for retrieving the speech data. These transcripts are indexed at the word or subword level and the index is subsequently searched. Several methods have been used to improve retrieval performance which covers lattice-based and subword based methods as basic approaches. Additionaly, some ASR modified techniques were studied recently including acoustic modelling, language modelling and confusion modelling for spoken content retrieval.

2.2.1. Subword Based Methods

It is common to transcribe the spoken documents into text using large vocabulary speech recognizer then to use a text retrieval system for recognition. However, there are some problems with this procedure. The major one is the recognizer vocabulary, which is needed to cope with new words from growing and diverse collections. Although very large vocabulary ASR systems are becoming more prevalent, users typically tend to search discriminative, therefore low-frequency words. This means that a typical query term is likely to be either an OOV term or a term which the language model has not been trained well because of lack of data [17].

Subword representation approach is proposed to deal with many of the problems for spoken document retrieval, which are stated above. The use of subword units in

the recognizer limits the size of the vocabulary, that is needed to cover the language. Moreover, the use of subword unit indexing terms allow for new query term detection during retrieval. Although there is a trade-off between the size of unit and accuracy of speech recognition, it can be reduced by the choice of the subword units and the modelling of their sequental constraints.

Several subword based approaches have been studied in the literature recently. Syllable-like units, phones and morphemes which are linguistically motivated units have been studied. Also, data driven units as multigrams, particles and morphs have been proposed for the spoken document retrieval taks. Syllables-like units composed of nonoverlapping, variable-length, phone sequences which are generated by rule [18]. The multigram model assumes that a phone sequence is constituted from concatenation of independent, nonoverlapping, variable-length, phone sequences and this model the parameters are the set of subsequences with their probabilities [19]. The particles are defined as within-word sequences of characters [20]. Statistical morphs, which are essentialy used in languages with rich morphology, are based on morpheme-like subword units learned in an unsupervised manner [21]. Since all are smaller than words, search space reduces. However, for all these approaches, as units get smaller the recognition accuracy decreases as well [17, 22].

2.2.2. Lattice Based Methods

Lattices are connected, directed, loop-free graphs where nodes represent points in time and edges are associated with phone hypothesis and scores respresenting their likelihoods. Figure, 2.2 shows an example of lattice structure. Lattices were proposed in [6] for vocabulary independent keyword spotting task. In [23], the queries are modelled as phone lattices and wordspotting is used to accomplish spoken document retrieval task. Lattices are used to index the document for retrieval purposes and the weighted automata, confusion networks (CN) and position specific posterior probability lattices (PSPL) are models to represent the probabilistic structure for indexing lattices.

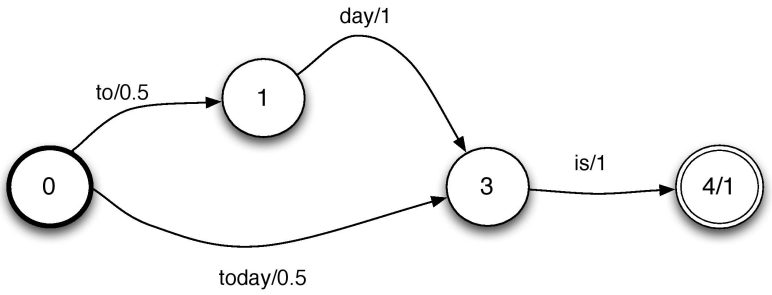

Figure 2.2. An example of lattice structure.

The weighted finite state transducer (WFST) is introduced to index lattices [3] to deal with a large number of ranked or weighted alternatives since indexing probabilistic data of ASR output is challenging due to the uncertainty of performance. The main idea of the work is to design an algorithm to represent each utterance as a WFST and each query as a automata. Afterwards, retrieving the queries with mapping each factor of the query automata to the set of indices of utterances when combined with the utterance WFST. Figure 2.3 shows an example of the weighted finite state transducer indexing of the lattices. This procedure is also suitable for spoken term detection since the knowledge of position information exists in the lattice structure. In [24], multiple pronunciations for the OOV terms are produced and used as query automata.

Confusion networks and position specific posterior probability lattices are other methods for indexing. They provides simpler approximations to lattices since ASR lattices require great deal of connectivity information. Confusion networks are concatenated series of hypothesis sets which are derived from lattices by clustering arcs (words) with similar timing to form hypothesis which are known as confusion sets. Confusion networks may have extra paths, which do not exist in the original lattice and may not have some of the paths, which exist in the original lattice. This may cause higher recall rates -higher false alarm rates- in spoken content retrieval applications. Figure 2.4 shows an example of confusion network structure.

Similar to the confusion networks, position specific posterior probability lattices

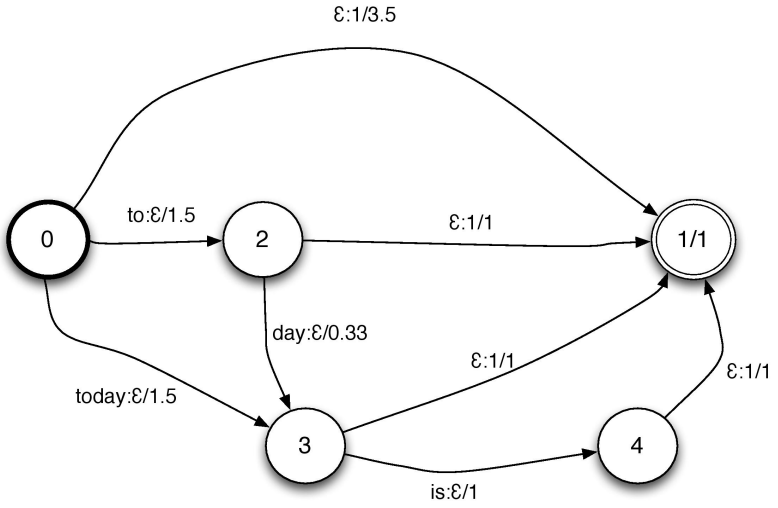

Figure 2.3. An example of the weighted finite state transducer indexing.

are used to provide the information of document identity, position and posterior probability of the query term. Confusion networks and position specific posterior probability lattices are used together for soft indexing [17] of the speech by binning lattice links approximately based on the time span of the link.

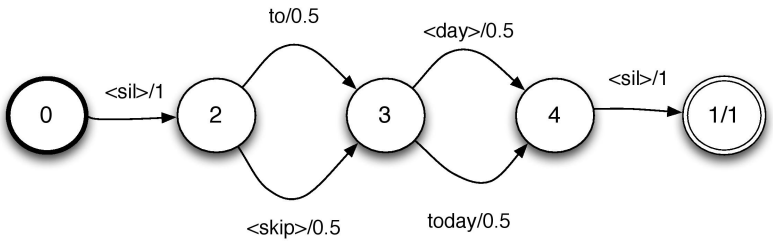

Figure 2.4. An example of confusion network structure.

2.2.3. Acoustic Modelling for Spoken Content Retrieval

For acoustic modelling approaches, there are three methods which are proposed in recent years. The weighted discriminative training, the first method, uses training techniques such as minimum classification error (MCE) [25], minimum phone error

(MPE), and lately the state level minimum bayes risk (sMBR). While MPE training have been commonly used to get better hidden Markov model (HMM) acoustic models [26, 27], sMBR training has been shown as one of the most effective discriminative training method for acoustic models with deep neural network (DNN) [28].

$$F(\theta) = \sum_{r=1}^{R} \sum_{s_r \in \mathcal{L}(u_r)} A(w_r, s_r) P_\theta(s_r|u_r) \tag{2.7}$$

Equation 2.7 shows how the objective function sets the acoustic model parameter. $A(w_r, s_r)$ is the accuracy estimated for u_r by comparing s_r with w_r, $P_\theta(s_r|u_r)$ is the posterior probability of the path s_r given u_r. The R is the total number of utterances in the training set, u_r is the rth training utturances, w_r is the reference transcription of u_r and s_r is an allowed word sequence in the lattice $L(u_r)$. The term $A(w_r, s_r)$ is defined as accuracies of different words and by adjusting the weight of $A(w_r, s_r)$ in favor of the keyword the accuracy for these terms can be favored.

Secondly, the whole word modelling is used for the keyword spotting scenerio. The whole word models are trained for the keywords only if the keywords have enough examples in training data and performance is increased in comparison with conventional phone-level models [29, 30].

Another method for acoustic modelling is relevance feedback which is used to combine the ASR and the retrieval engine. The user gives some feedback to the rank ordered results which system gives as an ouput, then a new set of acoustic model is re-estimated based on the feedback. Acoustic model specifies scores for ranking outputs. This query specific acoustic models can be used for the unrestricted number of acoustic conditions for the spoken content since the model is trained according to the specific query and corresponding feedback. This approach has been used successfully in [31–33].

2.2.4. Language Modelling for Spoken Content Retrieval

Language modelling proposes to increase the retrieval performance by repeating the sentences which include the query term during language modelling training. Also, in neural network based language models, similar approach is applied by using a feature vector which represents a history word sequence. In [34], it is stated that neural network based language models decrease the word error rate wheras they decrease term detection performance. To overcome this trouble, new method is proposed [35] by weighting the output distribution of neural network language model between a term which is less frequent in the training corpus. By this way, neural network language model learns to avoid making mistakes on these terms.

2.2.5. Confusion Modelling for Spoken Content Retrieval

Confusion modelling is used to model the occurence of the recognition errors by using three approaches: query transformation, content transformation and fuzzy match. These methods achieve better retrieval performance.

Query transformation is to transform the word or subword sequence of each query into the sequences that the query tends to be mis-recognized and the new sequences are used to retrieve documents. Phone to phone transducer, token to token transducer, phone to phone to token transducer are confusion models, which are used in [36] to create alternative reresentation for query terms. In [37] proxiy keywords are created using a phone confusion matrix. The discriminative trained phoneme confusion model which expands the phonemic index of a KWS system by adding phonemic variation is proposed in [38].

In content transformation, instead of the query, the recognition output is transformed for the spoken content. Term weigted figure of merit is used which is obtained by averaging the detection rate at each operating point accross a set of evaluation search terms in [39]. Also [40] optimizes the FOM, through its use as an objective function to train a transformation of the index.

Fuzzy match defines a distance between different word or subword sequences. The lattices containing word or subword sequences sufficiently close to the query being retrieved.

2.3. Query by Example Spoken Term Detection

Spoken term detection has received much interest for years because of the ever growing amount of speech data and necessity for their retrieval. However, text-based STD, which are presented in the previous section, suffers from several drawbacks. In the text-based STD, the resources and an exact knowledge are assumed to be known, and therefore we can utilize a resonable amount of transcribed data, phone sets and pronunciation dictionaries. Unfortunately, for the low resource languages, it is challenging issue to develop a spoken term detection system due to the lack of the availability of enough resources. Therefore, a need for language independent approach emerges and the query by example spoken term detection approach is developed to solve this problem. In query by example method, the users provides query samples, either via speech cuts corresponding to the query or by speaking the query. Symbol based and template based methods have been used for query by example spoken term detection approach.

2.3.1. Symbol Based Search Methods

Symbol based methods depend on phone transcription of the speech data corresponding the query. It decodes query with a speech recognizer and derives the query in word or subword representation, which is not suitable procedure since it is still text based spoken term detection like fashion. Therefore, an attractive symbol based method is proposed [41] called acoustic keyword spotting (AKWS) and it makes possible to apply symbol based method for the language independent scenerio.

In AKWS, the utterances are passed through a voice activity detection and are filtered from the nonspeech part firstly. Then, the remaining audio is converted to 3-state phone posterior features and logarithm of these posteriors are calculated. The

results are fed into a decoder, which is designed as keyword and background model decoders. In Figure 2.5, the scheme of acoustic keyword spotting system is shown.

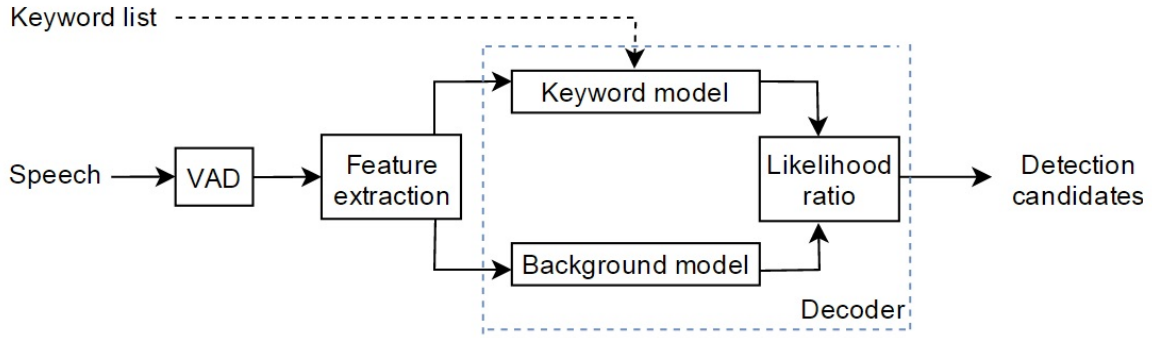

Figure 2.5. Scheme of acoustic keyword-spotting system [42].

The keyword model is a concatenation of phone models of which the keyword consists and the background model is a single phone loop that can take any sequences of phones [42]. A standart Viterbi-based decoder calculates the likelihood ratio. The filler, which is represented by a free phone loop, models all the speech preceding and succeeding the keyword. The ratio of two parts, which are filler and keyword model and filler background model, are calculated. The system returns a detection if the ratio is below some threshold and the likelihood ration is divided by the length of the detection to compensate for the different length of the queries.

2.3.2. Template Matching Based Search Methods

Template matching can be simply defined as searching and finding the location of a template in a larger version of it. Similarly for a given spoken query and an utterance, template matching method scans the utterance and locates the time interval which is pertained the query. In Figure 2.6, the general architecture of template matching based search approach is represented. After features of the utterance and query are extracted, a distance matrix is calculated and the best match is detected by a searching method. There are many acoustic features have been used which includes mel frequency cepstral coefficient (MFCC) [44,45], perceptual linear prediction (PLP) [46], bottleneck features

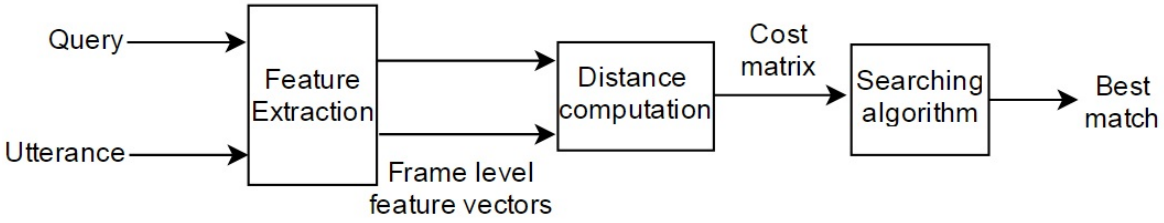

Figure 2.6. General architecure of template matching based search method [43].

[47] and frequency domain linear prediction features [48].

Different methods have been proposed to derive posterior features. One popular way is to train a Gaussian mixture model (GMM) from given speech data, and derive Gaussian posterior features. Gaussian posterior features have the posterior probabilities of the corresponding Gaussian components. Another way is the phoneme posterior feature, which is generated by an existing phoneme recognizer. Posterior features are also used in addition to Gaussian posterior features and phoneme posterior features. [49, 50].

Dynamic time warping and self similarity matrices (SSMs) are two common searching algorithm to detect possible matches from distance matrix. SSMs are square symmetric matrices storing the differences between frames. The matrices of query and utterance are similar if they are similar. Dynamic time warping method will be presented in the next chapter.

2.3.3. Query Modelling Methods for Text Based QbE-STD

Query by example spoken term detection approach is very efficient for low resources languages and out-of-vocabulary terms where there is not sufficient resources for the target language. However, in some cases we do not have any examples of the queries but given a list of text based keywords which is quite possible for the low resources term detection scheme. Therefore, some modelling techniques are used to convert the queries to same form with the spoken document.

In order to model the query, posteriorgrams are used. A posteriorgram is a time vs class matrix where the class can be Gaussian mixtures, HMM states and phonemes. Phone-posteriorgrams represent the posterior probability of each phone class for each specific time and they can be generated by a DNN trained with a labeled corpus of the target language. The DNN produces the column of the posteriorgram by using the spoken document frames as an input. To model the query as a series of posterior vectors text-to-speech (TTS) based modelling, example based modelling, binary and average modelling schemes are used.

The text-to-speech system is one of the most common methods to obtain a query posteriorgram. A TTS system is composed of a front-end and a back-end. The front-end assigns phonetic transcription to each word, then the back-end converts the linguistic representation into sound. The output of the TTS is piped to some neural network and feature extraction is performed [24]. One disadvantage of this system is that it inherently conflicts the low-resource assumption in such a way that the TTS system depends on data of the target language.

In example based query modelling, the examples for each word of the query can be extracted from the training corpus alignments and can be merged to obtain the multiword terms. Also, the pseudo relevance feedback technique can be used for example based query modelling. The best outputs of the retrieval system can be extracted and used as the query.

Binary query modelling is based on the assumption that a DNN ouput of the spoken query, in which the phoneme classes belongs to each frame has zero entropy. Such a probability distrubution over phonemes have 1 on one of the phonemes and 0 on others. By using the pronunciation dictionary or a grapheme-to-phoneme (G2P) system, the text query is first converted into a sequence of phonemes. Then, each phoneme in this sequence is simply repeated using the average durations obtained from the training corpus.

Average query modelling uses the phone averages obtained from the training

data and a certain class is generally labeled as a probability density function over classes that has a non zero entropy.

3. METHODOLOGY AND EXPERIMENTS

This chapter presents the proposed pre-filtered dynamic time warping method. We first give an overview of classical dynamic time warping approach and its high computational demand problem. Then we describe the previous methods proposed to deal with this drawback. In order to present our work, we introduce the experimental configurations and the dataset used in our experiments. Finally, experimental results for individual system and merged scores with the LVCSR lattice based system are given.

3.1. Dynamic Time Warping

The spoken content and the spoken query are not spoken at the same rate generally. Therefore, the similarity between them might not be detected due to the time difference. The dynamic time warping approach was proposed to deal with this trouble [51]. The method allows a nonlinear mapping between two sequences by stretching or compressing the time axis. Figure 3.1 shows the intuitive approach of dynamic time warping to compare two time series. The sum of distances between each pair of points is calculated to determine the warping path.

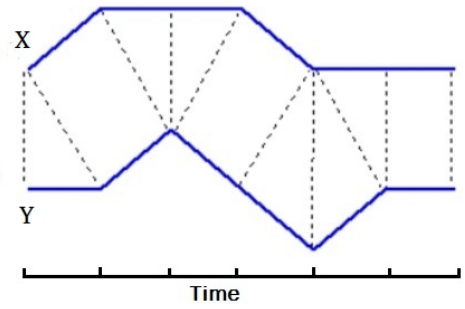

Figure 3.1. A warping between two time series.

Given the two sequences X and Y, of length x, and y, respectively, where:

$$X = x_1, x_2, ..., x_x$$

$$Y = y_1, y_2, ..., y_y$$

DTW constructs an x-by-y matrix, called the cost matrix, to determine the alignment. Both x and y axes of the cost matrix represent time. The algorithm fills the matrix by calculating the distance between points that corresponds to each cell. The (i, j)th element of the cost matrix, d(x_i,y_j), stores the distance between the two points x_i and y_j. For the distance measure, typically the Euclidean Distance is used. Once a starting point is given, the warping path is found by tracking the adjacent cells which have minimum distance. The dynamic programming algorithm evaluates the path iteratively as shown in Equation 3.1, in which $\gamma(i, j)$ represents the cumulative distance.

$$\gamma(i,j) = d(x_i, y_j) + \min \begin{cases} \gamma(i-1, j), \\ \gamma(i-1, j-1), \\ \gamma(i, j-1) \end{cases} \tag{3.1}$$

DTW algorithm, which is shown in Algorithm 1, calculates the distances between each pair of points and generates the cumulative distance. In Figure 3.2, a cost matrix between two sequences and the warping path are illustrated. Both axes of the cost matrix represent time. To find the warping path, every cell of the matrix must be filled. Once the matrix is filled, the warping path is calculated by starting $d(x_i, y_j)$. Whichever of the three adjacent cells has the smallest cumulative distance is added to the beginning of the warp path and it continuous until $d(x_1, y_1)$ is reached. If two sequences are identical, the warp path is a straight diagonal line. The horizontal and vertical lines on the path show the warping of single points to multiple points as a significant feature of DTW to align two sequences which have time difference.

Algorithm 1 DTW

for $i = 1$ to X, $j = 1$ to Y **do**
  if i=1 **or** j=1 **then**
    $\gamma(i,1) = \sum_{t=1}^{i} d(\mathbf{x}_t, \mathbf{y}_1)$
    $\gamma(1,j) = \sum_{t=1}^{j} d(\mathbf{x}_1, \mathbf{y}_t)$
  else
    $\Omega = \{(i, j-1), (i-1, j), (i-1, j-1)\}$
    $(r,s) = \underset{\forall (p,q) \in \Omega}{\operatorname{argmin}}(\gamma(p,q) + d(\mathbf{x}_i, \mathbf{y}_j))$
    $\gamma(i,j) = \gamma(r,s) + d(\mathbf{x}_i, \mathbf{y}_j)$
  end if
end for

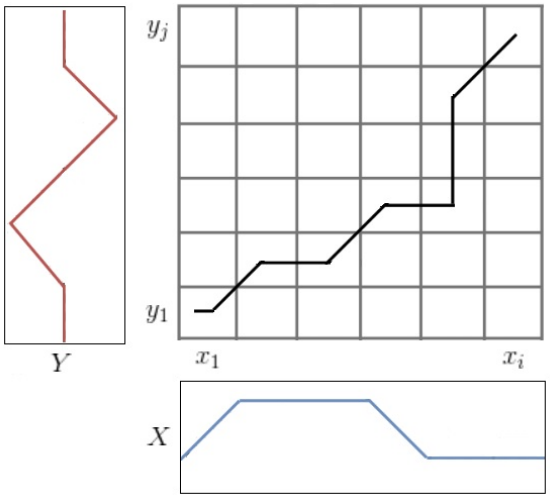

Figure 3.2. The cost matrix and warping path.

3.1.1. Segmental Dynamic Time Warping

The classic dynamic time warping matches two sequences primarily end-to-end. A modification for this approach is proposed since the query is generally shorter than the document. Segmental DTW is first used in [52] for the query by example spoken term detecton task. The algorithm divides the cost matrix into several overlapping

diagonal bands and applies regional search. Once a diagonal band starts from a point then segmental dynamic time warping iterates through all diagonal bands to find the corresponding optimal path which contains minimum cumulative distance. In Figure 3.3, the segmental dynamic time warping is illustrated.

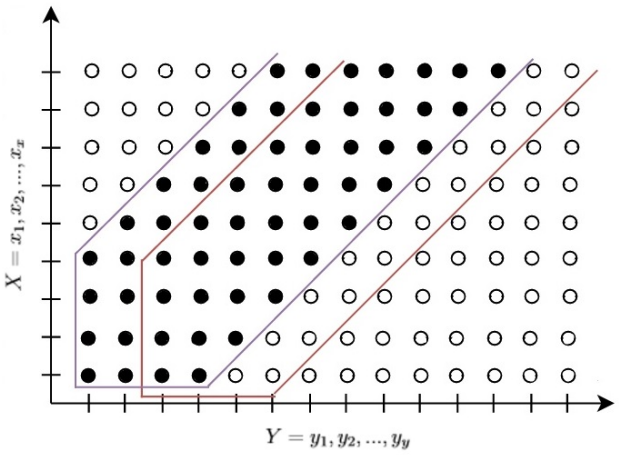

Figure 3.3. The segmental dynamic time warping.

The segmental dynamic time warping defines two constraints. One of is the adjustment band. Given the two sequences, the distance is $d(t) = (i_t, j_t)$ where i_t and j_t donotes the t-th coordinate of the warping path. The adjustment band conditions requires that $|i_t - j_t| \leq R$ to aligned two sequences. This constraint prevents the warping from going too far ahead or behind in both sequences. The second constraint is the step length of the start coordinates. The cost matrix can be divided into continuous diagonal bands with width 2R+1 by applying different start points of the warping process. The R is the adjustment band size and diagonal band moves R steps forward for a new search. Thus, given the length of the utterance n, the start coordinate is :

$$(1, (t-1)R + 1), \qquad 1 \leq t \leq \left[\frac{n-1}{R}\right]. \tag{3.2}$$

3.1.2. Subsequence Dynamic Time Warping

The subsequence dynamic time warping is another approach for classical dynamic time warping. The spoken document and the spoken query can have lengths differ at most by the width of the diagonal bands. Thus, when the speaking rates different, this constraint can be hazardous for proper search. Therefore, in subsequence dynamic time warping, the diagonal bands are not used and it allows to start from any point of the subsequences in contrast with classical approach. The algorithm considers the best match between the query and all subsequences of the document, which yields the most similar of them. The cumulative distances between the query and all subsequences averaged by the warping path length for the similarity detection. Algorithm 2 shows the subsequence dynamic time warping algorithm where $\mathcal{L}$ denotes the information of path length and $\mathcal{L}_O$ represents the optimal beginning point.

Algorithm 2 Subsequence DTW

for $i = 1$ to X, $j = 1$ to Y **do**
  if i=1 **or** j=1 **then**
    $\gamma(1, j) = d(\mathbf{x}_1, \mathbf{y}_j)$
    $\mathcal{L}(1, j) = 1, \mathcal{L}_O(1, j) = j$
  else if j=1 **then**
    $\gamma(i, 1) = \sum_{t=1}^{i} d(\mathbf{x}_t, \mathbf{y}_1)$
    $\mathcal{L}(i, 1) = i, \mathcal{L}_O(i, 1) = 1$
  else
    $\Omega = \{(i, j-1), (i-1, j), (i-1, j-1)\}$
    $(r, s) = \underset{\forall (p,q) \in \Omega}{\operatorname{argmin}} \dfrac{\gamma(p, q) + d(\mathbf{x}_i, \mathbf{y}_j)}{\mathcal{L}(p, q) + 1}$
    $\gamma(i, j) = \gamma(r, s) + d(\mathbf{x}_i, \mathbf{y}_j)$
    $\mathcal{L}(i, j) = \mathcal{L}(r, s) + 1, \mathcal{L}_O(i, j) = \mathcal{L}_O(r, s)$
  end if
end for

3.1.3. Speed-up Techniques for Dynamic Time Warping

Dynamic time warping algorithm has attracted much interest to align two time series because of its intuitive approach. However, one of the major isuues of this method is the high computation demand. DTW algorithm takes O(xy) time to compute a match between two time sequences consisting of x and y elements or frames. For the applications that have data rich points, this complexity constraints the effectiveness of the approach. Many studies have been done to reduce its computational cost ever since DTW was developed. Aside from paralleling the task by distributing the workload to multiple processors on multicore servers, some speed-up approaches for the algorithm are proposed.

The basic idea behind the lower-bound estimate based DTW search is to compute a lower-bound distance which approximates the actual DTW distance between the query and the document. Then, this lower-bound distance is used to build an index for subsequent search.The first lower-bound estimate was proposed by [53] which compares two one-dimensional time series by using Euclidean distance metric for the distance. Firstly, the whole series are scanned to determine their maximum and minimum points. The points in first time series that are larger than the maximum of the second time series must be taken into account at least the squared difference of their value, and the maximum value to the DTW distance. Also, the smaller points then the minimum of the other time series must contribute at least the squared difference of their value, and the minimum value to the DTW distance. However, there are some drawbacks in this method such as false dismissals and the high computational demand since the method scans the whole series to find the maximum and minimum points. An improved version of this study is proposed in [54] which uses the first, the last, the minimum and the maximum points of time series. Nevertheless, it still requires high computational demand because of determining the maximum and minimum points and the definition of these points does not allow to study on multi-dimensional time series [55].

Another lower-bound estimate, called piecewise aggregate approximation (PAA)

is proposed which uses Euclidean distance as the distance metric and studies on posteriorgrams which are series of multi-dimensional posterior probability vectors. The basic idea is to divide the query and the document into windows and use the maximum posterior values within that window for the dynamic time warping calculation. Given two posteriorgrams X and Y, of length N, and M respectively, the piecewise aggregate approximation creates lower-bound approximated posteriorgrams as $\widehat{U} = \widehat{U}_1, \widehat{U}_2, ..., \widehat{U}_B$ and $\widehat{Y} = \widehat{Y}_1, \widehat{Y}_2, ..., \widehat{Y}_B$, where $\widehat{U}_i$ denotes the ith block of the approximated upper-bound envelope U and defined as $\widehat{U}_i = \widehat{u}_i^1, \widehat{u}_i^2, ..., \widehat{u}_i^D$ where

$$\widehat{u}_i^p = max\left(u^p_{\frac{M}{B}(i-1)+1}, ..., u^p_{\frac{M}{B}i}\right). \tag{3.3}$$

$\widehat{Y}_i$ denotes the ith block of the approximated Y and defined as $\widehat{Y}_i = \widehat{y}_i^1, \widehat{y}_i^2, ..., \widehat{y}_i^D$ where

$$\widehat{y}_i^p = \frac{B}{M} \sum_{j=\frac{M}{B}(i-1)+1}^{\frac{N}{B}i} y_j^p \tag{3.4}$$

Since the cumulative distance of window is lower than the classical DTW cumulative distance, the computational demand reduces [56]. By using these shorter sequences, piecewise aggregate approximation cumulative distance can be calculated as

$$PAA(\widehat{Q}, \widehat{Y}) = \sum_{i=1}^{B} \frac{M}{B} \cdot d(\widehat{U}_i, \widehat{Y}_i) \tag{3.5}$$

where $d(\cdot)$ denotes the inner product function.

There has also been some approximation methods to reduce the size of the dynamic time warping cost matrix. Since the algorithm is able to run on these approximated cost matrices instead of on the whole cost matrix, the complexity decreases. Recently, a two stage cost matrix approximation method is proposed in [57–59] where

the first stage uses locality sensitive hashing (LSH) to represent the time series. The basic idea behind the method is to map high dimensional data to low dimensional bit vectors. Each signal frame $x \epsilon \mathbb{R}^d$ in both X and Y is mapped into low dimensional bit signatures $h(x) \in \{0,1\}^b$. After the mapping, the Hamming distance is used as a distance metric. The process is implemented by generating a transformation matrix $\mathcal{T} \in \mathbb{R}^{d \times b}$ populated by random numbers from $\mathcal{N}(0,1)$. The k_{th} column defines the normal to the k_{th} random hyperplane and the k_{th} bit of the signature $h(x_k)$ takes value 0 if the inner product of x and t_k is less then zero, and 1 otherwise. The approximated cost matrix can be formed by only most similar frames to the query frames. The cosine similarity between two frames can be approximated as:

$$cos(x_i, y_j) \approx cos\left(\frac{H(h(x_i), h(y_j))}{b}\pi\right) \tag{3.6}$$

where $H(\cdot,\cdot)$ denotes the Hamming distance and the Equation 3.6 approaches equality as $b \to \infty$.

Intuitively, a good alignment path should not occur away from the diagonal of the cost matrix, since the warping path is diagonal for two same sequences. Therefore, some constraints are proposed to limit the cells of the cost matrix with a band enclosed by lines which contains the diagonal of the cost matrix. Two commonly used global constraints are the Sakoe-Chiba Band [51] and the Itakura Parallelogram [60] which are illustrated in Figure 3.4.

Let $R \subseteq [1:N] \times [1:M]$ be a subset referred to constraints region. The Itakura Parallelogram defines a region which constraints the slope of the warping path whearas the Sakoe-Chiba Band has the main diagonal and a fixed width $T \in \mathbb{N}$ [62] for the warping path can be occured, which implies that a point x_n can be aligned with a point y_m where

$$m \in \left[\frac{M-T}{N-T}(n-T), \frac{M-T}{N-T}(n+T)\right] \cap [1:M]. \tag{3.7}$$

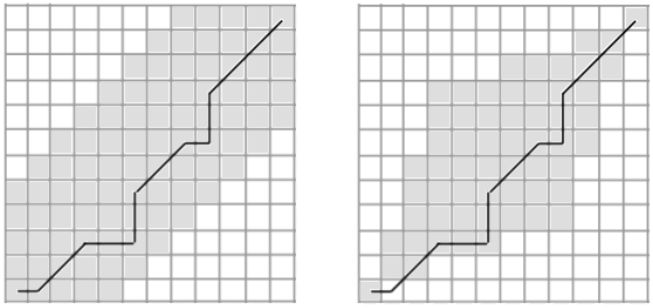

Figure 3.4. An illustration of the Sakoe-Chiba Band (left) and the Itakura Parallelogram (right) [61].

Although, constraints reduce the complexity of the dynamic time warping, they work well only in the scenerios that the optimal warping path is close to a linear warp. However, if there is time difference between the same sequences, constraining methods suffers from finding the optimal path since it can be occured far from the diagonal because of the the time difference [61].

Data abstraction approaches assert that the complexity of dynamic time warping can be decreased by reducing the representation of the query and document [63, 64]. The basic idea behind the method is reducing the sequences from n dimensions to N dimensions, by simply dividing them into N equal sized frames. These frames are calculated as

$$\bar{x}_i = \frac{N}{n} \sum_{j=\frac{n}{N}(i-1)+1}^{\frac{n}{N}i} X_j \tag{3.8}$$

where $\bar{x}_i$ is the ith element of the reduced sequences. The transformed representation is equal to the original when n=N. The study defines a constant c, which is the compression rate for the original sequence, which equals to n/N. Then, the time complexity of dynamic time warping is reduced from O(nm) to O(NM) where n and m are the original dimensions, N and M are the transformed dimensions of the sequences.

A lower resolution cost matrix is determined with the new sequences which are reduced in size. Therefore, running the algorithm on the dimensionally reduced cost matrix become more menageable. In Figure 3.5, an illustration of the original cost matrix and the lower resolution cost matrix which correspond to original data and transformed data is shown.

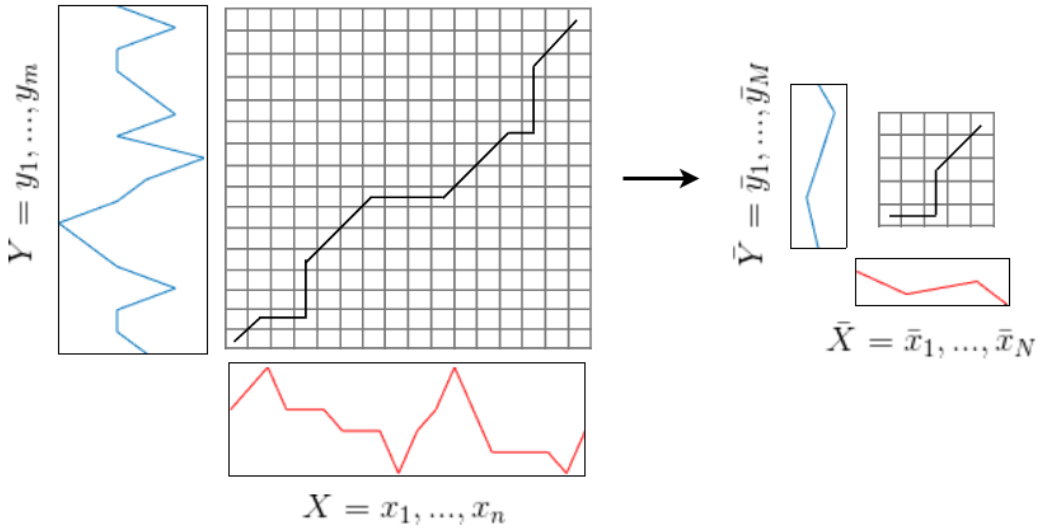

Figure 3.5. An example of the original cost matrix (right) and the lower resolution cost matrix (left) [63].

However, the warping path of the lower resolution matrix might has difficulties to determine an alignment when the optimal path does not passes through the low-resolution cells. Some variations on the optimal warping path can be meaningful to determine a match, which are ignored in the transformation process.

3.2. Pre-filtered Dynamic Time Warping

The speed-up techniques, which are presented in Subsection 3.1.3, reduce the computational complexity of dynamic time warping. However, these approaches have several drawbacks. Lower-bound estimation, data abstraction and matrix approximation methods are based on reducing the number of points in the document and query sequences by using some transformation methods, which can be harmful to detect

small variations between the query and the document. The similarity depends on the low resolution sequences or cost matrices in these approaches, thus, the optimal warping path might not be detected with these low resolution information. Also, the methods which constrain the cost matrix have the deficiency that the similarity might not be calculated between two sequences if the optimal warping path occurs far from the diagonal. However, the proposed method uses a filtering step before the calculation of the cost matrix. Thus, in our method, dynamic time warping algorithm is performed on original sequences of the query and the document and on the original cost matrix without any constraint.

Since, in classical dynamic time warping, the query is searched in the whole document posteriorgram, for each query the corresponding cost matrix is generated recurrently. Thus, this grand matrix calculation happens for every query and the algorithm searches the whole document posteriorgram including unlikely paths for the query. Our aim is to eliminate these unlikely paths. The intuition behind our approach is to divide the document posteriorgram matrix into segments, or submatrices, and turn them into supervectors, which are used to determine the possible candidates for query matching. Thus, dynamic time warping search will be performed on the possible candidates instead of the entire document. The block diagram of the proposed system can be seen in Figure 3.6.

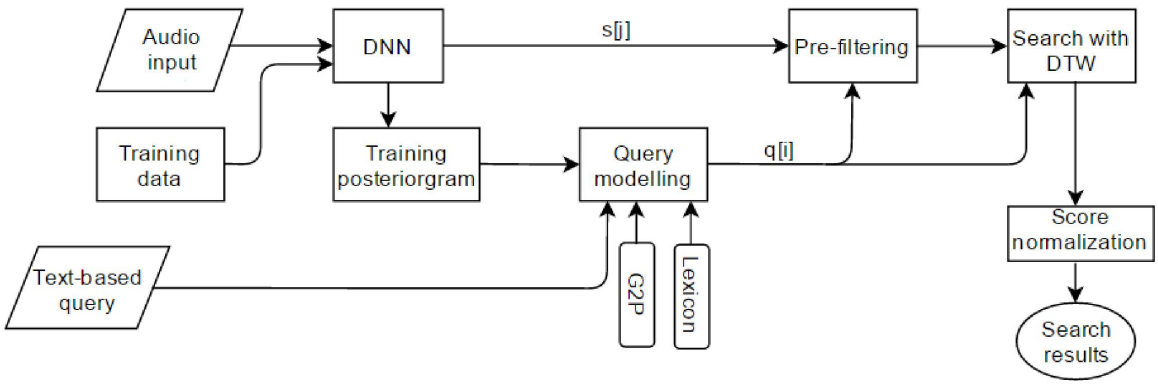

Figure 3.6. Flowchart of the proposed system.

3.2.1. Generation of Supervectors

The posteriorgrams of the document and the query are matrices whose rows and columns correspond to the time frames and the phonetic classes, respectively. In order to generate the supervectors, the first step is to divide the posteriorgram of the document into segments which have the same size as the query posteriorgram. However, the beginning point of the query may not be found if it is not included within the segments. To avoid this problem, the document posteriorgram is divided into segments with some overlap. Then, both query and segments are calculated as supervectors as:

$$\mathbf{S}_{kj} = \frac{1}{y}\sum_{i=1}^{y} s(i,j), \qquad j = 1, ..., m \tag{3.9}$$

$$\mathbf{Q}_{j} = \frac{1}{y}\sum_{i=1}^{y} q(i,j), \qquad j = 1, ..., m \tag{3.10}$$

where $\mathbf{S}_k$ denotes the supervector of the kth segment, $\mathbf{Q}$ is the supervector for the query, y is the number of the frames of the query and m is the number of the phonetic classes. Each $\mathbf{S}_k$ supervector is a possible candidate for matching that gets filtered. As seen in the Figure 3.7, the document posteriorgram is divided into segments with overlap and each segment has the same length with the query. The overlap amount is fixed with 150 msec shift as an initial condition, however, another shift amount is also used in experiments to understand the effect. $\mathbf{S}_1$ and $\mathbf{S}_2$ denote the first and second supervector for the document segments. The algorithm generates the supervectors by summing the entries for each phonetic class from 1 to y and normalizing the sum. For the query supervector, same calculation is performed.

3.2.2. Cosine Similarity and Filtering

After the document supervectors are calculated, cosine distance is used for the distance measure to find the similarity between the document supervectors and the

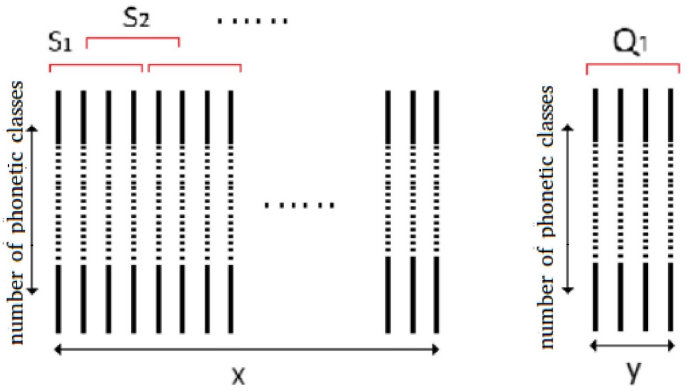

Figure 3.7. An illustration of supervector generation.

query supervectors. The angle between the two vectors will be the similarity measure to perform pre-filtering for unlikely matches. Note that since the supervectors have non-negative entries the similarity will always be non-negative.

$$\text{sim}(\mathbf{S}_k, \mathbf{Q}) = \frac{\sum_{j=1}^{m} \mathbf{S}_{kj} \times \mathbf{Q}_j}{\sqrt{\sum_{j=1}^{m} (\mathbf{S}_{kj})^2} \times \sqrt{\sum_{i=j}^{m} (\mathbf{Q}_j)^2}} \tag{3.11}$$

For each query, after we calculate the scores of all possible matches as in Equation 3.11, filtering is implemented by comparing the scores with a threshold value. The threshold value specifies the amount of filtering. The lower the threshold value, the more candidate matches are searched by DTW. For a coarse filtering, a low threshold value is sufficient whereas for a fine filtering, threshold value should be higher. The algorithm stores beginning points of segments, thus, after filtering segments are recomposed.

For the filtering step, the complexity is $\text{O}(k = \frac{x}{\delta})$, where the k is the number of supervectors of the document for the corresponding query, x is the length of the document and δ is the shift amount to generate overlapping submatrices. Since the number of the query supervector is 1, the filtering complexity only depends on the number of supervectors of the document. For instance, the complexity is O(234282 =

$\frac{3514259}{\delta}$) for the query "cay". The number of frames in 10 hours document posteriorgram is 3514259 and the δ equals to 150 msec shift. First of all, the algorithm extracts 234282 overlapping submatrices document posteriorgram. Afterwards, these submatrices are converted into supervectors and the query is also converted into a supervector. Cosine similarity is computed between the query supervector and the document supervectors. During the pre-filtering stage where the similarity is compared to a threshold value, most of the supervectors are eliminated. If n possible matches are detected after filtering, DTW will be implemented on a n-by-y^2 matrix, where the y denotes the length of the query. The complexity reduces from O(xy) to O(ny^2) since x is typically much larger than ny. For example, the complexity reduced from $3*10^6 \times 25 = 7.5 \times 10^7$ to 10×25^2 for the query "cay", where 25 is the number of frames in the query, and $3 * 10^6$ is the number of frames in the document. For this query, the pre-filtering step finds only $n = 10$ possible matches to perform dynamic time warping. Using the same perspective, the memory usage also decreases from O(xy) to O(ny^2).

3.3. Experimental Results

The experiments are conducted in BUSIM by using the IARPA Babel Turkish limited language pack (LimitedLP) dataset which contains 10 hours of conversational telephone speech. For keyword search evaluation, the 307 development queries, 88 (28.67%) of which are OOV, are used.

For the posteriorgram generation, Kaldi toolkit is used [65]. The document posteriorgram is generated using the DNN schemes for the language. This system uses a DNN with p-norm activations. Audio data are processed with this DNN network and a state level posteriorgram is formed. To obtain the phone posteriorgram, the posterior values of states corresponding to each phone are accumulated which yields a 43 dimensional posteriorgram. The text based queries are modeled by using average query modelling, which are explained in Subsection 2.3.3. For the evaluation of the performance of the system, the term weigted value is used, which is defined in NIST STD 2006 Evaluation Plan. This evaluation metric is presented in Subsection 2.1.4.

The experiments are conducted under various configurations in order to capture the behaviour of the system. The experimental results can be collected in two main groups. First of all, the individual results of the proposed approach are given. Then the results for the merged system, which is the combination of the baseline and proposed approach, are represented.

3.3.1. Individual Results of the Proposed System

The amount of filtering is specified by the threshold value. If the task requires a coarse filtering, a low threshold value is sufficient. However, if the task needs a fine filtering, the threshold value should be higher. The run time of the algorithm and the computational complexity of the DTW depend on the threshold value directly. Thus, in order to observe this relationship we use 8 different threshold values.

Table 3.1 and Table 3.2 show the MTWV, OTWV and STWV scores for individual results of proposed system. We use two different normalization method to compensate for the distance differences of queries having different length. The linear normalization (LN) basically maps all scores for a certain query on to the range [0, 1]. *STO* refers to sum-to-one (STO) normalization. The experiment, which is labeled with P_1, refers for the highest threshold value wheras the experiment, which is labeled with P_8, refers for the lowest threshold value. When the individual performances are concerned, the scores with linear normalization are higher than the scores with STO normalization for every threshold value. Also it can be seen that the MTWV scores increase until a point. After a point, the lower threshold value causes the coarse filtering. This causes participation of some candidate supervectors to search process, which have lower similarity with the query supervector. Thus, the MTWV scores start to decrease.

Table 3.1. Individual MTWV, OTWV and STWV scores of STO normalized experiments with different threshold values.

Exp. No	MTWV			OTWV			STWV		
	All	IV	OOV	All	IV	OOV	All	IV	OOV
P_1	0.0596	0.0746	0.0254	0.1430	0.1509	0.1230	0.1843	0.2016	0.1401
P_2	0.0775	0.1017	0.0274	0.2302	0.2438	0.1957	0.3194	0.3490	0.2439
P_3	0.0844	0.1043	0.0657	0.2739	0.2919	0.2278	0.4430	0.4815	0.3448
P_4	0.0996	0.1135	0.0785	0.2885	0.3056	0.2449	0.4765	0.5102	0.3906
P_5	**0.1024**	**0.1159**	**0.0813**	0.2919	0.3075	0.2521	0.4967	0.5312	0.4087
P_6	0.0952	0.1066	0.0810	0.2939	0.3082	0.2574	0.5171	0.5540	0.4228
P_7	0.0963	0.1077	0.0807	0.2942	0.3095	0.2553	0.5258	0.5642	0.4279
P_8	0.0966	0.1097	0.0806	**0.2965**	**0.3132**	**0.2539**	**0.5277**	**0.5669**	**0.4280**

In Figure 3.8 and Figure 3.9 for the proposed system without any combination, the MTWV of all, IV and OOV queries with respect to the run time are shown, where the run time for the no-filtering case is taken to be 1.0. The filtering time is included in these run times and it is 11 minutes for all threshold values. The run time of no-filtering DTW is approximately 30 hours. It can be seen that the run time for DTW increases according to the degree of threshold value. By setting the threshold appropriately, the run time can be reduced by approximately 8 times without significant loss in performance.

Table 3.2. Individual MTWV, OTWV and STWV scores of linear normalized experiments with different threshold values.

Exp. No	MTWV			OTWV			STWV		
	All	IV	OOV	All	IV	OOV	All	IV	OOV
P_1	0.0796	0.0942	0.0503	0.1482	0.1552	0.1306	0.1772	0.1885	0.1485
P_2	0.0995	0.1236	0.0769	0.2508	0.2655	0.2134	0.3315	0.3225	0.2773
P_3	0.1178	0.1401	0.1058	0.3058	0.3211	0.2669	0.4755	0.5165	0.3710
P_4	0.1266	0.1440	0.1152	0.3172	0.3322	0.2790	0.5167	0.5538	0.4221
P_5	0.1338	**0.1535**	0.1221	0.3199	0.3331	0.2861	0.5454	0.5844	0.4458
P_6	0.1321	0.1524	**0.1226**	0.3216	0.3347	0.2880	0.6068	0.6377	0.5279
P_7	0.1340	0.1496	0.1212	0.3244	**0.3397**	0.2859	0.6498	0.6822	0.5672
P_8	**0.1344**	0.1501	0.1202	**0.3245**	0.3379	**0.2901**	**0.6703**	**0.7009**	**0.5926**

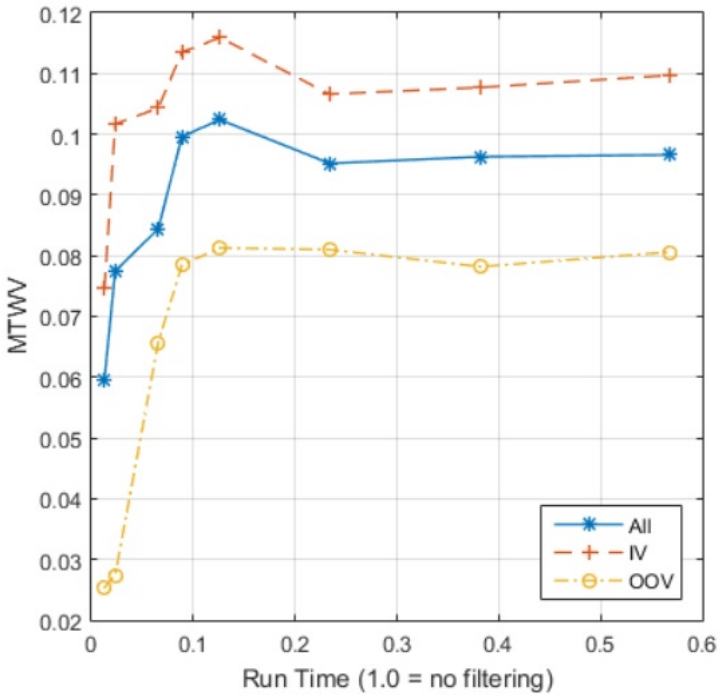

Figure 3.8. MTWV of STO normalized individual system for all, IV and OOV queries.

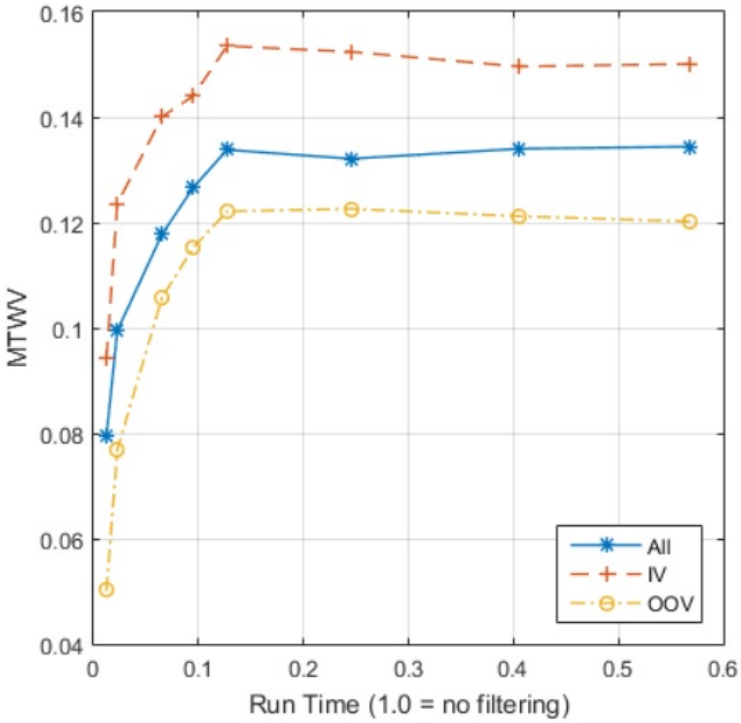

Figure 3.9. MTWV of linear normalized individual system for all, IV and OOV queries.

3.3.2. Merged Results with the LVCSR Lattice Based System

Main contribution of this thesis is to improve the performance of a large vocabulary continuous speech recognition based keyword search using our proposed system. Thus, before we introduce the result of merged system, we should describe the approaches of the baseline system.

The baseline system uses LVCSR, which are processed by lattice indexing for IV queries. The lattices are converted from individual WFST to a single generalized factor transducer structure. Lattice indexing is described in Subsection 2.2.2 in detail. Given a keyword, the system generates a finite state machine. Then, it composes the keyword with the factor transducer to obtain all occurences of the keyword along with identity, position and posterior probability of the query term. The baseline uses proxy

keywords to cope with OOV queries. The basic idea is to find acoustically similar IV terms for the OOV terms and use these IV terms as proxy keywords [37]. The WFST procedure for generating a proxy keyword K' can be described as:

$$K' = Project(ShortestPath(K \circ L_2 \circ E \circ (L_1^*)^{-1})) \tag{3.12}$$

where K represents the original keyword, L_2 denotes a lexicon that contains the pronunciation of K and L_1 is the original lexicon. E is an edit distance transducer that maps a phone sequence to other phone sequence and K' is a WFST which contains several IV terms that are acoustically similar to K.

Table 3.3. MTWV, OTWV and STWV results for LVCSR lattice based system (L) and merged (L+P) systems with STO normalized experiments.

System	MTWV			OTWV			STWV		
	All	IV	OOV	All	IV	OOV	All	IV	OOV
L	0.3745	0.4500	0.1887	0.4157	0.4896	0.2240	0.6624	0.7567	0.4220
L+P_1	0.3655	0.4388	0.1978	0.4459	0.5111	0.2796	0.6806	0.7665	0.4614
L+P_2	0.3766	0.4442	0.2326	0.4661	0.5212	0.3256	0.7055	0.7813	0.5121
L+P_3	0.3859	0.4484	0.2524	0.4854	0.5440	0.3361	0.7323	0.8006	0.5580
L+P_4	0.3906	0.4500	0.2582	0.4876	0.5449	0.3413	0.7411	0.8019	0.5861
L+P_5	0.3961	0.4577	0.2616	0.4876	0.5441	0.3434	0.7451	0.8068	0.5879
L+P_6	0.3982	0.4600	0.2627	0.4889	0.5444	0.3474	0.7472	0.8073	0.5938
L+P_7	0.3978	0.4596	0.2655	**0.4916**	**0.5476**	**0.3489**	**0.7495**	**0.8105**	**0.5938**
L+P_8	**0.3987**	**0.4603**	**0.2655**	0.4909	0.5469	0.3482	0.7495	0.8105	0.5938

To evaluate the merged scores, the CombMNZ method [66] is used. For each hit, the scores of the corresponding hits from different systems are summed and normalized by the number of systems. The start time and duration are obtained from the highest-scoring hit. Table 3.3 and Table 3.4 show MTWV,OTWV and STWV scores for LVCSR lattice based system (L) and merged (L+P) systems with STO and linear normalized experiments, respectively. Since to combine two system requires STO nor-

malization, scores with STO normalization are higher than linear normalization. We can see from Table 3.3, the proposed system improves the OOV MTWV score from 0.1887 to 0.1978 and similar gains are observed in the OTWV and STWV even with a 22 minutes DTW. There is a trade off between the complexity of DTW and improvement of the system. As the threshold decreases, the run time increases. However, system performance increases as well.

Table 3.4. MTWV, OTWV and STWV results for LVCSR lattice based system (L) and merged (L+P) systems with linear normalized experiments.

System	MTWV			OTWV			STWV		
	All	IV	OOV	All	IV	OOV	All	IV	OOV
L	**0.3745**	**0.4500**	**0.1887**	0.4157	**0.4896**	0.2240	0.6624	0.7567	0.4220
L+P_1	0.2689	0.3292	0.1259	0.4163	0.4722	0.2738	0.6838	0.7661	0.4739
L+P_2	0.2282	0.2727	0.1436	0.4216	0.4686	0.3018	0.7193	0.7907	0.5371
L+P_3	0.2307	0.2686	0.1388	0.4206	0.4591	0.3222	0.7519	0.8191	0.5806
L+P_4	0.2373	0.2766	0.1458	0.4249	0.4616	0.3312	0.7668	0.8214	0.6275
L+P_5	0.2300	0.2686	0.1361	0.4237	0.4579	0.3365	0.7668	0.8214	0.6275
L+P_6	0.2206	0.2554	0.1489	0.4184	0.4484	0.3420	0.7795	0.8273	0.6577
L+P_7	0.2112	0.2448	0.1382	**0.4252**	0.4570	**0.3442**	0.8001	0.8445	0.6869
L+P_8	0.2061	0.2371	0.1271	0.4240	0.4558	0.3429	**0.8091**	**0.8473**	**0.7119**

Figure 3.10 and Figure 3.11 represent the MTWV of all, IV and OOV queries of merged system with respect to run time, where the run time for the no-filtering case is taken to be 1.0. The filtering time is included in these run times and it is 11 minutes for all threshold values. The run time of no-filtering DTW is approximately 30 hours. The black lines show the baseline MTWV score for all, IV and OOV queries. By setting the threshold appropriately, the run time can be reduced by approximately 8 times and the performance of the baseline can be improved without significant loss. We can see the significant improvement for OOV queries from the Figure 3.10.

Another performance evaluation metric is DET curve, which represents the dependency of false alarm probability to the miss probability with all possible values of the threshold. Figure 3.12 represents the DET curves of the baseline and STO normal-

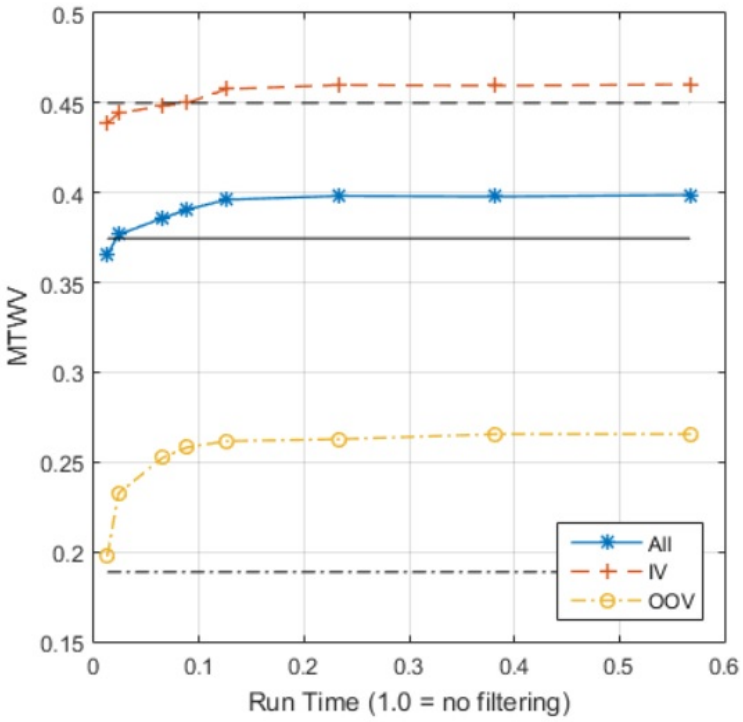

Figure 3.10. MTWV of STO normalized merged system for all, IV and OOV queries.

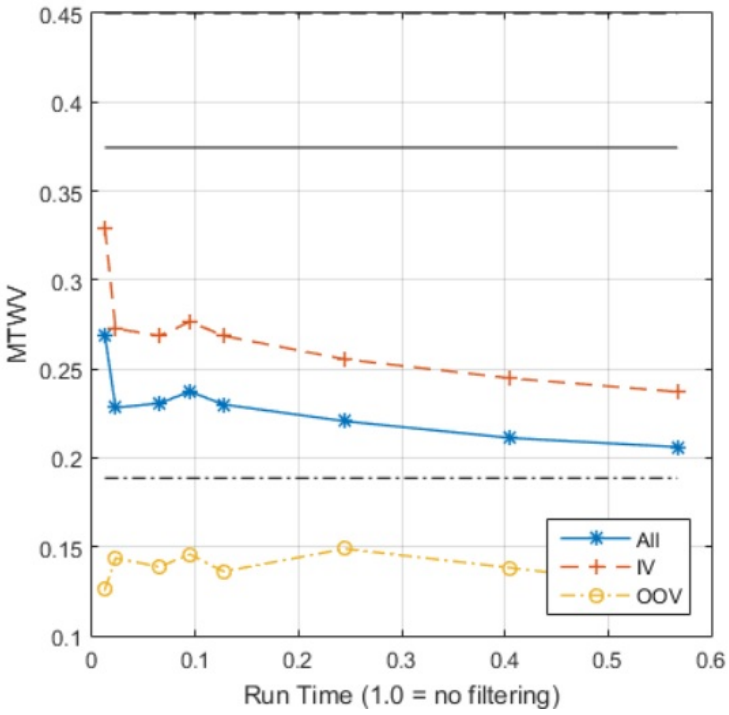

Figure 3.11. MTWV of linear normalized merged system for all, IV and OOV queries.

ized combination of the two system. The smooth red line on the upper right corner shows the random performance. Thus, DET curve should be as close as possible to the black curved line on the bottom left corner. The green line represents the DET curve of the baseline system. The DET curve of the combination of two system, which is the baseline and the proposed system, is represented with the blue line. Accordingly, it can be seen that the performance of the merged system is higher than the baseline system since the blue line is closer to the bottom left corner.

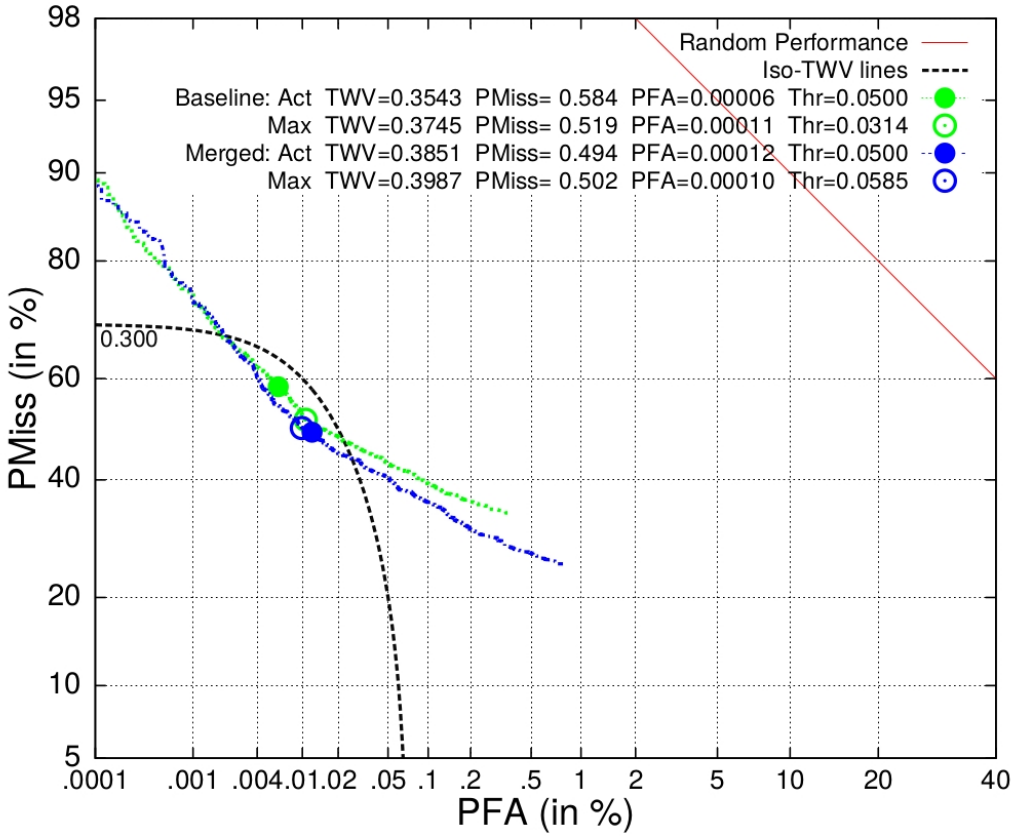

Figure 3.12. DET curve of the baseline and STO normalized merged system.

The supervector generation for the experiments above is performed with 150 msec overlap and the number of frames of the submatrices are equal to the number of frames with the queries. In other words, once a query is given to the system, the proposed algorithm divides the document posteriorgram into query sized submatrices with 150 msec shift and turns them into supervectors. To understand the effect of

these two important parameters on the performance, some additional experiments are conducted for 3 different threshold values. Table 3.5 shows the alternative segment size and shift amount that are used for the experiments. First line is our base condition for earlier experiments. Additionaly, we use two different configurations. First of all, the amount of the overlap is fixed to 150 msec shift and the document is divided into fixed sized windows, that is 60, the average number of frames of all 307 queries. For instance, if the number of frame is 100 for a query, the document posteriorgram is divided into segments with 150 msec shift, which have 60 frames for each segment. Secondly, the shift amount is half of the query size instead of 150 msec. For instance, if the number of frame is 100 for a query, then the shift is 50 for the corresponding query and the document posteriorgram is divided into segments have 100 frames. We label these two different experiments with fixed window (F) and 50% overlap (O), respectively. Same threshold values are used to compare results of these two configurations with the experiments above.

Table 3.5. Configurations for the experiments.

Config.	**Segment Size**	**Shift (δ)**
P	variable (query-sized)	fixed (150 msec)
F	fixed (average frame number of all queries= 60)	fixed (150 msec)
O	variable (query-sized)	variable (half of the query length)

Table 3.6 shows MTWV, OTWV and STWV results, where L denotes the LVCSR lattice based system and L+P, L+F, L+O represent combinations of the two methods. Subscripts denote the different threshold values for filtering step. The highest threshold is denoted by 1 and the lowest threshold is denoted by 8. The results show that configuration F, which has fixed segment size and fixed shift for the overlap, has the best all, IV and OOV MTWV scores for the lowest threshold. Figure 3.13 represents the MTWV of all, IV and OOV queries of the merged system with respect to run time, where the run time for the no-filtering case is taken to be 1.0. Blue bars represent the L+P results, red bars represent the L+F results and green bars denote the L+O results. Since the threshold value specifies the amount of filtering, the run time increases as the threshold decreases. However, the run time of configuration F, which is represented

with red bars, is the shortest among the other run times for all 3 threshold values. Thus, instead of dividing the document posteriorgram into query-sized segments, we can use fixed window to generate the supervectors that gets filtered.

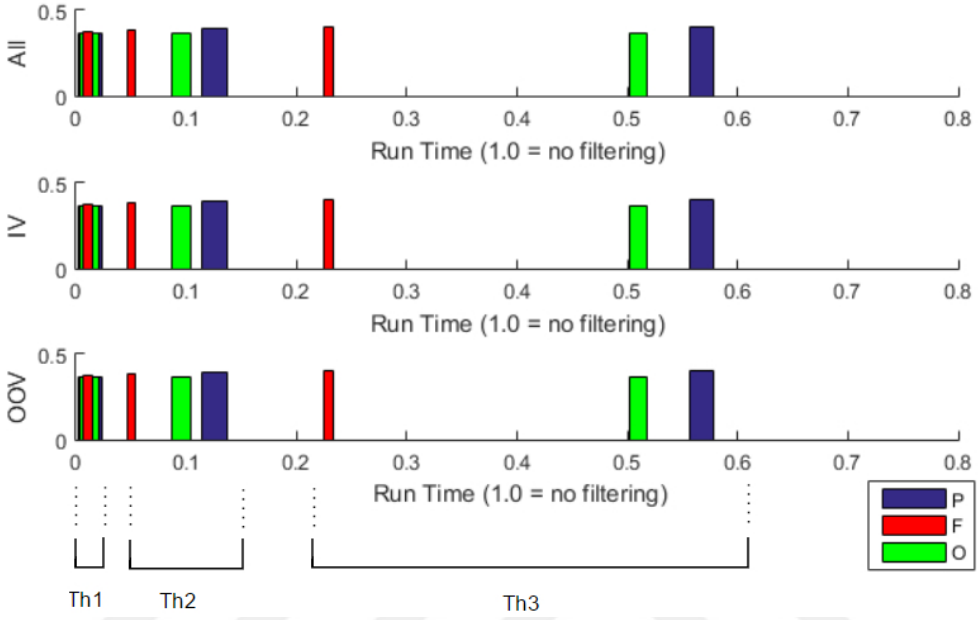

Figure 3.13. MTWV of all, IV and OOV queries of the merged system for 3 different thresholds.

Table 3.6. Merged MTWV, OTWV and STWV results for fixed window approach (L+F), 50% overlap approach (L+O) and base conditions (L+P).

System	MTWV			OTWV			STWV		
	All	IV	OOV	All	IV	OOV	All	IV	OOV
L	0.3745	0.4500	0.1887	0.4157	0.4896	0.2240	0.6624	0.7567	0.4220
$L+P_1$	0.3655	0.4388	0.1978	0.4459	0.5111	0.2796	0.6806	0.7665	0.4614
$L+P_5$	0.3961	0.4577	0.2616	0.4876	0.5441	0.3434	0.7451	0.8068	0.5879
$L+P_8$	0.3987	0.4603	0.2655	**0.4909**	**0.5469**	0.3482	**0.7495**	**0.8105**	**0.5938**
$L+F_1$	0.3732	0.4430	0.2107	0.4353	0.5018	0.2657	0.6794	0.7668	0.4562
$L+F_5$	0.3864	0.4501	0.2412	0.4789	0.5334	0.3399	0.7302	0.7952	0.5646
$L+F_8$	**0.3991**	**0.4619**	**0.2662**	0.4894	0.5447	**0.3485**	0.7476	0.8079	0.5938
$L+O_1$	0.3645	0.4381	0.2065	0.4323	0.4965	0.2683	0.6776	0.7623	0.4614
$L+O_5$	0.3631	0.4373	0.2199	0.4628	0.5174	0.3236	0.7190	0.7845	0.5518
$L+O_8$	0.3695	0.4386	0.2181	0.4704	0.5266	0.3273	0.7278	0.7894	0.5706

4. CONCLUSIONS AND FUTURE WORK

In this thesis, we adressed a real world challenge for spoken term detection task when there is not enough resources for the target language. To address this problem, we use the query by example spoken term detection as a specific scenerio. In order to build the framework, we implement a STD system using template based dynamic time warping algorithm with a pre-filtering step to decrease the complexity of the algorithm.

The work discussed in this thesis is motivated by reducing the computational complexity of dynamic time warping algorithm by using the pre-filtering step. The basic idea of our method is to divide the document posteriorgram into segments, or submatrices, then to perform DTW only on possible candidates for a match instead of searching whole document posteriorgram. Once a query is given, we divide the document into overlapping segments, which have the same size as the query posteriorgram. Then we calculate supervectors from these segments, by simply summing the entries for each phonetic class and normalizing the sum. Same calculation is performed on the query posteriorgram as well. Cosine distance is used to measure the similarity between supervectors of the segments and the query supervector. This distance is calculated for every supervector of the segmental posteriorgram and the query posteriorgram. The angle between the two vectors will be the similarity measure to perform pre-filtering for unlikely matches. Filtering is implemented by comparing the scores with a threshold value, which specifies the amount of filtering. Since the algorithm stores the beginning points of the segments, after filtering step, supervectors of the segments and the queries are reformed into posteriorgrams. DTW is run only on these segment posteriorgrams instead of the whole posteriorgram.

For the experiments, we use the IARPA Babel Turkish limited language pack (LimitedLP) dataset which contains 10 hours of conversational telephone speech. For keyword search evaluation, the 307 development queries, 88 (28.67%) of which are OOV, are used. Kaldi toolkit is used to generate the posteriorgrams. The audio data

are processed with a deep neural network and a state level posteriorgram is formed. Then the posterior values of states corresponding to each phone are accumulated which yields a 43 dimensional posteriorgram. The text based queries are modeled by using average query modelling. The term weigted value is used to evaluate the performance of the system.

The experiments are conducted under various configurations and threshold values in order to capture the performance of the system. We use 8 different threshold values, which specifies the amount of filtering. The threshold value is low for the tasks requires a coarse filtering wheras a higher threshold value is used for fine filtering. Since the size of the segments and overlap amount are important parameters for the computational complexity, 3 different configurations are taken into account. First one is the base condition, fixed overlap amount (150 msec) and query-sized segments. The other one is fixed window, that is, fixed segment size with the segments overlapping 150 msec. Last scenerio is to use 50% overlap for supervector calculation with query-sized segments. Two different normalization methods are used, linear normalization and STO normalization.

Experimental results indicate that, better scores are obtained using linear normalization for the individual performance. For the combined performance, on the other hand, STO normalization performs better. For different threshold values, MTWV increases until a point, then the increase stops after this point even if the threshold value reduces. Thus, by setting the threshold appropriately, the run time can be reduced by approximately 8 times without significant loss in performance. Also, results show that fixed window approach is the best performing approach for the selection of segment size in supervector generation, since it has the minimum run time compared to the other configurations for the same threshold value and has almost same MTWV score for all, IV and OOV queries.

Main contribution of this work is to improve the LVCSR lattice based system by combining our scores with the baseline. Experimental results show that, merged system with STO normalization increase the MTWV performance of the baseline from

0.3745 to 0.3987 (%6.46) for all terms, from 0.4500 to 0.4603 (%2.29) for IV terms and from 0.1887 to 0.2655 (%40.7) for OOV terms.

Although the work achieved in this thesis shows very promising results, there is still room for the improvement of the both individual and merged results. As a future work, experiments can be conducted on the other low resource languages such as Swahili and Javanese. Moreover, different fixed window size and overlap percentages can be investigated to optimize the run time and the performance. In this study, we use only average query modelling technique for the text based queries, also other query modelling methods can be applied to understand the effects of different modelling approaches on the performance.

REFERENCES

1. Chelba, C., T. J. Hazen, B. Ramabhadran and M. Saraçlar, "Speech Retrieval", *Spoken Language Understanding: Systems for Extracting Semantic Information from Speech*, pp. 417–446, 2011.

2. Glass, J. R., T. J. Hazen, D. S. Cyphers, K. Schutte and A. Park, "The MIT spoken lecture processing project", *Proceedings of HLT/EMNLP on Interactive Demonstrations*, pp. 28–29, Association for Computational Linguistics, 2005.

3. Allauzen, C., M. Mohri and M. Saraclar, "General indexation of weighted automata: application to spoken utterance retrieval", *Proceedings of the Workshop on Interdisciplinary Approaches to Speech Indexing and Retrieval at HLT-NAACL 2004*, pp. 33–40, Association for Computational Linguistics, 2004.

4. Fiscus, J. G., J. Ajot, J. S. Garofolo and G. Doddington, "Results of the 2006 spoken term detection evaluation", *Proc. SIGIR*, Vol. 7, pp. 51–57, 2007.

5. Wilcox, L. D. *et al.*, "HMM-based wordspotting for voice editing and indexing", ESCA, 1991.

6. James, D. A. and S. J. Young, "A fast lattice-based approach to vocabulary independent wordspotting", *Acoustics, Speech, and Signal Processing, 1994. ICASSP-94., 1994 IEEE International Conference on*, Vol. 1, pp. I–377, IEEE, 1994.

7. Larson, M. and G. J. Jones, "Spoken content retrieval: A survey of techniques and technologies", *Foundations and Trends in Information Retrieval*, Vol. 5, No. 4—5, pp. 235–422, 2012.

8. Wang, D., *Out-of-vocabulary spoken term detection*, Ph.D. Thesis, The University of Edinburgh, 2010.

9. Manning, C. D., P. Raghavan, H. Schütze *et al.*, *Introduction to information retrieval*, Vol. 1, Cambridge university press Cambridge, 2008.

10. Lee, L.-s., J. Glass, H.-y. Lee and C.-a. Chan, "Spoken Content Retrieval—Beyond Cascading Speech Recognition with Text Retrieval", *Audio, Speech, and Language Processing, IEEE/ACM Transactions on*, Vol. 23, No. 9, pp. 1389–1420, 2015.

11. Szöke, I., "Hybrid word-subword spoken term detection", *Faculty of Information Technology BUT*, 2010.

12. Rose, R., "Techniques for information retrieval from speech messages", *Lincoln Laboratory Journal*, Vol. 4, No. 1, pp. 45–60, 1991.

13. Rose, R. C., E. I. Chang and R. P. Lippmann, "Techniques for information retrieval from voice messages", *Acoustics, Speech, and Signal Processing, 1991. ICASSP-91., 1991 International Conference on*, pp. 317–320, IEEE, 1991.

14. James, D. A., *The application of classical information retrieval techniques to spoken documents*, Ph.D. Thesis, 1995.

15. Robertson, S. E. and S. Walker, "Some simple effective approximations to the 2-poisson model for probabilistic weighted retrieval", *Proceedings of the 17th annual international ACM SIGIR conference on Research and development in information retrieval*, pp. 232–241, Springer-Verlag New York, Inc., 1994.

16. Van Thong, J.-M., P. J. Moreno, B. Logan, B. Fidler, K. Maffey and M. Moores, "Speechbot: an experimental speech-based search engine for multimedia content on the web", *Multimedia, IEEE Transactions on*, Vol. 4, No. 1, pp. 88–96, 2002.

17. Chelba, C., T. J. Hazen and M. Saraçlar, "Retrieval and browsing of spoken content", *Signal Processing Magazine, IEEE*, Vol. 25, No. 3, pp. 39–49, 2008.

18. Ng, K. and V. W. Zue, "Subword-based approaches for spoken document re-

trieval", *Speech Communication*, Vol. 32, No. 3, pp. 157–186, 2000.

19. Deligne, S. and F. Bimbot, "Language modeling by variable length sequences: Theoretical formulation and evaluation of multigrams", *Acoustics, Speech, and Signal Processing, 1995. ICASSP-95., 1995 International Conference on*, Vol. 1, pp. 169–172, IEEE, 1995.

20. Logan, B., J.-M. V. Thong and P. J. Moreno, "Approaches to reduce the effects of OOV queries on indexed spoken audio", *Multimedia, IEEE Transactions on*, Vol. 7, No. 5, pp. 899–906, 2005.

21. Turunen, V. T. and M. Kurimo, "Indexing confusion networks for morph-based spoken document retrieval", *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, pp. 631–638, ACM, 2007.

22. Foote, J., "An overview of audio information retrieval", *Multimedia systems*, Vol. 7, No. 1, pp. 2–10, 1999.

23. Jones, G. J., J. T. Foote, K. S. Jones and S. J. Young, "Retrieving spoken documents by combining multiple index sources", *Proceedings of the 19th annual international ACM SIGIR conference on Research and development in information retrieval*, pp. 30–38, ACM, 1996.

24. Saraclar, M. and R. Sproat, "Lattice-based search for spoken utterance retrieval", *Urbana*, Vol. 51, p. 61801, 2004.

25. Juang, B.-H., W. Hou and C.-H. Lee, "Minimum classification error rate methods for speech recognition", *Speech and Audio Processing, IEEE Transactions on*, Vol. 5, No. 3, pp. 257–265, 1997.

26. Povey, D. and P. C. Woodland, "Minimum phone error and I-smoothing for improved discriminative training", *Acoustics, Speech, and Signal Processing*

(ICASSP), 2002 IEEE International Conference on, Vol. 1, pp. I–105, IEEE, 2002.

27. Povey, D., *Discriminative training for large vocabulary speech recognition*, Ph.D. Thesis, University of Cambridge, 2005.

28. Veselỳ, K., A. Ghoshal, L. Burget and D. Povey, "Sequence-discriminative training of deep neural networks.", *INTERSPEECH*, pp. 2345–2349, 2013.

29. Jansen, A. and P. Niyogi, "Point process models for spotting keywords in continuous speech", *Audio, Speech, and Language Processing, IEEE Transactions on*, Vol. 17, No. 8, pp. 1457–1470, 2009.

30. Liu, C., A. Jansen, G. Chen, K. Kintzley, J. Trmal and S. Khudanpur, "Low-resource open vocabulary keyword search using point process models", *Fifteenth Annual Conference of the International Speech Communication Association*, 2014.

31. Lee, H.-y., C.-p. Chen and L.-s. Lee, "Integrating recognition and retrieval with relevance feedback for spoken term detection", *Audio, Speech, and Language Processing, IEEE Transactions on*, Vol. 20, No. 7, pp. 2095–2110, 2012.

32. Lee, H.-y., C.-p. Chen, C.-f. Yeh and L.-s. Lee, "Improved spoken term detection by discriminative training of acoustic models based on user relevance feedback", *Eleventh Annual Conference of the International Speech Communication Association*, 2010.

33. Lee, H.-y., C.-P. Chen, C.-f. Yeh and L.-s. Lee, "A framework integrating different relevance feedback scenarios and approaches for spoken term detection", *Spoken Language Technology Workshop (SLT), 2010 IEEE*, pp. 389–394, IEEE, 2010.

34. Cui, J., X. Cui, B. Ramabhadran, J.-H. Kim, B. Kingsbury, J. Mamou, L. Mangu, M. Picheny, T. N. Sainath and A. Sethy, "Developing speech recognition systems for corpus indexing under the IARPA Babel program", *Acoustics, Speech and*

Signal Processing (ICASSP), 2013 IEEE International Conference on, pp. 6753–6757, IEEE, 2013.

35. Gandhe, A., F. Metze, A. Waibel and I. Lane, "Optimization of Neural Network Language Models for keyword search", *Acoustics, Speech and Signal Processing (ICASSP), 2014 IEEE International Conference on*, pp. 4888–4892, IEEE, 2014.

36. Saraclar, M., 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", *Automatic Speech Recognition and Understanding (ASRU), 2013 IEEE Workshop on*, pp. 464–469, IEEE, 2013.

37. Chen, G., O. Yilmaz, J. Trmal, D. Povey and S. Khudanpur, "Using proxies for OOV keywords in the keyword search task", *Automatic Speech Recognition and Understanding (ASRU), 2013 IEEE Workshop on*, pp. 416–421, IEEE, 2013.

38. Karanasou, P., L. Burget, D. Vergyri, M. Akbacak and A. Mandal, "Discriminatively trained phoneme confusion model for keyword spotting.", *INTERSPEECH*, 2012.

39. Wallace, R., R. Vogt, B. Baker and S. Sridharan, "Optimising figure of merit for phonetic spoken term detection", *Acoustics Speech and Signal Processing (ICASSP), 2010 IEEE International Conference on*, pp. 5298–5301, IEEE, 2010.

40. Wallace, R., B. Baker, R. Vogt and S. Sridharan, "Discriminative optimization of the figure of merit for phonetic spoken term detection", *Audio, Speech, and Language Processing, IEEE Transactions on*, Vol. 19, No. 6, pp. 1677–1687, 2011.

41. Rohlicek, J. R., W. Russell, S. Roukos and H. Gish, "Continuous hidden Markov modeling for speaker-independent word spotting", *Acoustics, Speech, and Signal Processing, 1989. ICASSP-89., 1989 International Conference on*, pp. 627–630, IEEE, 1989.

42. Tejedor, J., M. Fapšo, I. Szöke, J. Černockỳ, F. Grézl *et al.*, "Comparison of methods for language-dependent and language-independent Query-by-Example spoken term detection", *ACM Transactions on Information Systems (TOIS)*, Vol. 30, No. 3, p. 18, 2012.

43. Wang, H., *Query-by-Example Spoken Term Detection for Low-resource Languages*, Ph.D. Thesis, Chinese University of Hong Kong, 2014.

44. Joder, C., F. Weninger, M. Wöllmer and B. Schuller, "The TUM cumulative DTW approach for the MediaEval 2012 spoken web search task", *Proc. MediaEval*, 2012.

45. Muscariello, A., G. Gravier and F. Bimbot, "Zero-resource audio-only spoken term detection based on a combination of template matching techniques", *INTERSPEECH 2011: 12th Annual Conference of the International Speech Communication Association*, 2011.

46. Abad, A., R. F. Astudillo and I. Trancoso, "The L2F Spoken Web Search system for Mediaeval 2012.", *MediaEval*, 2012.

47. Szöke, I., M. Fapso and K. Veselỳ, "BUT2012 Approaches for Spoken Web Search-MediaEval 2012.", *MediaEval*, 2012.

48. Mantena, G., S. Achanta and K. Prahallad, "Query-by-example spoken term detection using frequency domain linear prediction and non-segmental dynamic time warping", *Audio, Speech, and Language Processing, IEEE/ACM Transactions on*, Vol. 22, No. 5, pp. 946–955, 2014.

49. Lee, C.-y. and J. Glass, "A nonparametric Bayesian approach to acoustic model discovery", *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers-Volume 1*, pp. 40–49, Association for Computational Linguistics, 2012.

50. Huijbregts, M., M. McLaren and D. Van Leeuwen, "Unsupervised acoustic sub-

word unit detection for query-by-example spoken term detection", *Acoustics, Speech and Signal Processing (ICASSP), 2011 IEEE International Conference on*, pp. 4436–4439, IEEE, 2011.

51. Sakoe, H. and S. Chiba, "Dynamic programming algorithm optimization for spoken word recognition", *Acoustics, Speech and Signal Processing, IEEE Transactions on*, Vol. 26, No. 1, pp. 43–49, 1978.

52. Zhang, Y. and J. R. Glass, "Unsupervised spoken keyword spotting via segmental DTW on Gaussian posteriorgrams", *Automatic Speech Recognition & Understanding, 2009. ASRU 2009. IEEE Workshop on*, pp. 398–403, IEEE, 2009.

53. Yi, B.-K., H. Jagadish and C. Faloutsos, "Efficient retrieval of similar time sequences under time warping", *Data Engineering, 1998. Proceedings., 14th International Conference on*, pp. 201–208, IEEE, 1998.

54. Kim, S.-W., S. Park and W. W. Chu, "An index-based approach for similarity search supporting time warping in large sequence databases", *Data Engineering, 2001. Proceedings. 17th International Conference on*, pp. 607–614, IEEE, 2001.

55. Zhang, Y., *Unsupervised speech processing with applications to query-by-example spoken term detection*, Ph.D. Thesis, Massachusetts Institute of Technology, 2013.

56. Zhang, Y. and J. R. Glass, "A Piecewise Aggregate Approximation Lower-Bound Estimate for Posteriorgram-Based Dynamic Time Warping.", *INTERSPEECH*, pp. 1909–1912, 2011.

57. Jansen, A., K. Church and H. Hermansky, "Towards spoken term discovery at scale with zero resources.", *INTERSPEECH*, pp. 1676–1679, 2010.

58. Jansen, A. and B. Van Durme, "Efficient spoken term discovery using randomized algorithms", *Automatic Speech Recognition and Understanding (ASRU), 2011 IEEE Workshop on*, pp. 401–406, IEEE, 2011.

59. Jansen, A. and B. Van Durme, "Indexing Raw Acoustic Features for Scalable Zero Resource Search.", *Interspeech*, 2012.

60. Itakura, F., "Minimum prediction residual principle applied to speech recognition", *Acoustics, Speech and Signal Processing, IEEE Transactions on*, Vol. 23, No. 1, pp. 67–72, 1975.

61. Salvador, S. and P. Chan, "FastDTW: Toward accurate dynamic time warping in linear time and space", *KDD workshop on mining temporal and sequential data*, 2004.

62. Müller, M., *Information retrieval for music and motion*, Vol. 2, Springer, 2007.

63. Chu, S., E. J. Keogh, D. M. Hart, M. J. Pazzani *et al.*, "Iterative Deepening Dynamic Time Warping for Time Series.", *SDM*, pp. 195–212, SIAM, 2002.

64. Keogh, E. J. and M. J. Pazzani, "Scaling up dynamic time warping for datamining applications", *Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 285–289, ACM, 2000.

65. Povey, D., A. Ghoshal, G. Boulianne, L. Burget, O. Glembek, N. Goel, M. Hannemann, P. Motlíček, Y. Qian, P. Schwarz *et al.*, "The Kaldi speech recognition toolkit", , 2011.

66. Mamou, J., J. Cui, X. Cui, M. J. Gales, B. Kingsbury, K. Knill, L. Mangu, D. Nolden, M. Picheny, B. Ramabhadran *et al.*, "System combination and score normalization for spoken term detection", *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*, pp. 8272–8276, IEEE, 2013.