KEYWORD SEARCH FOR LOW RESOURCE LANGUAGES

by

M.Batuhan Gündoğdu

B.S., Electrical and Electronics Engineering, Turkish Naval Academy, 2006

M.S., Electrical Engineering, University of Southern California, 2012

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

Graduate Program in Electrical and Electronics Engineering

Boğaziçi University

2017

to my wonderful wife, Natalie J.Reyes

ACKNOWLEDGEMENTS

There's no simple expression to define the privileged PhD experience I had under the supervision of Prof. Murat Saraçlar. He guided, mentored and supported me on several occasions, not only on my studies but also the hardships of life itself. I am grateful for his never-ending encouragements and understanding. His extraordinary brilliance always impressed me. His "magic touches" on the many times I got stuck with the implementation, kept the work going. It is definitely a privilege to work with him.

I would like to thank Prof. Levent Arslan and Assoc. Prof. Engin Erzin for their invaluable time and advises over the progress of this dissertation. I would also like to thank Prof. Nafiz Arıca and Assoc. Prof. Burak Acar for participating in my defense jury. I owe special thanks to Mr.Bọlaji Yusuf, for "automating" the procedures explained in this thesis, so that quick and organized experiments were made possible.

I should also thank my lab-mates from BUSIM and WCL, Görkem Ülkar, Gözde Çetinkaya, Leda Sarı, Alican Gök, Jülide Gülen Alaydın, Merve Ünlü, Mehmet Yamaç and Can Gürsoy for their invaluable friendship and making my time at Boğaziçi University a wonderful experience. Also, my colleagues Hasan Ateş, Recep Doğa Siyli, Bahri Maraş, Deniz Kumlu and Kubilay Savçı have given a tremendous support throughout this journey with their presence and friendship.

I completed the majority of my PhD work while I was working at the Istanbul Naval Shipyard. I am grateful to my supervisors, Mr.Cihan Ağaçayak, Mr.Fatih Ayyıldız and Dr.Birol Ülker for letting me take one day off every single week, to attend my courses and meetings with my advisor.

The datasets used in this thesis are provided by Intelligence Advanced Research Projects Activity (IARPA) Babel program which also partly supported the studies conducted for this thesis.

This study was supported in part by the Scientific and Technological Research Council of Turkey (TUBITAK) under Project 116E076. I should also thank TUBITAK-Directorate of Science Fellowships and Grant Programs (BİDEB) for supporting me with the 2224-A travel grant to present my work at the 2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), which was held in New Orleans, USA.

I owe a debt of gratitude to my wonderful parents Nil and Tümay Gündoğdu as well as my beautiful siblings Nilay and Berkehan Gündoğdu for always believing in me and their eternal support in making me who I am now.

And, the last but not least, my precious wife, mi amor, Natalie J. Reyes. I cannot thank you enough for the color you give to my life. I can never forget your sacrifices by agreeing to share me with my studies. I honestly wouldn't have been able to go through this long journey, if it weren't for your support.

ABSTRACT

KEYWORD SEARCH FOR LOW RESOURCE LANGUAGES

Retrieval of spoken content is one key endeavor, not only for finding the speech parts of interest, but also for an automated and facilitated speech mining towards better automatic speech recognition (ASR) systems. In particular, keyword search (KWS) systems aims to address these goals, by locating the specific parts of speech where a user provided keyword uttered. The most intuitive and convenient method for keyword search is to obtain text transcriptions from speech using ASR systems, and then conduct text based search on this ASR output. However, for low resource languages, for which available labeled speech training data is not sufficient, reliable ASR systems cannot be built and, KWS systems that depend on them will fail. Furthermore, if the keyword of interest is not within the vocabulary of the ASR system, it can never be found in the word level transcriptions. In this thesis, we address the above mentioned issues of KWS for the low resource languages. We aim to build a KWS system, using a completely different approach, with ideas inspired by the similarity search techniques of the query by example retrieval tasks. For this, we utilize a subsequence dynamic time warping-based search, after artificially modeling "pseudo examples" for text queries. Furthermore, we investigate a joint learning of these query representations and a proper distance metric for use in dynamic time warping. We show that, this new KWS system, we propose, outperforms the state of the art KWS techniques for retrieval of out-of-vocabulary terms, and provides significant improvements when combined with the conventional ASR-based KWS system due to its heterogeneity.

ÖZET

KISITLI KAYNAKLI DİLLERDE ANAHTAR SÖZCÜK ARAMA

Konuşma geri getirimi, yalnızca ilgilenilen konuşma parçalarının bulunması için değil aynı zamanda daha iyi otomatik konuşma tanıma (OKT) sistemlerinin kurulabilmesine yönelik, otomatikleştirilmiş ve kolaylaştırılmış bir konuşma madenciliği için, önemli bir problemdir. Bilhassa, anahtar sözcük arama (ASA) sistemleri, bir kullanıcının sağladığı anahtar sözcüğün telaffuz edildiği belirli kısımları bulmak suretiyle bu hedefleri gerçekleştirmeyi amaçlamaktadır. Anahtar sözcük arama için en akla yatan ve en çok kullanılan yöntem, OKT sistemleri kullanarak konuşmadan metin yazıları elde etmek ve bu OKT çıktısında metin tabanlı arama yapmaktır. Öte yandan, mevcut etiketli konuşma eğitim verilerinin yetersiz olduğu kısıtlı kaynaklı diller için güvenilir OKT sistemleri oluşturulamayacak ve kendilerine bağımlı ASA sistemleri başarısız olacaktır. Ayrıca, ilgilenilen anahtar sözcük OKT sisteminin dağarcığında yer almıyorsa, kelime düzeyi OKT çıktılarında bulunması imkansız olacaktır. Bu tezde, kısıtlı kaynaklı diller için ASA'nın yukarıda bahsedilen problemlerini ele alacağız. Tamamen farklı bir yaklaşımla, örnek ile sorgu problemlerinin benzerlik arama tekniklerinden esinlenen fikirlerle bir ASA sistemi kurmayı hedefledik. Bunun için, metin sorguları için yapay olarak "sahte örnekler" oluşturduktan sonra, bir alt-dizi dinamik zaman bükme araması kullanıyoruz. Ayrıca, dinamik zaman bükmede kullanılmak üzere, bu sorgu gösterimleri ile uygun bir mesafe metriğini bütünleşik ularak öğrenilmesini inceliyoruz. Önerdiğimiz bu yeni ASA sisteminin, dağarcık dışı terimlerin bulunmasında, mevcut en iyi ASA tekniklerinden daha iyi performans gösterdiğini, ve farklı yapısı nedeniyle geleneksel OKT tabanlı ASA sistemleri ile birlstirildiğinde ciddi iyileştirmeler sağladığını gördük.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF SYMBOLS

$\mathcal{A}$	Accumulated distance matrix
$\mathcal{B}$	Optimal beginning frame matrix
b	Bias value parameter for the DML network
C_{nxe}	Normalized cross-entropy cost
C_{nxe}^{min}	Minimum normalized cross-entropy cost
$\frac{dJ}{d\theta}$	Gradient of the parameter θ with respect to the cost J
$D(\mathcal{Q}, \mathcal{X})$	DTW distance between sequences $\mathcal{Q}$ and $\mathcal{X}$
$J(\theta)$	Cost function for the parameter θ
$d(\mathbf{q}_i, \mathbf{x}_j)$	Distance between frames $\mathbf{q}_i$ and $\mathbf{x}_j$
$\mathcal{L}$	Matrix that stores the path lengths
lr_t	Likelihood ratio for the trial t
llr_t	log-likelihood ratio for the trial t
$N_{target}(term)$	Number of correct occurrences of the $term$
$N_{HIT}(term, th)$	Number of correctly detected $term$ occurrences for the given th value
$N_{FA}(term, th)$	Number of false alarms for the $term$ for the given th value
$P(.\|.)$	Conditional probability
$p_{HIT}(term, th)$	Probability of the detecting the $term$ for the the given th value
$p_{MISS}(term, th)$	Probability of missing the $term$ for the given th value
$p_{FA}(term, th)$	Probability of false alarms for the $term$ given the th value
P_{target}	Prior probability of a term being in the dataset
r_t	Friendship label between a pair of inputs in DML
$\mathcal{Q}$	Query posteriorgram
$\mathbf{q}_i$	i-th frame of query posteriorgram
$sys(i)$	Output of the i^{th} KWS system
$\mathbf{SYS}\{i\}$	Vectorized output of the i^{th} KWS system
t_{beg}	Beginning time of a detection
t_{end}	Ending time of a detection

th	Score threshold value for detection
$\mathbf{W}$	Shared weight matrix parameter for the DML network
$\mathcal{X}$	Document posteriorgram
$\mathbf{x}_j$	j-th frame of document posteriorgram
Φ	Alignment function between query and document
β	Detection-false alarm cost parameter for the TWV metric
$\sigma(z)$	Sigmoid nonlinearity function for the input z
$\triangle\theta$	Update step for the parameter θ in training
η	Step size in training

LIST OF ACRONYMS/ABBREVIATIONS

AKWS	Acoustic Keyword Spotting
AM	Acoustic Model
ATWV	Actual Term Weighted Value
BUSIM	Boğaziçi University Signal and Image Processing Lab
CE	Cross Entropy
CA	Cosine Distance with Average Query Modeling
CB	Cosine Distance with Binary Query Modeling
CN	Confusion-Networks
DET	Detection Error Trade-off
DNN	Deep Neural Network
DML	Distance Metric Learning
DTW	Dynamic Time Warping
FOM	Figure of Merit
GMM	Gaussian Mixture Model
G2P	Grapheme-to-Phoneme
HAC	Hierarchical Agglomerative Clustering
HE	Histogram Equalization
HMM	Hidden Markov Model
IARPA	ntelligence Advanced Research Projects Activity
IV	In-vocabulary
JDML	Joint Query Modeling and Distance Metric Learning
KWS	Keyword Search
KNN	K-Nearest Neighbor
KST	Keyword Specific Threshold
LA	Log-Cosine Distance with Average Query Modeling
LB	Log-Cosine Distance with Binary Query Modeling
LM	Language Model
LSH	Locality Sensitive Hashing

LSTM	Long Short Term Memory
LPCC	Linear Prediction Cepstral Coefficient
LVCSR	Large Vocabulary Continuous Speech Recognition
MAP	Mean Average Precision
MCE	Minimum Classification Error
MFCC	Mel Frequency Cepstral Coefficients
MPE	Minimum Phone Error
MSE	Mean Squared Error
MTWV	Maximum Term Weighted Value
NIST	National Institude of Standards and Technology
OOV	out-of-vocabulary
OTWV	Optimal Term Weighted Value
PLP	Perceptual Linear Coefficient
PSPL	Position Specific Posterior Probability Lattices
PPM	Poisson Process Model
QbE	Query By Example
ROC	Receiver Operating Curve
RNN	Recurrent Neural Network
SCR	Spoken Content Retrieval
SDR	Spoken Document Retrieval
sDTW	Subsequence Dynamic Time Warping
sMBR	State-level Minimum Bayes Risk
SUR	Spoken Utterance Retrieval
STD	Spoken Term Detection
STO	Sum-to-one
STWV	Supremum Term Weighted Value
TWV	Term Weighted Value
WER	Word Error Rate
WFST	Weighted Finite State Transducer

1. INTRODUCTION

The digitalization of data and the ever-increasing storage capacity has brought us closer to a life like the fictional world in Jorge Luis Borges' Library of Babel [1], —a library of infinite size, with books that contain every possible string of letters and, hence, somewhere a key to the most important information for you. Though, in the fiction, it was highly doubted that the librarians would ever find this needle in that haystack. The situation we face today is somewhat similar: When it comes to retrieval of data, and especially the spoken data, the conundrum appears to be the "Paradox of Abundance" defined in [2]: "Quantity undermines the quality of our engagement". Hence, it is vital now, in the Internet era; to efficiently and correctly retrieve the data of interest to make our lives easier as evidenced by the well developed text retrieval technology, which is undoubtedly one of the biggest life-changing innovations of the past century. On the other hand, speech retrieval technology is still nascent even though speech is definitely one of the most appealing forms of information content. The retrieval of speech is therefore not only attractive but also necessary for users to efficiently browse across the vast quantities of multimedia content.

This dissertation focuses on a speech retrieval task called keyword search (KWS), also known as spoken term detection (STD). In KWS, the user provides a query in text form, consisting of one or more words. This query is searched in an untranscribed audio archive and the locations in the audio where the query exits are returned to the user. The text counterpart of this task is a very common one in everyday life: text retrieval. Nowadays, people rarely spend a day without searching something in Google. Text browsing also comes in handy and used in text editors with very well known short-cut button configurations. Despite this wide usage of text retrieval acknowledged by almost every computer user, searching for speech still remains a young research. Searching for audio-visual content in multimedia archives, such as YouTube and Ted Talks, video lectures such as MIT OCW, Coursera, edX and Udemy only provides videos whose title or meta-data has high similarity scores to the text query. Furthermore, the actual locations where the terms of interest are uttered are not available upon such a search.

In KWS, on the other hand, the audio documents are returned to the user along with the time stamps where the query occurs. Such a system may also be desired in situations where the user is interested in finding locations of their queries in broadcast news reports, radio communications, meeting recordings and telephone conversation recordings of a call center.

When a reliable transcription can be obtained automatically or manually for the audio recordings, the KWS problem becomes just a text retrieval task, for which the techniques have been relatively mature and efficient. However, if there is not enough training data to build speech to text systems in the language of interest, then the automated transcriptions become far from reliable. Such languages for which the available training data is scarce are called "*low resource languages*". According to [3], there are about 7000 human languages spoken in the world, and only $50 - 100$ of them have enough linguistic resources available to develop speech to text systems [4]. Furthermore, if we consider that it takes 4 hours to transcribe an hour long audio manually [5] with the average estimated costs of \$200/hour [6] and \$2000/hour [4], for word and phone level transcriptions respectively, the automatic processing of speech for low resource languages becomes even more important.

1.1. Statement of the Problem

In KWS, the query is in text form, whereas the search document is an untranscribed speech archive. Hence, the first developed and the most widely applied method to approach the problem is to use a automatic speech recognition system to convert the audio into text or statistical sequences of words in the language and then apply text retrieval on this output. However, when there is not enough linguistic resources to model reliable acoustic or language models,i.e. the target language is a low resource language, the speech recognition system will have high word error rates, which will then inevitably affect the KWS performance. KWS for low resource languages is a common problem in speech community and it has been addressed by OpenKWS [7], MediaEval [8], OpenSAT [9] and zero source speech challenge [10] campaigns, Johns Hopkins University's 2012 summer workshop on zero resource speech technologies [11], as well

as the Intelligence Advanced Research Projects Activity (IARPA) Babel Program [12] in recent years. Another instance of low resource speech processing is provided by University of Southern California Shoah Foundation in 2016 with audio-visual testimonies from survivors and other witnesses of the Holocaust in 63 countries and 39 languages [13]. Available linguistic resources for automatic processing of these languages are significantly lower than English or Mandarin. In this thesis, we will be addressing the problem of KWS on low resource languages using limited training resources as low as 10-hours of transcribed speech data provided by the IARPA Babel program.

Besides the high word error rates brought by the scarcity of the resources, one other conundrum we face with the low resource languages is the high out-of-vocabulary rate. The size and coverage of the vocabulary, i.e. the set of words that the speech recognizer "knows", is a critical aspect for speech recognition systems. Bazzi [14] and Szöke [15] analyze in their PhD Theses, the degradation in recognition accuracy when out-of-vocabulary words are encountered. Furthermore, for low resource languages, it was reported in [16] that practically half of the keywords in a keyword list in a KWS task include an out-of-vocabulary word, when the training set is as low as 10-hours. On top of that, if we consider that the users wish their keywords to be distinctive and discriminative, they will want to use content words instead of common words, the possibility of facing an out of vocabulary word in a real task would be higher. If a keyword entered by the user includes a word that is not within the vocabulary of the speech recognizer, it will not be found in the text transcription or the set of hypotheses, making it impossible to retrieve. A very basic idea to address this issue of out-of-vocabulary keyword retrieval, is to use sub-word units in the speech recognizer outputs. In this thesis, we will be addressing this problem proposing a novel search methodology that is not effected by the vocabulary size.

KWS can be considered as a detection problem. A KWS system returns an audio segment as a "relevant" hit, if it is "similar enough" to the query for some similarity measure. For the LVCSR-based methods, this similarity measure is the likelihoods obtained from the soft indexes. These likelihood values are then compared to a global

threshold value to mark the hypotheses as relevant or irrelevant. If a high global threshold is used it is likely to miss some hits, whereas a choice of a very low threshold may result in many false alarms. This global (keyword independent) thresholding calls for an efficient normalization of the scores, since similarity scores highly depend on keyword specific features, such as its length, frequency-in-language and phonetic structure of the keyword etc. Many of the normalization techniques in literature are based on the hypothesis scores being joint likelihoods of the query sequence. On the other hand, since our novel KWS approach is using a template matching method, novel score normalization techniques are needed to fit the new methodology. Hence, with this thesis, we introduce a number of novel normalization techniques to the literature. The proposed score normalization techniques prove to be efficient on all of the evaluation metrics of the task.

One other aspect we investigate is the fusion of KWS systems. Fusion of multiple systems is not only pertinent to KWS, but it also has been applied for many verification and detection problems in the literature. The main idea in fusion is to utilize powerful parts of one system to make up for weaknesses of another. It is widely practiced in KWS for combination of heterogeneous systems to reduce the number of false alarms and increase the number of correct hits by re-arranging the confidence scores of the hypotheses. The contemporary fusion algorithms call for iterating over the sets of hypotheses to find overlapping segments, then combining their scores upon some kind of normalization. This is a non-trivial task and may be time-consuming for large hypothesis sets and multiple systems in fusion. In this thesis, we propose a novel system fusion methodology for combination of several systems in a considerably reduced time and high efficiency.

1.2. Main Contributions of the Thesis

In this thesis, we address the above mentioned problems of KWS in low resource languages. For this, we propose and implement a new KWS system with methodologies inspired by query by example (QbE) retrieval tasks. In QbE retrieval tasks, the query and the document are in the same form, i.e both speech or video etc. Hence, in this

thesis, instead of following the contemporary recipe of converting the document into text form, we convert the query into the same representation of the audio document and conduct the KWS in a frame level similarity search manner.

In the end, with this thesis, we have a novel working pipeline that conducts KWS along with the proper normalization and system combination schematics. The proposed system performs significantly better than the state of the art KWS methodologies on out-of-vocabulary terms. Furthermore, when combined with the baseline systems it is shown to improve the performance on all of the metrics and all of the data sets we experimented on. We defend that, for out of vocabulary keywords, the KWS system that this dissertation proposes can be used as a standalone system and is a significant contribution to the related literature, for the average improvement is almost always two-folds the state of the art.

The main contributions of the thesis together with the corresponding scientific publications are listed as follows:

(i) Text queries are modeled as pseudo posteriorgrams using repeated one hot vectors (corresponding to the phoneme indexes for the pronunciation of each vector), and these queries were searched in the audio posteriorgram following the query by example recipe. To our knowledge, this is the first attempt the utilize the successful techniques of query by example retrieval techniques in KWS, and it was shown that such a technique helps improve the retrieval performance of contemporary speech recognizer-based systems in fusion. This work was published in [17] and [18].

(ii) Decoder phonemic confusions are included in artificial query modeling by using average posterior vectors for each phoneme in the pronunciation of the keyword, and different distance metrics were tested with different query models on two low resource languages. Furthermore, a novel "relative pruning" methodology was implemented on normalization of scores. This work was published in [19] and received the "Best student presentation award" by the organization committee [20].

(iii) After observing that varying the distance metrics on the proposed dynamic search methodology effects the performance, a novel distance metric learning model was proposed using frame level similarities. A Siamese neural network model was specifically tailored to output a distance measure between two input frames to incorporate the phoneme confusions into the distance metric and, at the same time, bounding the distance values within the desired range. It was shown that this new distance measure, using the query models of our previous work, yields better KWS performance than other known distance metrics. This work was published in [21].

(iv) We have extended the new distance metric learning neural network model in order to include learning of the query model representations hand-in-hand with the new distance metric. For this, we modified the Siamese neural network model into an asymmetric form, such that the one path has an initial layer to learn query model representations for each phoneme and the top (symmetric) layer learns the proper distance metric for these representations. We have shown that, such a joint learning of query model representations and the proper distance metric yields significantly better results than using other query modeling techniques and distance metrics. Furthermore, we have shown that this new methodology, as a standalone system, gives better out-of-vocabulary-keyword retrieval performance than the state-of-the-art baseline systems in the literature. This work was published in [22] and in extended form [23].

(v) Since the proposed search techniques offer a completely different range and interpretation of system scores, the current score normalization techniques of KWS literature diminish the efficiency of our technique. Hence, we proposed a set of novel score normalization techniques, not only effective for our model, but also potentially helpful for other query by example retrieval tasks. We have shown that the novel normalization techniques work better than the known ones, on all of the experiments and datasets. This work was published in [24].

(vi) The main power of the heterogeneity of the proposed system is effective upon combination with the other systems. In other words, the complementary nature of a novel system acts better when it makes up for the impotencies of current

systems, while utilizing powerful parts of them to improve its performance to build a single, better-than-all fusion system. However, fusion of different systems call for a re-scoring of the individual system scores to have the same relative meaning across systems. Also, a loop over all hypotheses for all systems to check for an overlap is necessary. To address this issue with the new proposed system, we implemented a time efficient combination procedure to work in significantly reduced time, which is not hurt by increasing the number of systems in combination and/or the number of keywords, at the cost of memory consumption. This work was submitted for publication as a Book Chapter in [25].

1.3. Organization of the Thesis

The thesis is composed of two parts. In Part-I, we provide a thorough literature survey on the subject. In Chapter 2, we introduce

(i) the definitions and details of the task in hand
(ii) the historical background and state-of-the-art methodologies
(iii) the evaluation metrics
(iv) the current approaches to the problems we address

In Part-II, we present the contributions of this thesis. In Chapter 3 we introduce the novel template matching based approach to KWS, with ideas inspired by query by example search techniques. In Chapter 4, we introduce our methodology to learn a distance metric and a query model to be used in the proposed search method. In Chapter 5, we present our novel score normalization and system fusion techniques for the proposed approach. In Chapter 6, we provide the experimental results on various data sets, along with comparison of baseline KWS systems, as individual systems and in fusion with them. In Chapter 7, we introduce an extension of the proposed work enhanced with the inclusion of recurrent neural networks and finally in Chapter 8 we draw our conclusions on this dissertation work.

2. LITERATURE SURVEY

Search and retrieval of speech documents has recently gained the interest of speech processing community. Detection of locations of certain keywords become important for scenarios where the primary information content of the multimedia documents lie within its audio content. Telephone conversations, broadcast or television programs, lectures, meeting recordings can be considered as examples of such cases. In this chapter we begin by introducing spoken content retrieval tasks to build up to where the task of interest, keyword search, stands. Then, we provide a thorough literature survey of keyword search by introducing techniques adopted in years of study along with recent developments on the task. This survey is followed by an introduction to the keyword search for low resource languages, where the main interest of keyword search community is focused on in the recent years. We conclude this chapter by providing a survey of the QbE-STD task, since we extend the techniques of QbE-STD to keyword search in this thesis, in order to address the difficulties accompanied by scarcity of resources in low resource languages.

2.1. Spoken Content Retrieval Tasks

The primary goal of the applications of spoken content retrieval (SCR) is to provide the user with the most relevant set of audio documents regarding to their query. More specifically, SCR can be defined as the task of retrieving the particular segments of some large speech archive, that is relevant to a query provided by a user.

The text counterpart of this task, text retrieval, is very familiar to everyone who use a computer or a smart phone, has become a major part of our daily lives and has relatively more mature products. Today, we all use search engines such as Google, Bing, Yahoo multiple times everyday. Furthermore, search of terms of interest in text is also a very common application with the short-cut keys like Ctrl+F. In SCR, on the other hand, the user provides a text query, composed of one or more words, and this query is searched in an untranscribed speech archive. The results are then returned to the

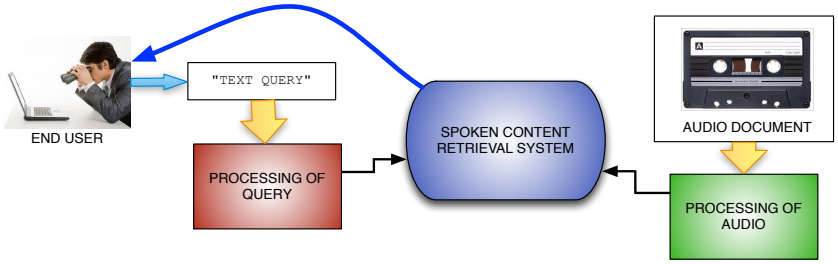

Figure 2.1. A general flowchart for spoken content retrieval.

end-user in an ordered fashion with respect to their relevance scores. A general SCR system can be visualized as in Figure 2.1. In this section, we will introduce various versions of spoken content retrieval tasks that were studied in the literature. Some relevant terms such as document, utterance, and keyword, to be defined here, will be useful for understanding the main problem addressed in this thesis.

The main spoken content retrieval tasks can be listed as follows:

- Spoken document retrieval
- Spoken utterance retrieval
- Keyword spotting (also called word spotting)
- Keyword search (also called spoken term detection)

The main difference between these tasks lay within the information returned to the user about their query.

2.1.1. Spoken Document Retrieval (SDR)

In spoken document retrieval, the user is interested in retrieving long recordings of audio that the query takes place. These recordings are called “documents”. Some analogous applications in text retrieval are Google and YouTube searches. The information retrieval system returns the user a list of web pages or videos ordered by their corresponding relevance scores. In SCR terms, the web pages and the videos can be defined as “documents”. On YouTube, the text query is searched within the meta data provided by the uploader. The scenario where the query is searched in what is being

spoken on the video would be an exact example of SDR. Recently Google's Audio Indexing (Gaudi) technology has started providing such a service on many YouTube channels. The song retrieval systems like Shazam and SoundHound can also be considered analogous to SDR, although the queries are samples (or examples) of the document in these applications.

2.1.2. Spoken Utterance Retrieval (SUR)

When the retrieval of long documents pertinent to the query is of little help and specific portions of the document are needed, the long document is generally segmented into shorter segments, called "utterances", which are roughly equivalent to sentences. The SUR system then returns the relevant smaller portions to the user. The task, where the user is interested in retrieving sentence-length segments, instead of entire audio documents, is called spoken utterance retrieval. Many successful systems such as AT&T ScanMail [26], BBN Rough'n'Ready [27], Carnegie Mellon University's Informedia [28] and SpeechBot [29] can be given as examples of applications of SUR systems.

2.1.3. Keyword Spotting

Keyword spotting, also known as word spotting, aims to find the exact locations of a text query within a speech document or a speech stream [30]. The query or the set of queries are known by the system developer prior to the search time, hence the keyword spotting systems are developed/optimized to detect these terms. The query term, composed of one or more words are called "keyword". Many applications of keyword spotting have been studied using likelihood maximization [31] and recurrent neural networks [32]; and many commercial products have been produced such SRI's DECIPHER [33], "Okay Google" [34], "Hey Siri" and Alexa.

2.1.4. Keyword Search (KWS)

KWS is the main focus of this thesis. Just like word spotting, keyword search, also known as spoken term detection, aims to find in speech all occurrences of a keyword along with the starting and ending time stamps. The main difference from keyword spotting is that the system developer does not have the knowledge of the keywords and the user can provide any keyword in run-time. KWS is a more realistic scenario than keyword spotting since the advance knowledge of the keywords is rarely the case. The KWS system development has to be implemented in such a way that the system operates on any keyword, of any length, then successfully finds and sorts them in the same relative relevance order. The difficulties presented by the liberty of choice of any run-time keyword will be discussed in the following sections and addressed in this thesis.

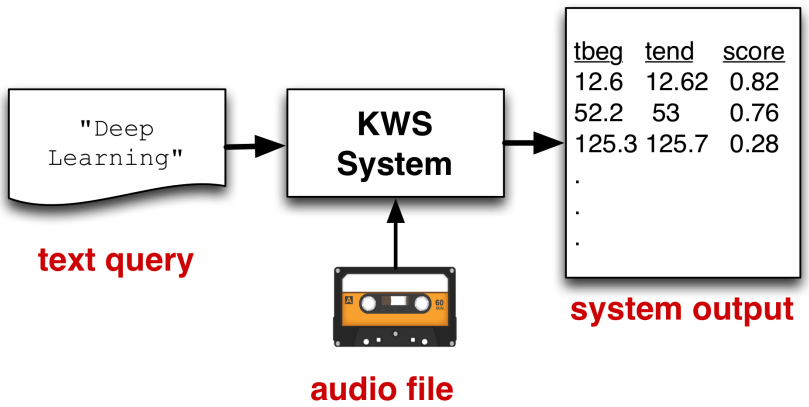

Figure 2.2. A simple flowchart and a sample output of KWS systems.

2.2. Previous Work and Methodologies for KWS

Since the query is given in text form, the most convenient and the first practiced approach to KWS is using a large vocabulary continuous speech recognition (LVCSR) system and executing text retrieval on the LVCSR output. Since LVCSR systems still play a major role in KWS research, we begin by briefly introducing the main components of LVCSR.

2.2.1. Main Components of LVCSR

Speech recognition can be defined as the task of automatic translation of spoken language into text. This translation, called transcription, is conducted by finding the most probable sequence of word sequences, $\hat{\mathbf{W}}$, among all possible word sequences $\mathbf{W}$ given the acoustic waveform sequence, $\mathbf{A}$.

$$\hat{\mathbf{W}} = \arg\max_{\mathbf{W}} P(\mathbf{W}|\mathbf{A}) \tag{2.1}$$

Since it is nearly impossible to obtain a model for $P(\mathbf{W}|\mathbf{A})$ for all possible word sequences, the search is conducted by maximizing the likelihood and the prior obtained by the Bayes rule:

$$\hat{\mathbf{W}} = \arg\max_{\mathbf{W}} P(\mathbf{A}|\mathbf{W})P(\mathbf{W}) \tag{2.2}$$

In (2.2) the term $P(\mathbf{A}|\mathbf{W})$ is called the "acoustic model" and $P(\mathbf{W})$ is called the "language model". Acoustic model presents a multivariate probability distribution for each possible word sequence in the language. In LVCSR, this value is generally obtained by calculating the joint probabilities of acoustic models (AM) of smaller sub-word level elements, such as phonemes. The AM of each sub-word unit is generally modeled with Gaussian mixture models (GMM) or directly from the output layers of deep neural networks (DNN). Durations of the phonemes are modeled by hidden Markov models (HMM). For this reason, the LVCSR systems, may sometimes be referred to as HMM/GMM-based or HMM/DNN-based systems.

The language model (LM), on the other hand, presents an n-gram conditional probability distribution for every word in the language, given the previous n words. Here, the set of words in the language, available to the LVCSR system on the training corpus is called the "vocabulary" of the LVCSR system. Therefore, if the speech involves words that are not in the vocabulary of the LVCSR system, they will not appear on the transcription, and will be replaced by the most probable hypothesis.

Such words are referred to as the "out of vocabulary (OOV)" words and it will be shown that this thesis also provides a novel and well-performing method for retrieving them.

2.2.2. Early KWS Work: Cascading LVCSR and text retrieval

The earliest mature works date back to mid 1990s when the LVCSR and text retrieval technologies began to achieve satisfactory performances. In [35], the outputs of the phone-based recognizers were used for finding the keywords. The discrimination was mainly dependent on the LMs used by the system. In the PhD Thesis [36] considerable contribution LVCSR system outputs and combining them with the information retrieval systems for SDR. The Text Retrieval Conference (TREC) evaluations conducted in 1999 and 2000 [37] are considered a milestone in the SDR applications. The application in [29] was the first SCR system used in web. Since these applications mainly focused on using Broadcast News speech, the LVCSR results had fewer errors due to better recording conditions when compared to conversational speech. Furthermore, in the first applications the OOV rate was as low as 1%. In general, the Word Error Rates (WER) for the LVCSR outputs to conversational speech is considerably worse than those used in TREC evaluations and other such early applications. Since LVCSR systems may fail to return reliable transcriptions under low resource and adverse recording conditions, the performance of the KWS systems that use them will be adversely effected. In order to alleviate this problem, the indexation of the LVCSR output was modeled as stochastic structures instead of 1-best (most likely) results.

2.2.3. KWS on unreliable LVCSR outputs

The main work on KWS, with more realistic low WER scenarios on conversational speech was initiated with the 2006 NIST Spoken Term Detection evaluations [38]. Since then, the main approach adopted for KWS has been first obtaining lattices from speech through LVCSR systems, and subsequently running the search using weighted finite state transducers (WFST) [39–41]. Many successful applications of KWS were implemented on broadcast news [42], video lectures [43], and web-videos [44] with this

approach. Most modern KWS systems make use of these structures, which will be introduced in the following sections.

2.2.4. Lattice-Based Methods

In order to alleviate the problem caused by high WERs, the retrieval is conducted on stochastic structures like lattices or confusion networks obtained from the LVCSR systems instead of the single-best transcriptions. The detection scores for the query term are then obtained from the lattices and returned to the user in an ordered manner. Lattices are compact representations of a large number of alternative sequences with their weights to represent the probabilistic structure of the data. Lattices are used for indexing the target document for fast retrieval purposes. Basically, indexing is the process of extracting from a document an expression of its content for use in retrieval. In other words an index is a compact structure which stores all the relevant information about a collection of documents. As per the indexing of lattices, models such as weighted automata, Confusion-Networks (CN) and Position Specific Posterior Probability Lattices (PSPL) are used to represent the probabilistic structure. Lattices were first introduced by [45] for a vocabulary-independent keyword search task. Later in [46], the queries were modeled as phone lattices and the word-spotting algorithm was used to achieve the SDR goal using the LVCSR output of the spoken document.

2.2.4.1. Lattices as Weighted Finite State Transducers. When searching or indexing probabilistic data like the LVCSR output where there is considerable amount of uncertainty dependent on the performance, it is required to deal with a large number of ranked or weighted alternatives. Thus, when indexing such a structure, work in [39] introduced a method for treating these lattices as WFSTs. The main idea of that work was to devise an algorithm to represent each utterance as a WFST and each query as an automaton then retrieving the queries with mapping each factor of the query automaton to the set of indexes of utterances when combined with the utterance WFST. Although this transducer model which aims to map terms to utterances is very suitable to SUR or SDR tasks, with the knowledge of position information existing in the

lattice structure, this procedure is also used in STD tasks. In [41], for the OOV terms whose pronunciations are not in the pronunciation dictionary, multiple pronunciations produced by a text-to-speech (TTS) front end was used as the query automaton. Applications of this inverted index scheme were further studied in [47], [48], [49] and [40]. For concrete automata theory and the set of algorithms behind this method, interested readers are encouraged to refer to very useful sources like [39] and [40]. A sample lattice structure was illustrated in Figure 2.3, and the WFST indexing applied structure of this LVCSR lattice is shown in Figure 2.4.

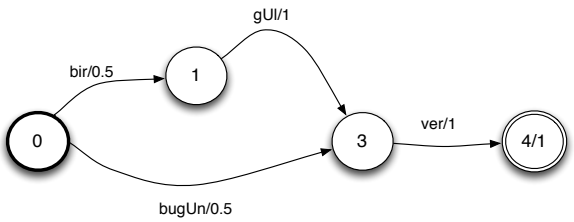

Figure 2.3. An example lattice structure.

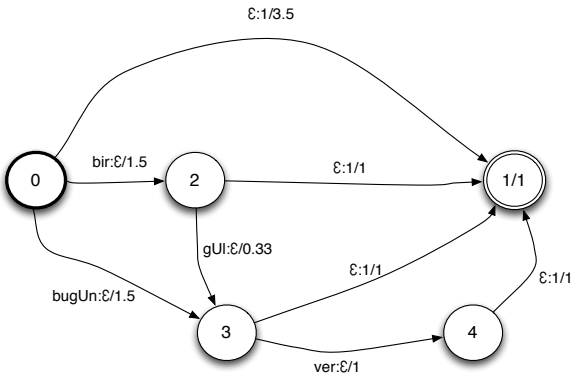

Figure 2.4. WFST indexing of the lattice in Figure 2.3.

2.2.4.2. Confusion Networks and PSPLs. LVCSR lattices posses a great deal of connectivity information which can be considered as redundancy for the SCR applications. Hence it makes sense to use simpler approximations to lattices, like confusion networks and PSPLs. The mentioned simplification may cause a lost in detection time accuracy when compared to raw lattices but due to their strictly linear and simpler structure, these models are perfect for the use of text indexers.

CNs, also called sausages, are concatenated series of hypothesis sets each of which are approximately time aligned words or sub-words. This simple structure can be

derived from lattices by clustering arcs (words) with similar timing to form the set of hypotheses, known as "confusion set". Typically lattice arks carry likelihood scores, in contrast, each ark in a confusion set carries posterior probabilities. Due to this structural scheme, confusion networks may have extra paths that do not exist in the original lattices and lack some of the paths that exist in the original lattices. This fact may result in higher recall rates along with the curse of high false alarm rates in SCR applications.

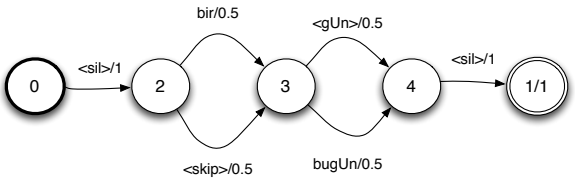

Figure 2.5. CN structure of the lattice in Figure 2.3.

Similar to CNs, PSPLs are designed to provide the 3-D retrieval information for each factor (or query word) : 1.document id, 2. position, 3.posterior probability. First two elements are two integers and the third one gives it a probabilistic characteristic. Hence the term "soft indexing" is used in literature for this procedure [50]. For soft-indexing of speech content CN and PSPL are used by reporting the posterior values on the lattice links as lists, based on the temporal durations of the pertinent link. This structure can also be used in SDR tasks, for which word counts are of importance. The word counts can be estimated using the posterior probability distribution $P(\mathbf{W}|\mathbf{A})$ that would directly appear on PSPLs or CNs. CN approximation of the lattice in Figure 2.3 was illustrated in Figure 2.5.

In summary, the LVCSR-based KWS systems have a two stage methodology. In the first stage, the audio document is indexed into statistical models like lattices, sausages or confusion networks. On the second stage, the entered query is searched within this index without further processing of the audio. This mechanism makes the retrieval a fast and efficient procedure, which makes it desirable for in vocabulary terms. The LVCSR-based retrieval scheme is visualized in Figure 2.6.

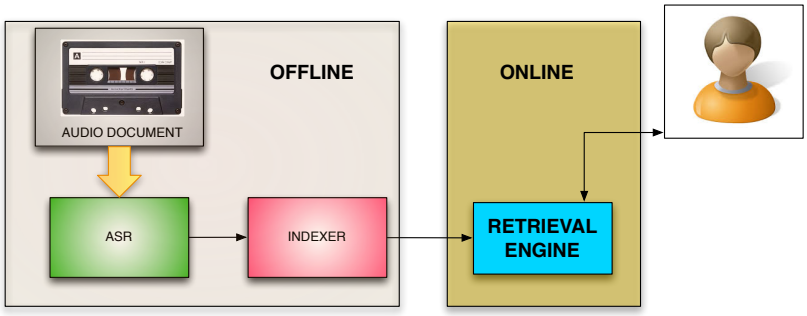

Figure 2.6. LVCSR-based KWS system flowchart.

2.3. Retrieval of OOV Terms

The traditional approach of linking an LVCSR system with a text retrieval system using word-based retrieval (and recognition) techniques faces the problem of having to know a-priori the vocabulary to search for. Although it is desired from a very large recognition vocabulary to cover the diverse message collections, at this era, where with the burst of social media information exchange, it has become practically impossible to cover the word variations. Query terms that involve words outside of the vocabulary of the LVCSR system, are called OOV terms. One major obstacle that the LVCSR-based KWS systems face, especially in low-resource or morphologically rich languages, is the OOV problem. Since the LVCSR systems utilize the lexicons and the language models obtained from the limited transcribed data to generate the lattices, any keyword containing an OOV term will not exist in the lattices. Common words are considered too general for a retrieval system hence users tend to go after discriminative therefore low-frequency words. As a result, it is more likely for a typical query term, in practice, to include either an OOV word or a word for which the language or acoustic model has not been trained well due to the lack of resources.

The most widely applied techniques to address the OOV problem can be listed as follows:

- sub-word based retrieval
- web-crawling for OOV terms
- retrieval using proxy-keywords

- keyword modeling as point process models

2.3.1. Sub-word Based Retrieval

In order to make the recognition and retrieval process independent of any word vocabulary, an obvious choice of action is to represent the query and document using sub-word units and then combine lattices such units. In other words, the method is to conduct the search on sub-word-based lattices, such as phonemes, syllables or morphemes. Such systems can be considered "open vocabulary", even though the LVCSR system has a fixed and limited vocabulary [51–54]. For retrieval, keywords are represented as sequences of sub-word units and matched against the lattice. While sub-word-based systems make it possible to retrieve OOV terms, they often suffer from high false alarm rates due to the absence of the lexical constraints. It should be noted here that using sub-word level units is a trade-off in the sense that the recognition performance will naturally decrease, but the retrieval performance might increase on average considering situations with high OOV rates. Hybrid systems, that address IV and OOV keywords with different level transducers have also been proposed [55]. They build word level lattices for in-vocabulary (IV) and sub-word level lattices for OOV, a computationally expensive procedure that carries the cost of running the decoding two times and increases the size of the document index drastically.

2.3.2. Web Crawling for OOV Terms

Another effort to address the OOV problem involves searching the web for text and using the obtained data to extend the LVCSR lexicon and the language model. Instead of solely using data provided with the low resource language set, additional text resources are automatically collected from the web [56, 57]. Since the low resource languages are relatively rare on the Internet and most of the web text is in formal written form, it is very challenging to find text data on-line to match the conversational training data. Besides, since it is likely to obtain spurious data by this automated collection, this technique also investigates methods to automatically filter the text data.

2.3.3. Retrieval Using Proxy Keywords

One other approach is to expand the LVCSR lexicon prior to the lattice generation using automatically generated pronunciations. In [16], the efficacy of lexicon expansion, given that an anticipation of the OOV terms are possible, is presented. However, in low-resource KWS, such knowledge is generally unavailable in advance. Hence, a cheap yet effective approach that involves using proxy keywords was proposed to utilize automatically generated pronunciations of the OOV keywords after the lattice has been generated. Since the OOV keywords do not exist in the lattice, the search is conducted on acoustically similar IV words instead of the OOV words [58]. For this, the phone confusion transducers are also integrated in the proxy keyword generation [59]. The proxy keyword approach not only provides a satisfactory performance in OOV handling, but it also alleviates the high false alarm rate and the computational expense issues associated with the sub-word or hybrid lattice-based KWS systems respectively. In [60] it was reported that, searching for word index using word proxies is more effective than searching for phone indexes. In this thesis, the proxy keyword approach is taken as the baseline OOV handling method.

2.3.4. Keyword Modeling as Point Process Models

As will be described in detail in Part-II, we conduct the search on frame-level acoustic similarities independent of the LVCSR system and the LM. A similar approach to ours obtained a comparable performance to the state of the art proxy keyword approach. In [61], the keywords were modeled using their phonetic indexes as point process models (PPM), and the search was conducted on the document posteriorgram without using WFST's.

2.4. Recent Developments on KWS

Recently, new directions to SCR has been studied some of which include modifying the LVCSR systems in order to improve the SCR performances. Throughout recently concluded IARPA Babel Program [12], various novel techniques were stud-

ied by the participants such as discriminative training for acoustic models [62], use of multilingual features [63] and data augmentation [64].

2.4.1. Retrieval Oriented Acoustic Modeling

A retrieval oriented acoustic modeling can be made by using a weighted version of the discriminative training. For the acoustic models of the LVCSR systems that are trained by DNNs, the discriminative criteria like minimum classification error (MCE), minimum phone error (MPE) [65], and the recent state-level Minimum Bayes risk (sMBR) can be used to achieve better recognition results [62] than that of cross entropy. A recent study suggest that adding weights to the pre-defined keywords favoring their correctness improves the precision of the retrieval system and also reduces the number of false alarms [66]. The objective function setting the acoustic model parameters can be then defined as

$$J(\theta) = \sum_{r=1}^{R} \sum_{s_r \in L(u_r)} A(w_r, s_r) P_\theta(s_r|u_r) \tag{2.3}$$

where u_r is the r-th training utterance, w_r the reference transcription of u_r, s_r an allowed word sequence in the lattice $L(u_r)$, $A(w_r, s_r)$ the accuracy estimated foru_r, $P_\theta(s_r|u_r)$ the posterior probability of the path given u_r. Here, by arranging the weight of $A(w_r, s_r)$ in favor of the expected keywords, the accuracy for these words can be improved.

Another approach can be adopted using the pseudo-relevance feedback applications. The retrieval results are given to the user, and the user marks relevant and irrelevant results. With the relevance counts for that query word, the DNN can be retrained using the counts, giving more confidence score to the relevant results [67].

2.4.2. Retrieval Oriented Language Modeling

Language models can be trained in such a way to improve retrieval performance for known or expected keywords. Simply repeating the sentences which include the query terms during the training improves the retrieval performance. Another approach is adopted in the neural network based language model whose inputs are history word sequences and the outputs are probability distribution over the words. By using weights for the likely queries in the objective function increases the KWS performance [68].

2.4.3. Retrieval Oriented Confusion Models

The confusion models are used to model the occurrence of the recognition errors, and these models can be optimized to achieve better retrieval performance. This application is very appropriate for the low-resource language that have a very small vocabulary and high OOV rate. The usage of the confusion models can be done in the following three ways:

- Query transformation: Using a confusion model, learned from the training corpus, transforming the different segments of each query into the sequences that the decoder generally mistakenly outputs for them. The new sequences are then, used to retrieve documents. This is particularly applicable to SDR scenarios
- Spoken Content transformation: Instead of the query, the recognition output of the document is transformed, same idea as the previous one.
- Fuzzy match : Learning a similarity rule between different word or sub-word sequences, so that the lattices containing word or sub-word sequences sufficiently close to the query can now be retrieved as relevant

In above mentioned techniques, the necessary confusion model is represented as a matrix, considering the number of sub-word units. In this naturally asymmetric matrix, the value of the element at i-th row and j-th column indicates the probability that the i-th sub-word unit may be mis-recognized as the j-th sub-word unit. It has been shown in [59] that using confusion models yielded significant improvement in the performance

of the KWS system on 4 different low resource languages.

2.4.4. Multilingual Features for Low Resource KWS

Many recent work in low resource KWS and LVCSR study using multilingual representations for acoustic models [63]. The methodology is generally based on a two-stage network, where on the first recurrent neural network stage the multilingual features are extracted. These features are then used as input to the second stage being a feed-forward DNN. This paper presents two novel methods for deriving ML representations. It has been reported that utilizing multilingual bottleneck features on a network trained on many low resource languages can be used to improve the KWS performance of a new low resource language upon fine tuning [63].

2.5. A Related Task: Query by Example Spoken Term Detection

In this section we include a very pertinent task to KWS called Query-by-Example Spoken Term Detection. The very basic definition of QbE-STD is searching for audio content within audio content using an audio query. In other words, QbE-STD can be defined as the task of retrieving the locations of a query term, provided by the user as an audio snippet, in an audio archive. In the text-based KWS tasks, all of the methodologies discussed were assuming an availability of enough resources in the target languages, depending on training models based on transcribed data, phone sets and pronunciation dictionaries. However, for the languages which has low or no resources, implementation of the previously discussed KWS systems is a challenging issue. For the low resource languages, it is not possible to construct reliable acoustic or language models with statistical tools. Therefore a need for language-independent modeling approaches emerge. The task of QbE-STD is developed as a solution to this problem. In contrast to the text based KWS, in QbE-STD applications, the user searches the spoken content using a part of the spoken document which has been known as relevant. The academic work on QbE-STD was expanded by the MediaEval [8] campaign which aimed to build a language independent system using an unsupervised scheme, have yielded promising results.

The approaches adopted for QbE-STD during the MediaEval campaign can be grouped into two main categories:

- Symbol based methods : using phone transcription of the speech signal gathered by acoustic models of similar languages and running a methodology similar to text retrieval
- Template matching based methods : using features from the document and the query and borrowing methodologies from Dynamic Time Warping (DTW) based speech recognition

2.5.1. Symbol based QbE-STD

In the first symbol based method, the query and the document is labeled into sequences of word or sub-word units, then techniques similar to lattice-based search methods are employed. Similar to the WFST based indexing scheme, the (spoken) query can also be expressed as a lattice, and the queries are retrieved via the combination of the document index transducer and the query lattice. This time, the pronunciation variability model of the query is replaced by the uncertainty of the recognizer performance for the spoken query. This method has two major drawbacks:

(i) It is assumed that the resources are available for the target language where it generally is not the case in QbE-STD tasks,
(ii) The recognition accuracy for the short queries may be inherently low.

One interesting symbol based search method, introduced by [69] is the method called acoustic keyword spotting (AKWS). The AKWS technique was initially proposed for in-language and text based spoken retrieval tasks [70]. Therefore, it is applicable when enough resources are of availability. This is how the procedure follows: First the utterances are filtered out and freed from the non-speech parts using a speech activity detection scheme. Then, the audio is converted into sub-word unit posteriors (generally 3-state phone posteriors). Logarithm of the posteriors are taken and fed into the keyword spotter. The keyword spotter is designed as decoders of two models:

the “keyword model” and the “background model”. The ratio of the likelihoods of the utterance to the two models are calculated and an alarm is raised if the likelihood ratio is below some threshold level. The so-called “keyword model” and the “background model” are two HMMs modeled as follows: The keyword model is nothing but the phone (or state) left-to-right-topology HMM derived from the pronunciation dictionary and the statistical acquisitions from the training corpora. The background model is a dummy phone (or state) loop that can take any sequence of phones (or states). The detection scheme based on these two models are depicted in Figure 2.7.

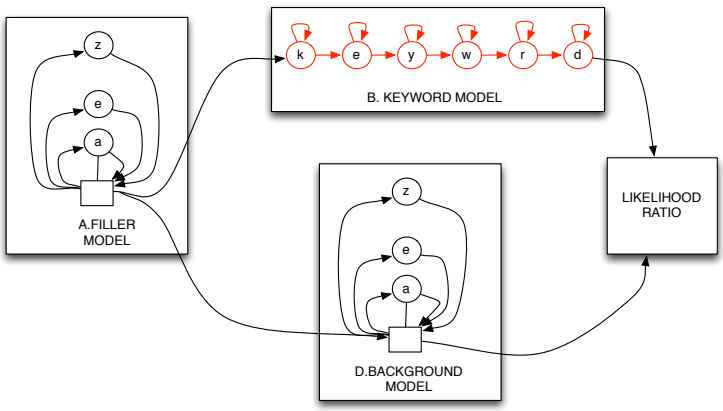

Figure 2.7. Illustration of keyword model and background model scheme for AKWS.

In this scheme, the main part of an AKWS system is a Viterbi-based decoder with the modification of calculating the likelihood ratio. The ratio of the likelihoods of two models “keyword” and “background” are calculated and the system outputs a detection, if the ratio is below some threshold. When the ratio approaches 1, it means that it is more likely that the underlying frame sequence has a keyword beneath it. Hence the log-likelihood ratio $log(\mathcal{L}(keyword)) - log(\mathcal{L}(background))$ is as low as 0. The likelihood ratio is divided by the length of the detection to compensate for the different lengths of the queries. One superiority of this scheme to the lattice based symbolic systems (to appear on low resource scenario) is that no language model is used. Furthermore, the vocabulary is used only for the set of keywords.

In the MediaEval 2014 task named QUESST [71], the work in [72] proposed applying this scheme for QbE-STD by using phone recognizers for the spoken query and achieved promising results. In order to introduce language independence and free

the system off the assumption of phone sequences, the work in [70] proposed modeling the keywords as a concatenation of states, the number of which is decided by the duration of the keyword and modeled these states as Gaussian components trained from the examples.

2.5.2. Template Based Search Method

Template-based search method have proved to be the most effective search method for QbE-STD tasks and it is preferred since it overcomes the limitations of LVCSR based systems and AKWS techniques [67, 73–80]. Template based search methods are inherently language independent, hence very appealing to the scenario of interest. The availability of working directly on acoustic level without the need for an LVCRS system is another advantage of this scheme. Template-based methods take their roots from the early speech recognition systems [81] and have previously been explored in a variety of audio applications such as music retrieval [82, 83]. These techniques typically follow a DTW-based search methodology, where the query being "the template" matched the searched section of the audio. Firstly, the query and the document are converted into sequences of feature vectors, and then a similarity matrix is constructed to search for the query within a spoken audio.

In the early DTW-based isolated word recognition systems, the acoustic parameters such as linear prediction cepstral coefficients (LPCC) or mel frequency cepstral coefficients (MFCC) are used as feature vectors [84]. However, these acoustic parameters are easily effected by the speaker and environmental mismatches. Therefore, speaker-independent feature representations such as Gaussian and phone posteriorgrams are preferred in the recent template-based search methods. The work in [85–87] used MFCCs for modeling query and document and employed a DTW-based search. Similarly the work in [88] used phone posteriorgrams derived from a multilayer perceptron as the feature vectors to fulfill speaker independence. Similarly, [73] used Gaussian posteriorgrams as feature vectors and employed the search through a version of the DTW algorithm called segmental-DTW.

This procedure borrows ideas from DTW based speech recognition since the direct alignment of two same words is not possible due to the fact that the durations of each sub-word units in general differ from utterance to utterance. The dynamic programming and warping algorithm (i.e. DTW) is properly offered as a solution to this problem. The traditional DTW algorithm tries to find an alignment and a distance for two sequences [83]. We present the basic ideology of DTW here, and in the following sections we will build up to the modifications of DTW algorithms used for KWS.

2.5.2.1. Mathematics of Dynamic Time Warping. Given two sequences (henceforth called utterances) $\mathcal{Q}$ and $\mathcal{X}$ to be aligned, we represent each as a time series of feature vectors, $(\mathbf{q}_1, \cdots, \mathbf{q}_M)$ and $(\mathbf{x}_1, \cdots, \mathbf{x}_N)$, respectively. We denote the optimal alignment path between $\mathcal{Q}$ and $\mathcal{X}$ as $\hat{\Phi}$. An alignment Φ between $\mathcal{Q}$ and $\mathcal{X}$ is a sequence of pairs;

$$\Phi = (i_k, j_k), \quad k = 1, \cdots, T \tag{2.4}$$

that represents the mapping

$(\mathbf{q}_{i_1} \Leftrightarrow \mathbf{x}_{j_1}), (\mathbf{q}_{i_2} \Leftrightarrow \mathbf{x}_{j_2}), \cdots, (\mathbf{q}_{i_T} \Leftrightarrow \mathbf{x}_{j_T})$ where T is the length of the *alignment path.*

In the traditional DTW, the two utterances are aligned in such a way that their starting and ending frames coincide, i.e. $i_1 = j_1 = 1$, $i_T = M$ and $j_T = N$. The optimal alignment path, $\hat{\Phi}$, is the one that has the minimum total distance, henceforth called as accumulated distance, defined as:

$$D_{\Phi}(\mathcal{Q}, \mathcal{X}) = \sum_{k=1}^{T} d(\mathbf{q}_{i_k}, \mathbf{x}_{j_k}) \tag{2.5}$$

where, $d(\mathbf{q}_{i_k}, \mathbf{x}_{j_k})$ is the distance between the two vectors $\mathbf{q}_{i_k}$ and $\mathbf{x}_{j_k}$. Many different distance measures can be used depending on the environment and application. When

employing the algorithm on utterances with MFCC features, euclidean distance measure can be used while cosine distance measure is preferable when posterior probability vectors are used in the comparison. Instead of calculating all possible alignments between $\mathcal{Q}$ and $\mathcal{X}$, the DTW algorithm dynamically progress the distance values and calculates the accumulated distance in a Viterbi-like manner shown in Algorithm 1. Due to the very nature of the speech signals, sequence evolution constraints are applied so that only the neighboring frames (see line 6) are used in the calculation.

Algorithm 1: DTW

1: **for** $i = 1$ to M, $j = 1$ to N **do**
2: **if** i=1 **or** j=1 **then**
3: $\mathcal{A}(i,1) = \sum_{k=1}^{i} d(\mathbf{q}_k, \mathbf{x}_1)$
4: $\mathcal{A}(1,j) = \sum_{k=1}^{j} d(\mathbf{q}_1, \mathbf{x}_k)$
5: **else**
6: $\Omega = \{(i, j-1), (i-1, j), (i-1, j-1)\}$
7: $(r,s) = \underset{\forall (p,q) \epsilon \Omega}{\operatorname{argmin}}(\mathcal{A}(p,q) + d(\mathbf{q}_i, \mathbf{x}_j))$
8: $\mathcal{A}(i,j) = \mathcal{A}(r,s) + d(\mathbf{q}_i, \mathbf{x}_j)$
9: **end if**
10: **end for**

Figure 2.8. Dynamic time warping algorithm

The accumulated distance matrix, $\mathcal{A}$, introduced on line 3 with size NxM, is used in dynamic programming to avoid the calculation of all possible alignment paths. The (i,j)-th element of $\mathcal{A}$ can be expressed as the total distance on the best alignment between $\mathcal{Q}$ and a $\mathcal{X}$ up to $(\mathbf{q}_i, \mathbf{x}_j)$. Hence the distance between $\mathcal{Q}$ and a $\mathcal{X}$ can be retrieved from $\mathcal{A}$ as:

$$d_{DTW}(\mathcal{Q}, \mathcal{X}) = D_{\hat{\Phi}}(\mathcal{Q}, \mathcal{X}) = \mathcal{A}(N, M) \tag{2.6}$$

It should be noted that in QbE-STD, one of the sequences (query) is considerably shorter than the other sequence (document). Furthermore, the query can exist anywhere inside the document. Therefore the constraint of DTW that the two sequences start and end at the same location does not work for direct application in QbE searches.

For this reason, the classical dynamic time warping algorithm have been modified to conduct search and, "segmental" and "subsequence" DTW versions have been introduced.

2.5.3. Segmental Dynamic Time Warping

The segmental DTW is a modification to the traditional DTW, which is suitable for finding alignments between different "segments" of the two utterances $\mathcal{Q}$ and $\mathcal{X}$. For the QbE-STD, the shorter utterance (i.e.query) is searched within smaller segments of the longer utterance (i.e the document). The first application of segmental DTW to the task of QbE-STD was with the work [73]. The segmental DTW employs the regional search by incorporating global constraints and restricting the allowable shapes that the alignment path can take. Likewise, it is also possible to allow generating multiple alignment paths, in many sub-sequences of the two utterances. It is intuitive to expect that for the two utterances to be aligned, one should not get too far ahead of frames from the other. Hence the following global constraint is put to the indices of the allowable search: for the k-th alignment in alignment path, $\mathcal{P}_k = (i_k, j_k)$ originating at (i_1, j_1),

$$|(i_k - i_1) - (j_k - j_1)| \leq R \tag{2.7}$$

which yields and allowable diagonal search region of width $2R + 1$. For the spoken document $\mathcal{X}$ with length N, the allowable segment starting coordinates will be:

$$(1, (2R+1)k + 1), \quad 1 \leq k \leq \left\lfloor \frac{N-1}{2R+1} \right\rfloor \tag{2.8}$$

These constraints can be added to step 2 of Algorithm 1 and can be seen in Figure 2.9. The segmental DTW algorithm iterates through all diagonal bands seen on the figure, finding one optimal warping path with minimum accumulated distance within each diagonal band. Each diagonal band serves as a candidate location of the spoken query, with allowed temporal distortion defined by the width of the band $(2R + 1)$.

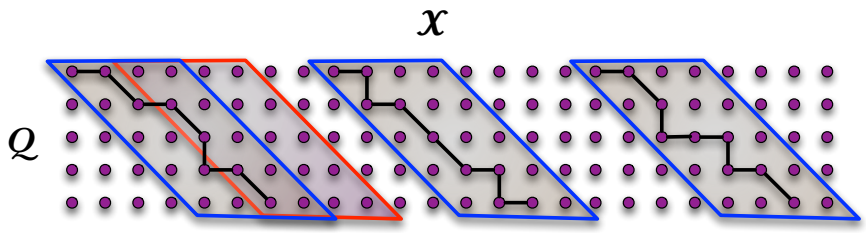

Figure 2.9. Illustration segmental DTW scheme.

2.5.4. Subsequence Dynamic Time Warping

The Subsequence Dynamic Time Warping (sDTW) algorithm is another version of the DTW algorithm, specifically suitable for the task of interest. While in the traditional DTW, the best alignment between $\mathcal{Q}$ and $\mathcal{X}$ are found by fixing and aligning the start and end frames of the both utterances, in sDTW, it is a fact that $\mathcal{Q}$ is considerably shorter than $\mathcal{X}$ and the start point of $\mathcal{Q}$ can be aligned with any point of $\mathcal{X}$. Furthermore the segmentation modification offered by segmental DTW fails to perform well when the durations of the two utterances differ by a larger margin. It has been reported by the study [89] that the audio recordings can be as long as twice the original term in the spoken documents like broadcast news since the speaker tend to slow down the speaking pace drastically in order to make sure the pronunciations are good. Hence, sDTW is used for finding the best alignment between $\mathcal{Q}$ and all sub-sequences of $\mathcal{X}$ with a solution to this problem of segmental DTW. The best alignment yields the most similar subsequence of $\mathcal{X}$ to $\mathcal{Q}$. The similarity search is done through the accumulated distance scores between $\mathcal{Q}$ and all sub-sequences of $\mathcal{X}$ averaged by the alignment path lengths $T_{\mathcal{P}}$.

The previously defined $\mathcal{A}(i,j)$ of the DTW algorithm now defines the total distance of the best alignment between $\mathcal{Q}$ and a sub-sequences of $\mathcal{X}$ up to $(\mathbf{q}_i, \mathbf{x}_j)$. In the accumulated distance calculation, the length of the path can be incorporated to prevent favoring small but unlikely sequences. Hence a new matrix $\mathcal{L}$ can now store the path-length information. The main difference of the sDTW algorithm is that it allows beginning from any frame of $\mathcal{X}$ while in DTW the start frames are always 1. The sDTW algorithm can be seen on Algorithm 2.10:

Algorithm 2: Subsequence DTW

for $i = 1$ to M, $j = 1$ to N **do**
  if i=1 **then**
    $\mathcal{A}(1,j) = d(\mathbf{q}_1, \mathbf{x}_j)$
    $\mathcal{L}(1,j) = 1$
  else if j=1 **then**
    $\mathcal{A}(i,1) = \sum_{k=1}^{i} d(\mathbf{q}_k, \mathbf{x}_1)$
    $\mathcal{L}(i,1) = i$
  else
    $\Omega = \{(i,j-1), (i-1,j), (i-1,j-1)\}$
    $(r,s) = \underset{\forall (p,q)\epsilon\Omega}{\operatorname{argmin}} \dfrac{\mathcal{A}(p,q) + d(\mathbf{q}_i, \mathbf{x}_j)}{\mathcal{L}(p,q)+1}$

    $\mathcal{A}(i,j) = \mathcal{A}(r,s) + d(\mathbf{q}_i, \mathbf{x}_j)$
    $\mathcal{L}(i,j) = \mathcal{L}(r,s) + 1$
  end if
end for

Figure 2.10. Subsequence DTW algorithm.

Here, the $\mathcal{L}$ matrix stores the length of the best alignment up to $(\mathbf{q}_i, \mathbf{x}_j)$. There may be multiple occurrences of the query in the document file. In order to search for all matches that are above the threshold, a recursive search algorithm is used as follows: After the best match is found with Algorithm 2 and if it is above the threshold, this subsequence is removed from the document and the remaining parts are searched for their best matches. This procedure is continued until the best matches in all document parts are below the threshold or the document parts are too short to be searched. This

recursive search is visualized in Figure 2.11.

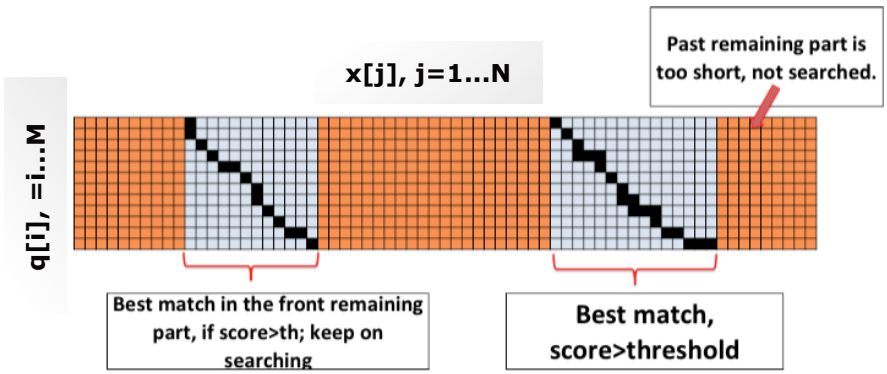

Figure 2.11. Illustration of recursive detection scheme for sDTW.

2.5.5. Multiresolution Dynamic Time Warping

Since the DTW is a computationally expensive procedure, many speed-up techniques have been proposed in the literature. Aside from using multiple GPUs to run the algorithm in parallel computation, multiple fold calculations to decrease the level of complexity wuthout hurting the performance are also possible. We call these techniques multi-resolution DTW. Some multi-resolution DTW methods studied in the literature are as follows:

2.5.5.1. Lower-bound estimation. This approach was initially proposed for DTW-KNN (K-nearest neighbor) for the systems whose goal is to retrieve the K best matches [90]. The lower-bound estimation technique for QbE-STD works on posteriorgram features. This application is better for segmental DTW where the query and the segment of the document have similar sizes. The basic idea is to segment the query and the document segment into windows and use the maximum posterior values within that window in the DTW calculation. This way, the calculation reduces in size drastically without losing any valuable information because it is proved that the windowed accumulated distance is always lower than the original accumulated distance [91].

2.5.5.2. Segmenting the utterances. One other very intuitive speed-up method is to segment the query and the utterance by clustering the acoustically similar frames

and run a coarse DTW search on these segments. Instead of averaging or summing the features within the segment, a bigger vector of segments can be produced and a distance measure working on such segments can be used. Such segmentation scheme is called hierarchical agglomerative clustering (HAC) and done by minimizing the total variance. This technique have been applied to QbE-STD and achieved a significant speed-up on the algorithm [76].

2.5.5.3. Unsupervised Offline Indexing of the Document. In the work [92], the off-line indexing of the document have been studied in an unsupervised fashion. The indexing scheme is named Locality Sensitive Hashing (LSH) and the procedure is as follows: Each signal frame $\mathbf{x} \in \mathbb{R}^d$ in both $\mathcal{Q}$ and $\mathcal{X}$ is mapped into low dimensional bit signatures $h(\mathbf{x}) \in \{0,1\}^b$, such that the hamming distance can be used as an approximation to the distance measure of interest. The mapping is done through a transformation matrix $\mathcal{T} \in \mathbb{R}^{d \times b}$ populated by random numbers from $\mathcal{N}(0,1)$. The k-th column of this matrix, $t_k \in \mathbb{R}_d$ can be used to find the kth bit of the signature $h(\mathbf{x})$ denoted $h(x_k)$ by zero thresholding the inner product $\mathbf{x} \cdot t_k$ and assigning 0 if less than zero, and 1 otherwise. The transformed document frames can be sorted in a alphabetical order and each frame of the transformed query can be searched using the logarithmic-time binary search. Then the approximated DTW is conducted only on the frames only on the most similar frames to the current query frame. The most widely used cosine similarity between two frames can be approximated by:

$$cos(q_i, x_j) \approx cos(\frac{H(h(q_i), h(x_j))}{b}\pi) \tag{2.9}$$

where $H(q,x)$ is the hamming distance and this approximation tends to equality as $b \to \infty$

2.5.5.4. Information Retrieval Based-DTW. Another improved DTW method is the one called Information Retrieval Based DTW (IR-DTW). Similar to the idea suggested in Sec. 2.5.5.3, IR-DTW makes a similarity search for each frame of the query and saves the relevant frames. Then the retrieval is done on the selected frames and a sub-set

of the whole document. From these frames, a vector of extendable path end locations were recorded and the speed and memory efficiency was achieved [93].

2.5.5.5. Pre-filtering for sDTW. In one of the works that were done in Bogazici University BUSIM Lab, automatically eliminating dissimilar parts of the document was studied [94]. For this, time-average posterior vectors were calculated from overlapping segments of the document to constitute lower resolution posteriorgrams. The cosine similarity measure was used to retrieve the most similar average vectors to the query time-average. Then a second fold sDTW was implemented only on the retrieved frames on the pre-filtering to account for the within word variability and to find the exact start and end frames. It was shown that, this procedure not only improves the speed, but also the recall rate.

2.5.6. Using Multiple Examples of the Query

It has been shown in [70] that using multiple examples of the query instead of one single example, improves the retrieval performance. This can be done by asking the user provide more than one example, using a database of examples in hand or using pseudo-relevance feedback using the best matches. The scheme is as follows: (i) Sort the examples according to a certain metric (ii) Run DTW in-between the best and the worst query by treating the best one as "query" and the worst one as "utterance". (iii) Update the phone state posteriors according to resulting alignment. If we call the best query example $\mathbf{q}$ and the worst query example as $\mathbf{r}$, the update process is as follows:

$$q_i^{new} = \frac{q_i + \sum_{\forall j:(i,j)\in\hat{\Phi}} r_j}{1 + N_j} \tag{2.10}$$

where N_j is the number of vectors in $\mathbf{r}$ belonging to q_i according to $\hat{\Phi}$. The sorting metric in step (i) can be according to the confidence to the query, i.e. according to test results to each of them, or phonetic recognition performance. A more realistic and quantitative measure of metric is acquired using the DTW distances, i.e, $D_{\hat{\Phi}}(\mathcal{Q}_i, \mathcal{Q}_j)$. Assuming we are using k query examples or query models, we will get a symmetric

$k \times k$ DTW matrix whose (i, j)-th element is $D_{\hat{\Phi}}(\mathcal{Q}_i, \mathcal{Q}_j)$. The *confidence* on $\mathbf{q}_i$ can be used as the sum of the i-th row of the DTW matrix. When using multiple examples, say 5, we get the query combination of the 4-th and 5-th queries first, denoted $\mathbf{q}_{45}$, then the combination of the 2-nd and 3-rd, denoted $\mathbf{q}_{23}$, then combination of these, $\mathbf{q}_{2345}$ and combine it with $\mathbf{q}_1$ to get $\mathbf{q}_{12345}$.

To summarize, since the task of QbE-STD is multi-lingual and there is no LM, the lattice-based approach of the state of the art KWS is not an option for QbE- STD. Hence, the symbol-based approaches utilize the HMM-based keyword vs. background model likelihood maximization technique. Template matching-based approaches, on the other hand, are centered around running versions of the DTW algorithm for an acoustic similarity search using the query as a template. Although some works used spectral features like MFCC's to represent query and document frames, Gaussian or phone posteriorgrams proved to be more appealing to many other groups due to their speaker independence. There are two important conclusions of the MediaEval campaign, which lead the QbE-STD challenge: (i) The template matching-based systems yield better results than the symbol-based systems and (ii) the fusion of heterogeneous systems (symbol+template) improves performance significantly.

Therefore, in this thesis work, we make use of the successful applications of template-based methods of QbE-STD and extend them to KWS. Furthermore, we investigate the fusion of such a new KWS system with the contemporary LVCSR based KWS systems to improve the retrieval performance. Practically, it is expected for the LVCSR-based systems to work better for IV terms, whereas the main contribution from the novel technique we propose (using the template-based search) is expected to be on OOV terms.

2.6. Evaluation Metrics

The evaluation methodology is a very important aspect for system development. For almost all of artificial intelligence systems, the most obvious choice of evaluation is using human judges for telling the degree of success of the system. In this application,

the success can be defined as degree of relevance for SDR and SUR systems; and detection perfection (accuracy and location) for KWS and QbE-STD systems. As stated in the previous section, since text retrieval systems possess a considerable degree of maturity, one option in SCR building and evaluation would be to try to close the gap between text and speech search technology. As a reference, the text retrieval system output for a manually transcribed audio data can be used, then the SCR system would be built with the goal of achieving this performance. In the following sections, we will be providing introductions of the commonly used evaluation metrics, along with their interpretations.

2.6.1. Precision and Recall

For SDR or SUR tasks, two very pertinent evaluation metrics are Precision-Recall Rate and the F-Measure. Precision (Pr) is defined as 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. The calculation of precision of recall is conducted counting the retrieval and ground truth numbers:

$$Pr = \frac{1}{Q}\sum_{q=1}^{Q}\frac{C(q)}{A(q)} \qquad Rc = \frac{1}{Q}\sum_{q=1}^{Q}\frac{C(q)}{R(q)} \qquad \text{and} \qquad F = \frac{2 \text{ x } Pr \text{ x } Rc}{Pr + Rc} \tag{2.11}$$

where,

- q :a query term in the keyword list
- Q: number of queries
- $R(q)$: number of documents that are relevant to query
- $A(q)$: number of retrieved documents
- $C(q)$: number of relevant documents correctly retrieved

Similar metrics derived from precision and recall rates are Mean Average Precision (MAP), R-Precision, Precision@N metrics. For the computation of MAP, Pr values are calculated at each of the Rc values (it should be noted that Rc varies between 0 to 1) and the average of the areas under this $Pr - Rc$ curve for all queries are calculated. Precision@N is the Pr value of the top N returned documents (where $N = 10$ is a common choice). R-precision is similar to Precision@N, only that N varies for each given query q, and is set to the number of relevant documents $R(q)$.

2.6.2. Detection Scheme for KWS

For SUR and SDR, we count the number of relevant documents returned. On the other hand, for KWS, the locations of the hypotheses also matter. In order to describe the performance metrics for KWS, the methodology for detection of queries and counting the correct/incorrect detections should be introduced. The outputs of the system are ordered with respect to some confidence score. If the score of an output is above some given threshold, then the output is marked "YES", and it is marked "NO" if it is below the threshold. In other words, a binary decision making is required per the task definition. The evaluation system looks for the overlaps between the reference occurrences and the outputs of the system, then marks the outputs as "HIT" if the central point of the outputs labeled as "YES" overlaps with the 0.5 sec extension of the reference and "FA" (referring to False Alarm) otherwise. The missed detections and the detections labeled as "NO" which overlaps with the reference occurrence are marked "MISS". In summary three labels for system performance evaluation are:

- HIT : If the decision is YES and overlaps with the reference.
- MISS :If the there is no system decision for the reference, or the system decision for the reference is NO.
- FA : If the system decision is YES while there is no reference for this time span.

This scoring scheme was introduced in NIST STD 2006 evaluations [95] and visualized in Figure 2.12.

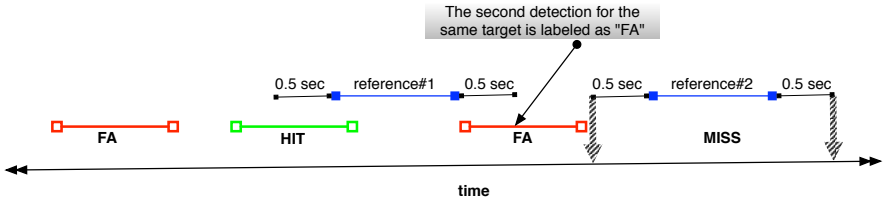

Figure 2.12. Detection and labeling scheme for KWS.

Given the detection scheme for KWS tasks, the evaluation system counts the "MISS", "HIT" and "FA" labeled outputs for the calculation of the statistics to be used in the above mentioned evaluation metrics. Let us denote:

- $term$ the term being searched,
- th the decision threshold,
- T the total duration of the document,
- $N_{target}(term)$ the number of all correct occurrences of the $term$ in the dataset,
- $N_{HIT}(term, th)$ the number of correctly detected $term$ occurrences for the given th value,
- $N_{FA}(term, th)$ the number of false alarms for the $term$ for the given th value,

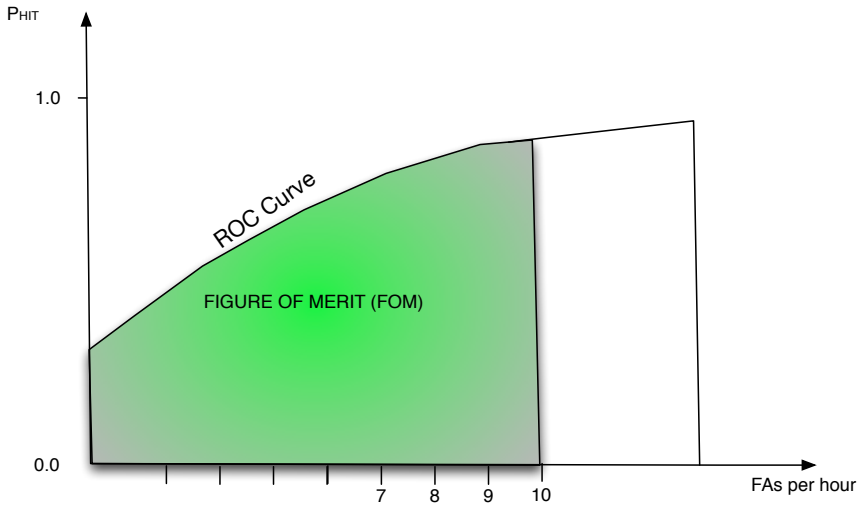

Figure 2.13. Illustration of ROC Curves and FOM.

With the definitions of the above counts, the following statistics are calculated:

$$p_{HIT}(term, th) = \frac{N_{HIT}(term, th)}{N_{target}(term)}$$

$$p_{MISS}(term, th) = 1 - p_{HIT}(term, th) = 1 - \frac{N_{HIT}(term, th)}{N_{target}(term)} \tag{2.12}$$

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

Generally speaking, the performance of an KWS system is defined by the trade-off between p_{HIT} and p_{FA}. In response to demands of the specific application, different measures are considered using these statistics. A Receiver Operating Curve (ROC) for one $term$ is $term$'s $p_{HIT}(term, th)$ as function of $N_{FA}(term, th)$ per hour. Figure Of Merit (FOM) is an upper-bound estimate on KWS accuracy averaged over 1 to 10 false alarms per hour. It can be defined as the area under ROC from 0 to 10 false alarms per hour. The definitions of these metrics are illustrated in Figure 2.13.

However, the FOM metric is somewhat dependent on each terms' threshold and does not convey useful information about how the system would perform with regards to a single th for all queries. Hence the Detection Error Trade-off (DET) Curve, which is a more explanatory metric considering the decision threshold, is introduced. For a given threshold th, the DET curve shows the dependency of $term$'s false alarm probability $p_{FA}(term, th)$(x-axis) to the miss probability $p_{MISS}(term, th)$ (y-axis). Contrary to ROC, the DET curve is a plot for all possible values of the threshold this is why it is widely used in the recent KWS applications [15]. An example of a DET curve is illustrated in Figure 2.14.

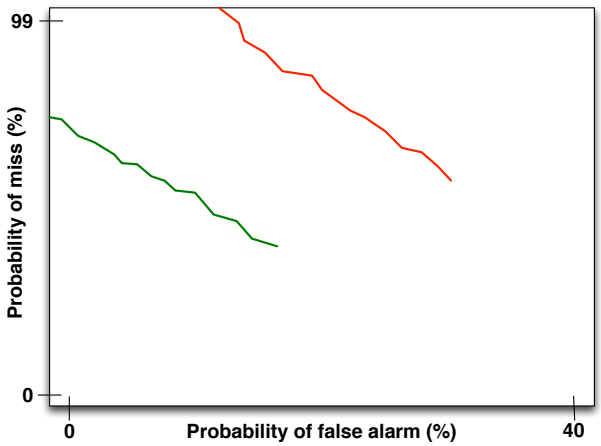

Figure 2.14. Illustration of DET curves for two systems: the system with the green curve depicts a better system.

2.6.3. Term Weighted Value

DET curves can present a visual comparison of two systems. The system whose DET curve is closer to the bottom-right corner of the figure, can be interpreted as a "better" system. In order to be able to compare two systems based on one number, the metric Term Weighted Value (TWV) was introduced. Although helpful in monitoring the system performance, DET curves can not give such a number. The definition of TWV is as follows:

$$TWV(th) = 1 - \underset{term}{average}(p_{MISS}(term, th) + \beta p_{FA}(term, th)) \qquad (2.13)$$

where

$$\beta = \frac{C}{V}(pr_{term}^{-1} - 1) \tag{2.14}$$

C is the cost of incorrect detection (FA), V is the value of correct detection (HIT), pr_{term}^{-1} is the prior probability of the $term$. $\frac{C}{V}$ was set to 0.1 and pr_{term}^{-1} was set to 10^{-4} for NIST STD 2006 evaluations. Then, β is constant equal to 999.9. In other words, β value gives an intuition on how valuable a "HIT" is when compared to many number of false alarms. Hence, TWV gives a number between $-\infty$ and 1. A system that succeeds to detect all of the occurrences of all terms with no false alarms will have a TWV of 1, and a system that gives no outputs will give a TWV of 0. Hence, it is theoretically possible to do worse than doing nothing at all.

Since TWV is a measure depending on a th value, other Term Weighted Value based metrics are introduced such as Actual Term-Weighted Value ($ATWV$), Maximum Term-Weighted Value ($MTWV$), Optimal Term-Weighted Value ($OTWV$) and Supremum Term-Weighted Value ($STWV$). The definitions and interpretations are as follows:

- $ATWV$ represents the system's ability to predict the optimal operating point given the TWV scoring metric. It is the TWV that was calculated for the threshold learned by the system developer during the development experiments.
- $MTWV$ is the maximum value the TWV can get over all th values. The mismatch between $ATWV$ and $MTWV$ signals poor system development and threshold selection.
- $OTWV$ is calculated using the best th value for each query. The mismatch between $MTWV$ and $OTWV$ signals poor score calibration for different queries. If the scores are well calibrated, one single global threshold would suffice to discriminate between HITs and FAs.
- $STWV$ is calculated by only considering the HITs and ignoring the cost of FAs. It is simply the ratio of the system's correctly detected outputs to the actual

number of occurrences averaged over all *terms*. The mismatch between $OTWV$ and $STWV$ signals poor search and scoring methodology. It means that there are FAs that have higher scores than correct HITs.

Clearly;

$$ATWV \leq MTWV \leq OTWV \leq STWV.$$

2.6.4. Normalized Cross Entropy

TWV is a very well defined and a widely used metric for KWS tasks. Though, one peculiarity of the TWV metric is that it is very much related to frequency of the terms. In TWV, missing an infrequent term is considerably more punishable than than missing a frequent one. Furthermore, the TWV metric is calculated over a global single threshold, and because of the miss/false alarm cost ratio the maximization calls for forcing the threshold down for infrequent terms. For this reason, the normalized cross entropy cost function is proposed in [8] that measures the system performance based on system scores, not the system decisions.

To build the statistical detection reasoning for the normalized cross entropy metric, we consider the system scores for each trial $t = (q, x)$ to be , where q is the query and x is the document segment. On optimal decision can be made for a trial if the posterior probability of a HIT is higher than that of not a HIT (denoted !HIT).

$$decision(t) = YES \qquad \Leftrightarrow \qquad \frac{P(HIT|t)}{P(!HIT|t)} > 1 \tag{2.15}$$

From the Bayes rule, we can also write the same rule as a comparison between the likelihood ratios (lr) versus the ratios of priors:

$$decision(t) = YES \qquad \Leftrightarrow \qquad lr_t = \frac{P(t|HIT)}{P(t|!HIT)} > \frac{1 - P_{target}}{P_{target}} \tag{2.16}$$

Hence, if the system scores are well calibrated to be log-likelihood ratios (llr) $\in [-\infty, \infty]$ then, the decision would be

$$decision(t) = YES \qquad \Leftrightarrow \qquad llr_t > -\text{logit}(P_{target}) \tag{2.17}$$

where $\text{logit}(p) = log(\frac{p}{1-p})$.

If we define the sets T_{true} and T_{false} to be the sets containing ground truths of the trial matching the segment or not, respectively, the posterior probabilities shown in (2.15) can be obtained as

$$\begin{aligned} P(HIT|t) &= \sigma(llrt + \text{logit}(P_{target})) \\ P(HIT!|t) &= \sigma(-(llrt + \text{logit}(P_{target}))) \end{aligned} \tag{2.18}$$

where

$$\sigma(z) = \frac{1}{1 + e^{-z}} \tag{2.19}$$

Then the logarithmic cost functions for a trial t can now be defined as

$$C_{log}(llr_t) = \begin{cases} -\text{log}(\sigma(llr_t + \text{logit}(P_{target}))) & \text{if} \quad t \in T_{true} \\ -\text{log}(\sigma(-(llr_t + \text{logit}(P_{target})))) & \text{if} \quad t \in T_{false} \end{cases} \tag{2.20}$$

Over the whole set of trials, the empirical cross entropy (in bits) can be calculated as:

$$C_{xe} = \frac{1}{\text{log}(2)} . \Big(\frac{P_{target}}{|T_{True}|} \sum_{t \in T_{True}} C_{\text{log}}(llr_t) + \frac{1 - P_{target}}{|T_{False}|} \sum_{t \in T_{False}} C_{\text{log}}(llr_t) \Big) \tag{2.21}$$

The empirical system is normalized by the trivial system with the prior entropy

$$C_{xe}^{prior} = \frac{1}{\text{log}(2)} . \Big(P_{target} \text{log}(\frac{1}{P_{target}}) + (1 - P_{target}) \text{log}(\frac{1}{1 - P_{target}}) \Big) \tag{2.22}$$

Finally, the normalized cross entropy (C_{nxe}) is calculated as:

$$C_{nxe} = \frac{C_{xe}}{C_{xe}^{prior}} \tag{2.23}$$

Assuming that the system scores are log likelihood ratios for each trial hypothesis, C_{nxe} measures the fraction of information that is not provided by the system. It is a measure to evaluate the goodness of system scores, not the decisions. In the limit, the target scores are desired to be $+\infty$ and scores of $-\infty$ are desired for non-target hypotheses. This metric shows how much sensitive the system scores to different operating points (of β) and different threshold values.

2.6.5. C_{nxe} vs TWV

The main difference between the C_{nxe} and TWV metrics can be described as follows: "C_{nxe} measures the goodness of score values while TWV measures the goodness of discrete YES/NO decisions" [96]. The C_{nxe} is a measure of how much separated relevant and irrelevant parts (trials) are, while explaining how much sensitive system decisions are to threshold variations. A perfect system, therefore, would be expected to have targets with scores ∞ and non-targets with scores $-\infty$. For a system like that, the decisions would always be correct, no matter the threshold we chose. It should be noted that C_{nxe} is calculated over scores and does not depend on decisions. As will be discussed in detail in Chapter 5, system development involves the estimation of a score normalization and a global threshold (that works on all keywords) to maximize TWV, which is obtained by minimizing p_{FA} and p_{MISS}. In other words, the goal is to reduce the number of wrong decisions as much as possible. One way to achieve this goal, is to separate target and non-target scores as much as possible, i.e. to minimize C_{nxe}. Therefore, minimizing C_{nxe} will help operating on a wider range of threshold values, which will give rise to low Pp_{FA} and p_{MISS}, thus good TWV values. Nevertheless, building a system to optimize TWV does not necessarily produce good C_{nxe} scores.

2.6.6. System Calibration and Minimum Normalized Cross Entropy

The cross entropy measures not only the separation between target and non-target trials but also the calibration of scores as log likelihood ratios. To evaluate the score calibrations with respect to the C_{nxe}, we define an affine transformation of scores

$$\hat{llr}_t = \gamma.llr_t + \delta \tag{2.24}$$

where γ and δ are the optimization parameters to obtain the minimum value of C_{nxe}, i.e. C_{nxe}^{min}. If we refer the normalized cross entropy, obtained by transformed scores $\hat{llr}_t$ as $\hat{C}_{nxe}$, C_{nxe}^{min} is obtained by

$$C_{nxe}^{min} = \min_{\gamma,\delta} \left\{ \hat{C}_{nxe} \right\} \tag{2.25}$$

3. QUERY MODELING AND TEMPLATE MATCHING BASED SEARCH WITH PSEUDO QUERIES

In this thesis, we extend the template-based techniques of QbE-STD task to KWS, applied for low resource languages. With this, we aim to implement a novel methodology to contribute to the existing techniques of retrieving especially the OOV words, as well as the IV words. The primary methodology of search in the proposed technique is the template matching based dynamic search. As stated in Chapter 2 the most practiced and the most useful set of features of acoustic units are Gaussian posteriorgrams or phone posteriorgrams. Hence, the first action to take is to obtain the phonetic posteriorgrams from the speech document.

3.1. Posteriorgram Representation

A posteriorgram is phone by frame matrix depicting the posterior probability mass function of the corresponding frame over the set of phonetic units [78]. The posteriorgram is mainly obtained by means of a DNN or a long short-term memory recurrent neural network (LSTM RNN), shortly referred to as LSTM in the literature. Once the front end features, such as MFCCs or perceptual linear prediction (PLP) coefficients, are obtained from the raw speech for each window frame, the fully connected output layer activations of the neural network are trained to mimic the one-hot vector label of that frame.

Briefly, a posteriorgram of a speech utterance $\{\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_T\}$ is a matrix $\mathbf{P}$ of size KxT, with

$$\mathbf{P}(k,t) = P(k|\mathbf{x}_t), \quad k = 1 \cdots K, \quad t = 1 \cdots T$$

where K is the number of phonemic units in the language. A sample posteriorgram matrix from a speech segment is demonstrated in Figure 3.1.

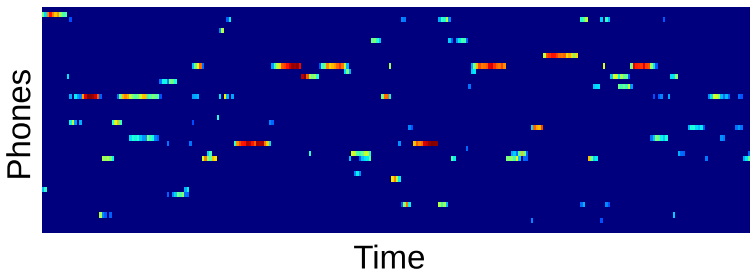

Figure 3.1. A sample posteriorgram segment.

In this thesis work, the posteriorgrams are obtained using the Kaldi speech recognition toolkit [1] trained with the HMM-DNN [2] recipe [97]. During training and decoding we used PLP features and the DNN [3] architecture of the Kaldi recipe based on p-norm activations [98]. We also store the posteriorgram and the phoneme-level alignment obtained from the training data to be used in the query model and distance metric learning, which will be explained in the following sections.

3.2. Modeling of Text Queries as Posteriorgrams

The acquisition of posteriorgrams from the acoustic features of speech document is straightforward. However, in order to conduct the template matching based search, we need the text queries to be modeled as posteriorgrams as well. In the next section, we describe this modeling. The first step in query modeling is transforming the graphemic sequence of letters into sequences of phonemic units or phonemes. For IV keywords this is done by the usage of the pronunciation lexicon. For the OOV keywords, or for the keywords including OOV words, we use Sequitur grapheme to phoneme (G2P) converter [99]. The G2P converter is trained to estimate the most probable sequence of phonemes for a given sequence of letters. The G2P training is done with joint-sequence models, using the limited lexicon as training data. Using these phonemic indexes we model a posteriorgram for each phoneme using two methodologies:

[1] version 5.1.58
[2] babel/s5d
[3] tri6_nnet

3.2.1. Binary Query Modeling

One very obvious choice of query modeling is treating the phoneme indexes as "sure events" of frames and concatenating the one-hot vectors of the corresponding phone indexes. Such a query model would simulate the output of a DNN which decodes its frames with perfect certainty. This method of modeling the query is referred to as "binary query modeling" in this thesis and the schematic is depicted in Figure 3.2.

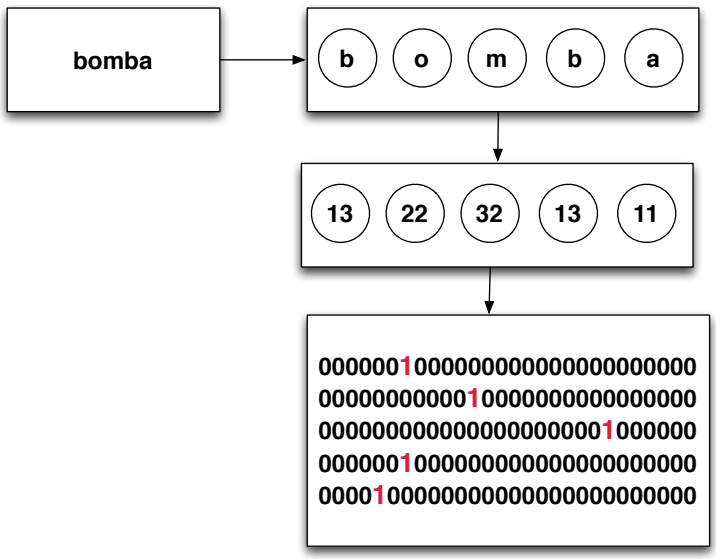

Figure 3.2. Binary query modeling scheme.

The binary query model in Figure 3.2 can be considered as a posteriorgram where the phonemes have durations of one frame each. This posteriorgram model is to be used in a DTW-based search with the document posteriorgram. In theory, this duration modeling of one frame would give the same total similarity (or distance) value accumulated on the alignment path with an actual one-hot posteriorgram query with corresponding phoneme durations. However, for the locations of the document where the query does not exist, the total distance value would be incomparable to the true examples. To address this conundrum, we propose modeling the phoneme durations in this pseudo query by using the average durations for each phoneme. For this, we replicate each one hot vector with the numbers equal to the average frame durations estimated from the training alignment for the corresponding phoneme.

3.2.2. Average Query Modeling

The second query modeling technique aims to model the phone confusions of the decoder. For this, instead of using one-hot vectors for each phoneme index, we use the average of the posterior vectors that the DNN decoder outputs for that phoneme. To model the pseudo queries we replicate the average posterior vectors as many times as the average durations for the corresponding phonemes. This method of modeling the query is referred to as "average query modeling". The schematic is depicted in Figure 3.3. As a visual example, the query models for a sample word can be seen in Figure 3.4.

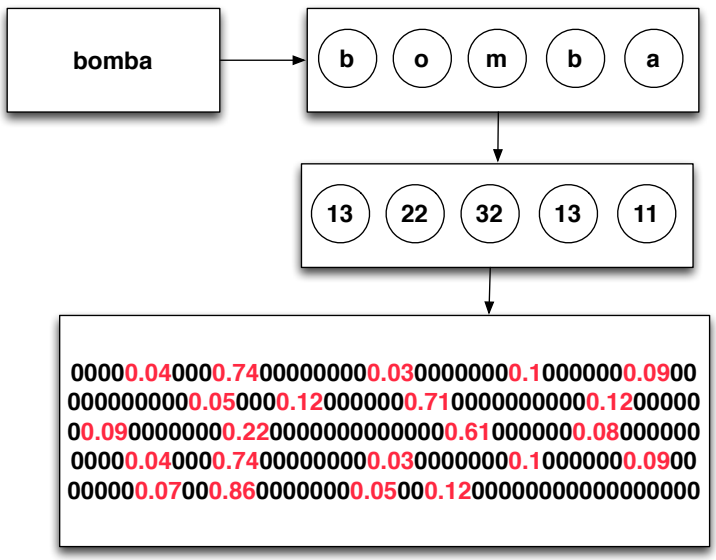

Figure 3.3. Average query modeling scheme

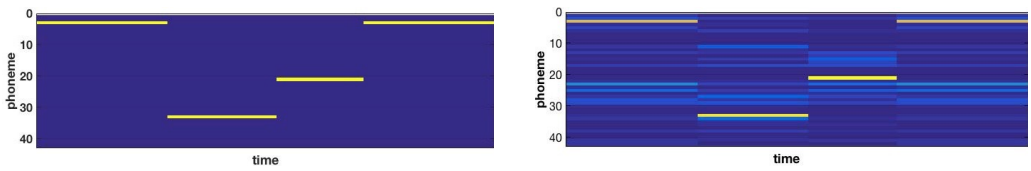

Figure 3.4. The pseudo posteriorgram models of a sample word '*arma*' obtained using the binary (left) and average (right) query modeling.

3.3. Dynamic Search with Pseudo Queries

Once we obtain the pseudo queries using the techniques explained in the previous section, we apply the sDTW algorithm to obtain the acoustically similar segments

of the document. We altered the search methodology of [100] such that the path length averages are obtained and the optimal beginning of the subsequence is stored in memory, so that the need for back tracing is avoided. The sDTW for KWS algorithm is given in Figure 3.5.

Algorithm 3: sDTW for KWS

for $i = 1$ to M, $j = 1$ to N **do**
  if i=1 **then**
    $\mathcal{A}(1, j) = d(\mathbf{q}_1, \mathbf{x}_j)$
    $\mathcal{L}(1, j) = 1$, $\mathcal{B}(1, j) = j$
  else if j=1 **then**
    $\mathcal{A}(i, 1) = \sum_{k=1}^{i} d(\mathbf{q}_k, \mathbf{x}_1)$
    $\mathcal{L}(i, 1) = i$, $\mathcal{B}(i, 1) = 1$
  else
    $\Omega = \{(i, j-1), (i-1, j), (i-1, j-1)\}$
    $(r, s) = \underset{\forall (p,q) \epsilon \Omega}{\operatorname{argmin}} \frac{\mathcal{A}(p, q) + d(\mathbf{q}_i, \mathbf{x}_j)}{\mathcal{L}(p, q) + 1}$

    $\mathcal{A}(i, j) = \mathcal{A}(r, s) + d(\mathbf{q}_i, \mathbf{x}_j)$
    $\mathcal{L}(i, j) = \mathcal{L}(r, s) + 1$, $\mathcal{B}(i, j) = \mathcal{B}(r, s)$
  end if
end for

Figure 3.5. sDTW Algorithm for KWS.

Definitions of the variables in this algorithm can be found below:

- $\mathcal{X} = \{\mathbf{x}_1, \mathbf{x}_2, \cdots \mathbf{x}_N\} \in \mathcal{R}^{K \mathrm{x} N}$: the document posteriorgram
- $\mathcal{Q} = \{\mathbf{q}_1, \mathbf{q}_2, \cdots \mathbf{q}_M\} \in \mathcal{R}^{K \mathrm{x} M}$: the query posteriorgram
- $\mathcal{A} \in \mathcal{R}^{N \mathrm{x} M}$: the accumulated distance matrix. $\mathcal{A}(i, j)$ stores the total distance of the best alignment between $\mathcal{Q}$ and the best subsequence of $\mathcal{X}$ that ends at $(\mathbf{q}_i, \mathbf{x}_j)$
- $\mathcal{L} \in \mathcal{Z}^{+ N \mathrm{x} M}$: the alignment lengths matrix. $\mathcal{L}(i, j)$ stores length of the best alignment between $\mathcal{Q}$ and the best subsequence of $\mathcal{X}$ that ends at $(\mathbf{q}_i, \mathbf{x}_j)$
- $\mathcal{B} \in \mathcal{Z}^{+ N \mathrm{x} M}$: the path beginnings matrix. $\mathcal{B}(i, j)$ stores the beginning of the

path of the best alignment between $\mathcal{Q}$ and the best subsequence of $\mathcal{X}$ that ends at $(\mathbf{q}_i, \mathbf{x}_j)$

As the time progresses along the sequences, the optimal beginning frame information is transferred dynamically. The ending frame of the best subsequence of $\mathcal{X}$ that includes the whole query can be found on the last row of $\mathcal{A}$. A search score that favors longer alignments can be calculated by incorporating the actual length information of the aligned subsequence:

$$\mathbf{score}(j) = 1 - \frac{\mathcal{A}(M, j)}{\mathcal{L}(M, j)} \tag{3.1}$$

Here, **score** is a vector, where its j^{th} element stores the score of the most similar subsequence of $\mathcal{X}$ that ends at $\mathbf{x}_j$. The subsequence of $\mathcal{X}$, that has the best similarity to the query is the one yielding the maximum score. The subsequences whose score is above a certain threshold is marked as " hit". The beginning and ending frames of the hit can be calculated as:

$$\begin{aligned} t_{end} &= \arg\max(\mathbf{score}) \\ t_{beg} &= \mathcal{B}(M, t_{end}) \\ hit &= \mathcal{X}(t_{beg} : t_{end}) \end{aligned} \tag{3.2}$$

From Equation 3.1, it is easy to see that the score of the sDTW alignment ϕ between $\mathcal{Q}$ and the corresponding subsequence of $\mathcal{X}$ can be formulated as:

$$\text{score} = 1 - \frac{1}{\text{length}(\Phi)} \sum_{(i,j)\in\Phi} d(\mathbf{q}_i, \mathbf{x}_j) \tag{3.3}$$

We can argue that, $\mathcal{B}(M, j)$ points at the best starting point of the subsequence that ends at (j). Naturally, the keyword may exist multiple times in the search document or the utterance. Hence, the algorithm in 3.6 for a recursive search is implemented to address this issue.

Algorithm 4: *searchmultiple*(**score**, $\mathcal{B}$, th, **interval**)

Require: **score**, $\mathcal{B}$, th, **interval** {The initial interval is the whole document}

Ensure: list of **HITs**

1: $hit_{end} \leftarrow argmax(\textbf{score}(\textbf{interval}))$
2: $hit_{begin} \leftarrow \mathcal{B}(M, end)$
3: **hit** $\leftarrow \mathcal{X}(begin : end)$
4: **if score**$(hit_{end}) \geq th$ **then**
5: **HITs** $\leftarrow$ **HITs** $\cup$ **hit**
6: **pre–interval** $\leftarrow [1 :$ **interval**$(hit_{begin})]$
7: **post–interval** $\leftarrow [$**interval**$(hit_{end}) : N]$
8: $temp1\text{–}hit_{end} \leftarrow argmax(\textbf{score}(\textbf{pre–interval}))$
9: $temp1\text{–}hit_{begin} \leftarrow \mathcal{B}(M, temp1\text{–}hit_{end})$
10: **temp1–hit** $\leftarrow \mathcal{X}(temp1\text{–}hit_{begin} : temp1\text{–}hit_{end})$
11: $temp2\text{–}hit_{end} \leftarrow argmax(\textbf{score}(\textbf{post–interval}))$
12: $temp2\text{–}hit_{begin} \leftarrow \mathcal{B}(M, temp2\text{–}hit_{end})$
13: **temp2–hit** $\leftarrow \mathcal{X}(temp2\text{–}hit_{begin} : temp2\text{–}hit_{end})$
14: **if score**$(temp1\text{–}hit_{end}) \geq th$ **then**
15: *searchmultiple*(**score**, $\mathcal{B}$, th, **pre–interval**)
16: **end if**
17: **if score**$(temp2\text{–}hit_{end}) \geq th$ **then**
18: *searchmultiple*(**score**, $\mathcal{B}$, th, **post–interval**)
19: **end if**
20: **end if**

Figure 3.6. Recurrent sDTW search algorithm for multiple keywords.

Briefly, the main components of this new sDTW based methodology can be summarized as follows:

- The DNN is trained using the limited training data, the posteriorgram for the document and the training data as well as the phoneme alignments are obtained,
- The phoneme statistics such as the average durations and the average posterior vectors are obtained from the training alignment,
- The text query is modeled as pseudo posteriorgrams using the methodologies explained in the previous section,
- sDTW based search is conducted to obtain the list of similar subsequences along with their similarity scores.

The flowchart of this methodology is demonstrated in Figure 3.7.

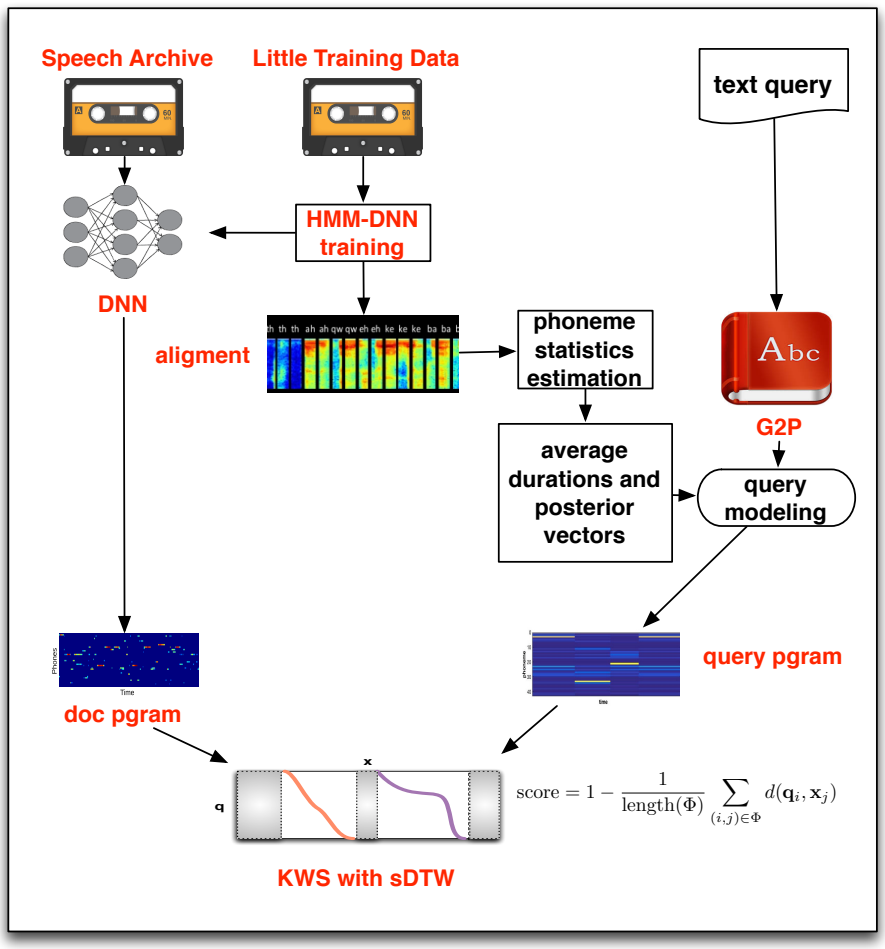

Figure 3.7. Flowchart of template matching-based search with pseudo queries.

The experimental results of this methodology along with the analysis of usage of different query modeling techniques and different distance measures were published in

conferences [18] and [19][4] .

The two very important conclusions of this work were:

(i) The new scheme of pseudo query modeling and template matching based search actually works and helps the LVCSR based systems in combination, especially for the OOV terms

(ii) A very important component of the sDTW based KWS is the distance metric used between the frames of the query and the document (seen on line-6 of the Algorithm 3 and Equation 3.3), and therefore, the choice of distance metric and query modeling matter.

Hence, in the next chapter, we introduce a novel distance metric learning methodology and use the learned distance in the sDTW. Furthermore, we extend the distance metric learning model to include learning of the query model frames (instead of one-hot or average vectors) to test if such a distance metric learning schematic would work better than the existing distance metrics.

[4]This paper was given the "Best Student Presentation Award" by the organization committee [20]

4. DISTANCE METRIC LEARNING FOR KEYWORD SEARCH

In the previous chapter, we showed that given a word based query, acoustically similar subsequences of the document can be obtained by means of the sDTW algorithm even though we use pseudo, artificially modeled queries. The similarity score of this subsequence is obtained on the alignment path with Equation 3.3.

4.1. The Importance of Distance Metric in sDTW-Based KWS

As can be seen from Equation 3.3, the distance measure used between the frames holds a significant importance on the detection score. And this detection score is used to mark the hypotheses as relevant or irrelevant. Therefore, the score, hence the distance metric, effects the performance of KWS directly. Since the phone posterior vectors are used as the representation of frames, the most widely used distance metrics are cosine distance and logarithmic cosine distance measures [101]. Each metric has its own interpretations. The Euclidean and cosine distance metrics have geometric meanings. The Euclidean distance metric (Equation 4.1) between $\mathbf{x}$ and $\mathbf{y}$ is the length of the shortest path between them in the Euclidean space. The cosine distance (Equation 4.2), on the other hand, is based on the angle between $\mathbf{x}$ and $\mathbf{y}$. The logarithmic cosine distance (Equation 4.3) has been shown to be very useful in DTW-based QbE-STD tasks using posteriorgrams, since it has a probabilistic interpretation in that it can be considered as the negative log probability of the frames belonging to the same distribution [73]. Furthermore, all of these metrics, use the inner-product of the two vectors as a similarity measure. It can be argued that these similarity values are transformed into dissimilarities (or distances) by an application of a kernel function. The kernels applied to the inner-product similarity values, for vectors with unit norms, can be seen on Figure 4.1 for Euclidean, cosine and logarithmic cosine distance measures.

$$d_{\text{euc}}(\mathbf{x}, \mathbf{y}) = \sqrt{\|\mathbf{x}\|^2 + \|\mathbf{y}\|^2 - 2\mathbf{x}^T\mathbf{y}} \tag{4.1}$$

$$d_{\cos}(\mathbf{x}, \mathbf{y}) = 1 - \frac{\mathbf{x}^T\mathbf{y}}{\|\mathbf{x}\|\|\mathbf{y}\|} \tag{4.2}$$

$$d_{\text{log-cos}}(\mathbf{x}, \mathbf{y}) = -\log(\frac{\mathbf{x}^T\mathbf{y}}{\|\mathbf{x}\|\|\mathbf{y}\|}) \tag{4.3}$$

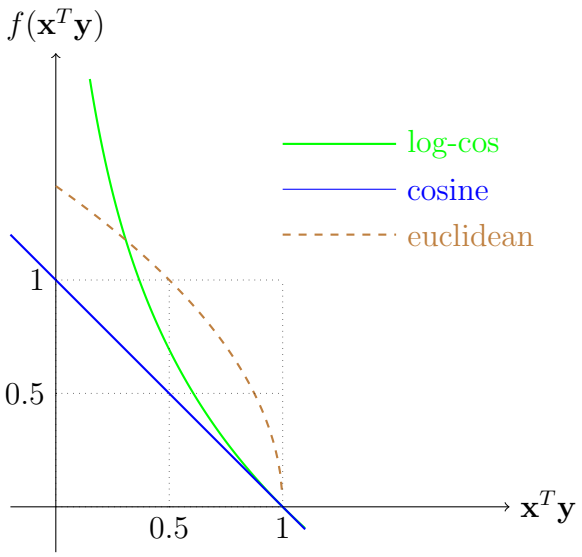

Figure 4.1. Analysis of kernel functions for each distance metric: the x-axis represents the inner product and the y-axis represents the kernel that makes it a dissimilarity.

4.2. What to "Learn" for a Distance Metric?

Although the above mentioned metrics are useful in certain applications and provide mathematically valuable similarity values, none is perfectly tailored to reflect the characteristics of the distribution of the data. One very intuitive solution to adjusting the distance metric for the data in hand would be to use weighed inner-products with covariances estimated from data as in Mahalonobis distance, yet the question of which dissimilarity kernel to use still remains. We would like to find a distance metric which will return more desirable results when used in sDTW based KWS scheme. Our goals in distance metric learning (DML) can be enumerated as follows:

- We would like the new distance metric to have smaller distance values between frames that belong to the same class (phoneme) and larger distance values between frames that belong to different classes, so that we get more discrimination along the alignment path of sDTW.
- The phonetic confusions that the decoder has for different phonemes should be modeled and included in the distance metric. For example, if two phonemes are generally confused with each other by the decoder, we would like the distance metric to model this, and output a lower distance value for instances of these phonemes.
- The distance values of the new distance metric must be in an interpretable and manageable range so that the it also helps the scoring. For example the distance value of log-cosine metric between foes can go up to ∞ which may cause problems along the alignment path.

4.3. Distance Metric Learning Neural Network Model

In order to achieve the goals proposed in the previous section, we propose the neural network-based DML model in Figure 4.2, similar to the Siamese networks used in signature [102] and face [103] verification applications. With this, we aim to obtain a better discrimination in frame level, lower distance between examples of the same phoneme, higher distance between examples of different phonemes by incorporating

phone confusions into the distance value. We also aim to have the distance value to be in $[0, 1]$ such that the it can be interpreted as the probabilities that two frames belong to the same phoneme or different phonemes. We call the distance obtained from this network "sigma distance" since sigmoid non-linearity is used to map the weighted inner-product similarity to the desired range, with weights and the bias learned in training. Here, we use the term distance metric for our system loosely, since the output of the model does not satisfy the axioms of metric spaces, yet it merely successfully provides a solution to our above stated objectives.

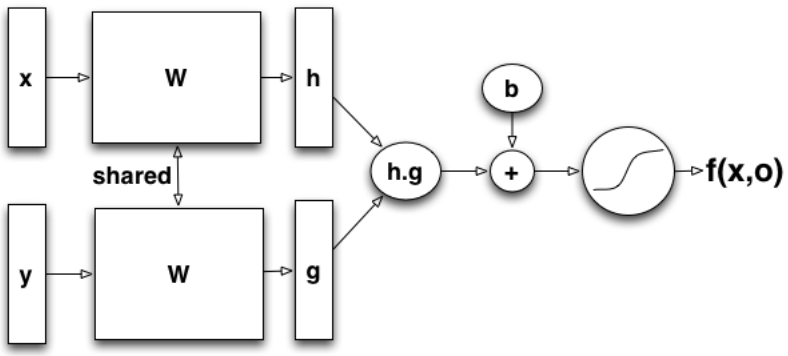

Figure 4.2. DML Siamese neural network model. The new similarity value $f(\mathbf{x}, \mathbf{y})$ will be the output of this network.

The first layer maps the inputs onto a (possibly lower dimensional) subspace using the shared weight matrix $\mathbf{W}$. The output vectors of the first layer are denoted $\mathbf{h}$ and $\mathbf{g}$.

$$\begin{aligned} \mathbf{h} &= \mathbf{W}\mathbf{x} \\ \mathbf{g} &= \mathbf{W}\mathbf{y} \end{aligned} \tag{4.4}$$

On the second layer, similarity of the two inputs is found by taking the inner product of the outputs of the first layer. This similarity measure is then ran through a sigmoid nonlinearity given in Equation 2.19 to get the desired range. The most discriminative center of the sigmoid is achieved by introducing a bias term to the output of the second layer. The optimization parameters of this neural network model

are the shared weight matrix ($\mathbf{W}$) and the bias (b) value of the sigmoid.

$$\begin{aligned} f(\mathbf{x},\mathbf{y}) &= \sigma(\mathbf{h}^T\mathbf{g}+b) = \sigma(\mathbf{x}^T\mathbf{W}^T\mathbf{W}\mathbf{y}+b) \\ d_\sigma(\mathbf{x},\mathbf{y}) &= 1 - f(\mathbf{x},\mathbf{y}) \end{aligned} \tag{4.5}$$

4.3.1. DML Training

As can be seen on Figure 4.2, the network takes a pair of inputs and emits a scalar similarity value. So, we can consider the training set as triplets $(\mathbf{x}_t, \mathbf{y}_t, r_t)$ where r_t is the label indicating the kinship of the inputs $\mathbf{x}_t$ and $\mathbf{y}_t$. For the sake of simplicity, we call the pairs $(\mathbf{x}_t, \mathbf{y}_t)$ "friends" if they belong to the same phone, and "foes" otherwise. Then, as the labels r_t, we use 1 for friends and 0 for foes, in other words

$$r_t = \begin{cases} 1, & \text{if} \quad \text{class}(\mathbf{x}_t) = \text{class}(\mathbf{y}_t) \\ 0, & \text{if} \quad \text{class}(\mathbf{x}_t) \neq \text{class}(\mathbf{y}_t) \end{cases} \tag{4.6}$$

Clearly, for a training set of more than two classes, the size of the foes class will be significantly larger than the size of friends. In order to solve this problem, we separated the training dataset into friends and foes. Then in each epoch, we trained the network using the whole friends set in random order and a random subset of foes with the size of the friends set. We also applied prior equalization on the phoneme classes by taking the same number of samples from each phoneme class into the distance learning training.

Since we have the objective of interpreting the system output as a probability value, we used the cross-entropy (CE) objective function in the back-propagation. It can be considered as an objective of increasing the likelihood of friends to output 1 and vice versa. The objective function is then defined as:

$$J_{CE}(\mathbf{W}, b; \mathbf{x}_t, \mathbf{y}_t, r_t) = -r_t \log(f(\mathbf{x}_t, \mathbf{y}_t)) - (1 - r_t)\log(1 - f(\mathbf{x}_t, \mathbf{y}_t)) \tag{4.7}$$

If we express $f(\mathbf{x}_t, \mathbf{y}_t)$ as f for simplicity, the gradient with respect to the parameters are found as follows:

$$\begin{aligned} \triangle b = \frac{dJ}{db} &= \frac{dJ}{df}\frac{df}{dz}\frac{dz}{db} \\ &= (\frac{r-f}{f(1-f)})(f(1-f))(1) = r - f \end{aligned} \tag{4.8}$$

and

$$\begin{aligned} \triangle \mathbf{W} = \frac{dJ}{d\mathbf{W}} &= \frac{dJ}{df}\frac{df}{dz}\frac{dz}{d\mathbf{W}} \\ &= (\frac{r-f}{f(1-f)})(f(1-f))(\frac{d}{d\mathbf{W}}(\mathbf{x}^T\mathbf{W}^T\mathbf{W}\mathbf{y})) \\ &= (r-f)\mathbf{W}(\mathbf{x}\mathbf{y}^T + \mathbf{y}\mathbf{x}^T) \end{aligned} \tag{4.9}$$

where $z = \mathbf{h}^T\mathbf{g} + b = \mathbf{x}^T\mathbf{W}^T\mathbf{W}\mathbf{y} + b$.

With the gradients in hand and using learning rates μ and η, we update the parameters with on-line gradient descent as,

$$\mathbf{W} \leftarrow \mathbf{W} - \mu\triangle\mathbf{W} \quad \text{and} \quad b \leftarrow b - \eta\triangle b$$

The flowchart and algorithm of the DML training are given in Figures 4.3 and 4.4, respectively.

4.3.2. Sigma Distance Discrimination Performance

To see the discriminative power of the DML network, we observed the statistics of the distances between friends and foes; and compared this statistics between the

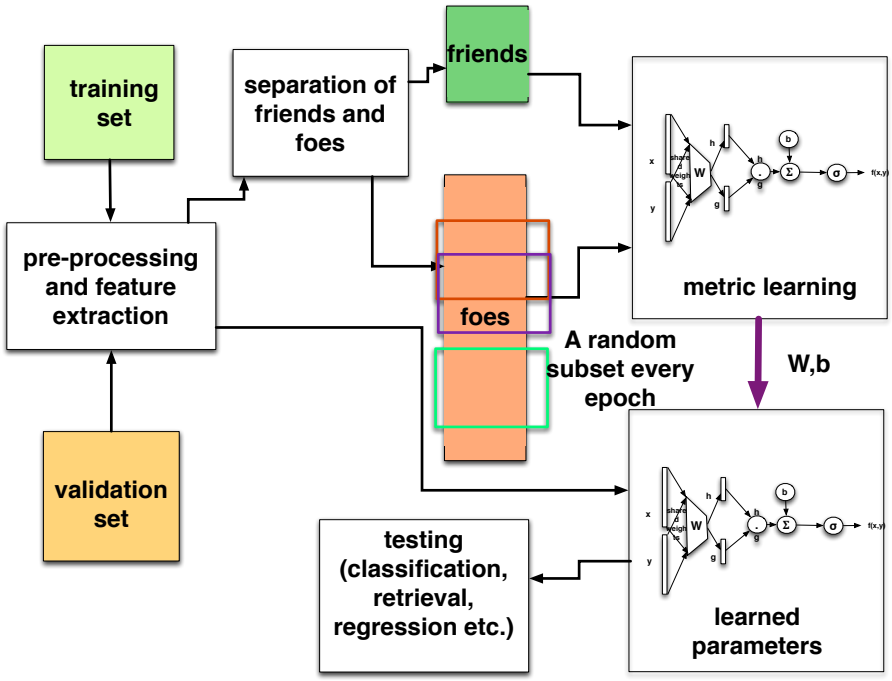

Figure 4.3. Flowchart of DML training.

sigma distance and other common distance measures. For this experiment, we used 200 random frames per phoneme class, from different speakers for training the DML network. The detailed explanations of the size, source and language of the training sets are deferred to Chapter 6. When these frames are combined into DML training pairs, a training set of approximately 36 million pairs was obtained. For validation, we used another random subset of the training data, consisting of 100 random samples per phoneme, about 8.8 million pairs. We have seen that the Gini mean [104] of friends (mean of distances between friends - also called the statistical dispersion) and the Gini mean of foes get farther from each other with sigma distance when compared to other known distance metrics. In other words, friends get closer as foes get farther. This was one of our key goals for achieving discrimination in detection. The normalized histograms of the distances between friends and foes using different distance measures can be seen on Figure 4.5. We see that euclidean distance and logarithmic cosine distance does not perform well on discrimination of friends and foes. Histogram of cosine distance looks similar to that of sigma distance, however, while in cosine distance the distance histogram of friends looks a flat, it decays as required in sigma distance. The first and second order statistics of the dispersions can be seen on Table 4.1.

Algorithm 5: DML

Require: $\mathbf{t}_{1:M}$ and $c(\mathbf{t})_{1:M}$ (labeled set of k classes)

Ensure: $\mathbf{W}$ and b for $f(\mathbf{x}, \mathbf{y})$

1: Separate the dataset into friends and foes:

$friends = \{\mathbf{x}^{(i)}, \mathbf{y}^{(i)}, r^{(i)} = 1\}_{i=1}^{N}$

$foes = \{\mathbf{x}^{(i)}, \mathbf{y}^{(i)}, r^{(i)} = 0\}_{i=1}^{kN}$

2: Initialize $\mathbf{W}$ and b randomly

3: Set the learning rates μ and η

4: **repeat**

5: choose a random patch of size N from *foes* and call this set *sub_foes*

6: $\mathcal{X} = friends \cup sub\_foes$

7: **for** $\forall(\mathbf{x}^{(i)}, \mathbf{y}^{(i)}) \in \mathcal{X}$ in random order **do**

8: Calculate $f^{(i)}$ using $f(\mathbf{x}, \mathbf{y}) = \sigma(\mathbf{h}^T\mathbf{g} + b)$

9: Calculate gradients using

$\triangle\mathbf{W} = (r^{(i)} - f^{(i)})\mathbf{W}(\mathbf{x}^{(i)}\mathbf{y}^{(i)T} + \mathbf{y}^{(i)}\mathbf{x}^{(i)T})$

$\triangle b = (r^{(i)} - f^{(i)})$

10: update parameters

$\mathbf{W} \leftarrow \mathbf{W} + \mu\triangle\mathbf{W}$

$b \leftarrow b + \eta\triangle b$

11: **end for**

12: **until** convergence

Figure 4.4. Distance metric learning algorithm.

4.3.3. DML as a New Kernel

Discussion on the proposed new distance metric can now be extended to the kernel argument presented in Figure 4.1. The kernel function of the sigma distance metric on the (now learned) inner product space is presented in Figure 4.6 along with kernels of other distance metrics. The smooth and symmetric behavior of the sigmoid kernel function could be desirable for KWS. We see that for low similarity values, the log-cosine distance gives extremely high distance values. This may prevent us finding keywords when there is a lot of pronunciation variability and phone confusion.

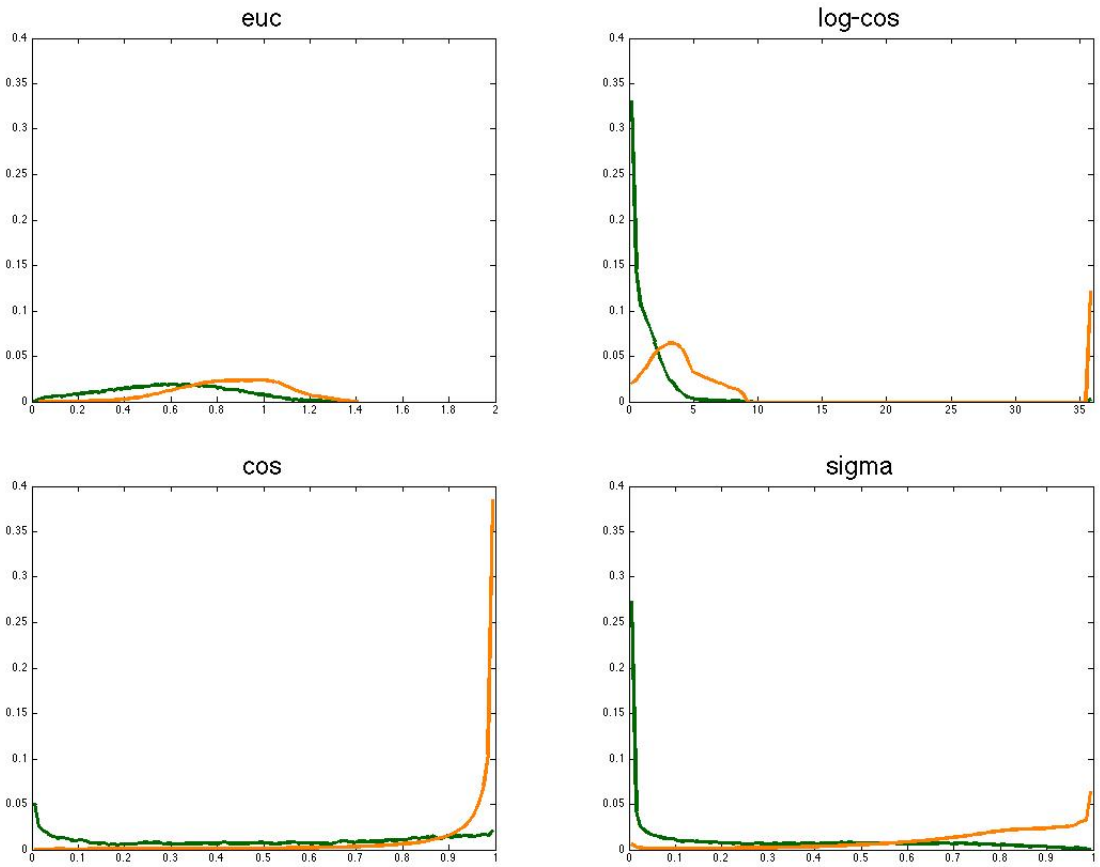

Figure 4.5. Dispersion histograms: orange lines denote the histogram of foes and the green lines denote friends.

Table 4.1. Dispersion statistics of different distance metrics.

DISTANCE/SUBSET	friends		foes	
	mean	variance	mean	variance
euclidean	0.5930	0.0685	0.8603	0.0455
log-cos	1.2514	4.4896	7.6956	114.9505
cos	0.5130	0.1122	0.9123	0.0263
sigma with initial weights	0.3313	0.0019	0.3714	0.0001
sigma	0.2760	0.0849	0.7559	0.0489

However, for sigma distance, this is not the case. For sigma distance, similar enough instances will have a distance of approximately 0, and dissimilar enough instances will have a distance of 1. This proximity will be controlled by the slope of the sigmoid. Furthermore, for the sigma distance, the inner product of two vectors may be increased or decreased with the $\mathbf{W}$ matrix to meet the desired distance value, whereas this is not an option for the other distance measures.

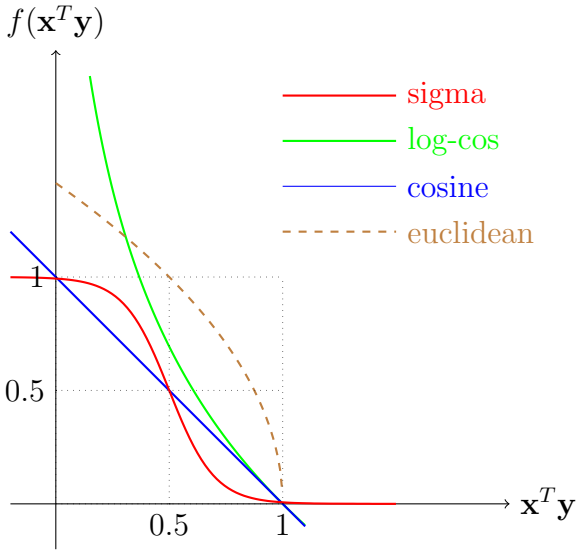

Figure 4.6. Comparison of the kernel function for new distance metric with other distance metrics. The sigma distance parameters are taken to be $\mathbf{W} = 6I$ and $b = -3$ to fit in the same scale.

4.3.4. KWS with the New Sigma Distance

We extend our methodology explained in Chapter 3 to include the DML parameter estimation. We use the alignment from which we estimate the phoneme statistics for DML training. This way the decoder characteristics are conveyed to the new distance metric. For KWS we use the average query model with the new distance metric with learned parameters $\mathbf{W}$ and b. The flowchart of the KWS methodology using the DML training is given in Figure 4.7. The usage of such a learned distance metric learning was tested on a low resource language training set on a KWS experiment and was published in [21]. The experimental results are presented in Chapter 6 along with other techniques for a better comparison of different techniques, datasets and languages

altogether.

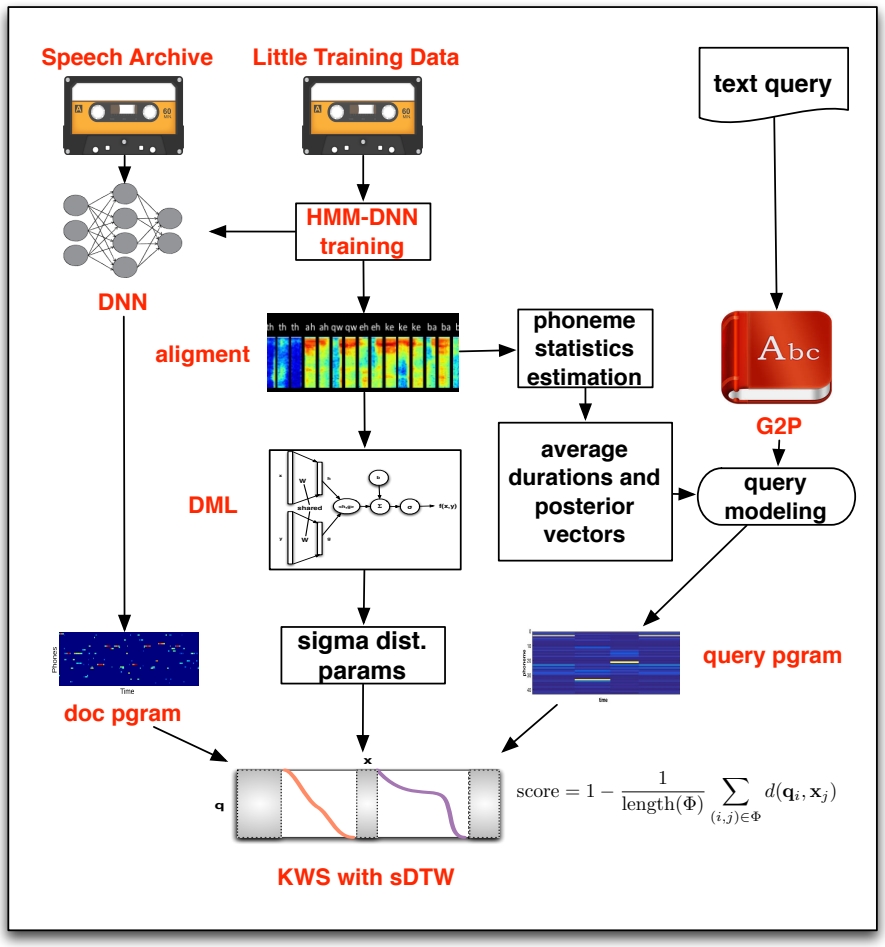

Figure 4.7. sDTW-based KWS flowchart with DML and the usage of the new sigma distance measure.

4.4. Joint Learning of Query Model and Distance Metric

It was shown in [19] that use of different distance measures and query models in the posteriorgram based KWS affects the performance of the system. We first addressed the choice of distance metric in the previous section by learning a new distance metric, which we call sigma distance. In KWS, we modeled the queries with the average query modeling and observed that the learned distance metric works better than the other distance metrics using different query modeling techniques. However, we can still question if selection of average query vectors for each phoneme is actually the optimal query model for this new distance metric. Hence, we extend this research with the aim to learn a distance metric and the appropriate query model for it. To incorporate the query modeling into DML, we add an extra layer at the query side and call it Joint DML (JDML). The proposed JDML model is shown in Figure 4.8. In the following subsections we justify such an extended model with two approaches and interpretations.

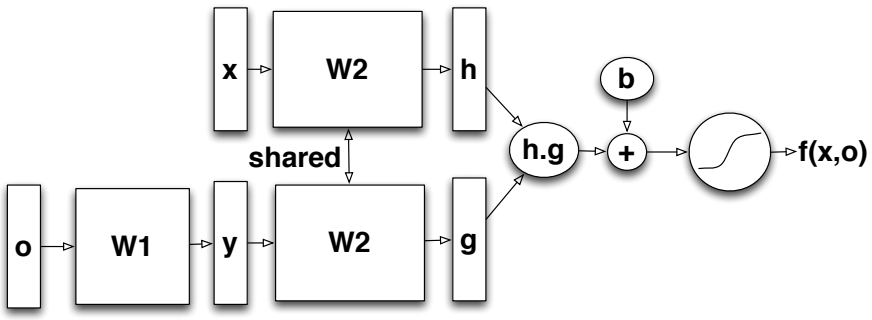

Figure 4.8. JDML asymmetric Siamese neural network model.

While the goal of DML is to learn a distance measure that models the frame level phone confusions, minimize the average distance between friends and maximizes the average distance between foes, JDML is optimized for KWS purposes. Its goal is to simultaneously learn proper representations for each class (phoneme) and the distance parameters. Hence, now the input of the network is a pair, composed of the posterior vectors of the training alignment and a one-hot vector $(\mathbf{x}_t, \mathbf{o}_t)$. The output labels are the desired sigma similarities, given as

$$r_t = \mathbf{o}_t[\text{class}(\mathbf{x}_t)]$$

Simply, $\mathbf{x}_t$ is a frame belonging to either of the classes and $\mathbf{o}_t$ is a one-hot vector depicting a class. When the training is over, class centroids are desired to be found on columns of $\mathbf{W}_1$.

In training, the prior of each class (friend vs foe) will inevitably be non-uniform. In other words, for any $\mathbf{x}_t$, the number of foe centroids will be much higher than the number of the friend centroids. Since the ground truth label for a pair of foe inputs is 0, there is a risk of the system learning to favor 0 outputs. To overcome this problem, we propose the JDML algorithm with prior equalization given in Algorithm 6. To equalize the priors (of friend and foe classes), we force to update one friend and one foe (or equal number of friends and foes) in each epoch (Algorithm 6, lines 7 and 13). This procedure is very useful and can be utilized on other machine learning problems with unequal priors.

4.4.1. Interpretations of JDML

In this section, we provide mathematical and intuitive interpretations and justifications for the proposed JDML recipe.

4.4.1.1. JDML as a Representation Learner. JDML can be considered to be a representation learner for query frames in the initial layer on the query side, and a distance metric learner on the second layer. We solve for the model in Figure-4.8 we use the cross entropy cost function given in (4.10).

$$J_{\text{JDML}}(\mathbf{W}_1, \mathbf{W}_2, b; \mathbf{x}_t, \mathbf{o}_t, r_t) = -r_t \log(f(\mathbf{x}_t, \mathbf{o}_t)) - (1 - r_t)\log(1 - f(\mathbf{x}_t, \mathbf{o}_t)) \quad (4.10)$$

where,

$$f(\mathbf{x}_t, \mathbf{o}_t) = \sigma(\mathbf{x}^T \mathbf{W}_2^T \mathbf{W}_2 \mathbf{W}_1 \mathbf{o} + b).$$

Algorithm 6: JDML with Prior Equalization

1: Separate the dataset into subset of classes:
$Set - class(i) = \{\mathbf{x}_t|, \mathbf{x}_t \in C_i\}_{i=1}^{K}$
2: Initialize network parameters $\mathbf{W}_1$,$\mathbf{W}_2$ and b, set the learning rates μ_1, μ_2, η
3: **repeat**
4: sample a class C_i randomly $1 \leq i \leq K$
5: sample $\mathbf{x}$ from $Set - class(i)$ randomly, set $\mathbf{o} = I(:, i)$
6: Calculate $f(\mathbf{x}, \mathbf{o}) = \sigma(\mathbf{x}^T\mathbf{W}_2^T\mathbf{W}_2\mathbf{W}_1\mathbf{o} + b)$ and the gradients (Figure-4.8)
7: Update network parameters for this pair of friends $\mathbf{W}_1 \leftarrow \mathbf{W}_1 + \mu_1\triangle\mathbf{W}_1$, $\mathbf{W}_2 \leftarrow \mathbf{W}_2 + \mu_2\triangle\mathbf{W}_2$ and $b \leftarrow b + \eta\triangle b$
8: sample a class C_i randomly $1 \leq i \leq K$
9: sample $\mathbf{x}$ from $Set - class(i)$ randomly
10: sample a class C_j $1 \leq j \leq K, j \neq i$, (foes of C_i)
11: set $\mathbf{o} = I(:, j)$
12: Calculate $f(\mathbf{x}, \mathbf{o}) = \sigma(\mathbf{x}^T\mathbf{W}_2^T\mathbf{W}_2\mathbf{W}_1\mathbf{o} + b)$ and the gradients
13: Update network parameters for this pair of foes $\mathbf{W}_1 \leftarrow \mathbf{W}_1 + \mu_1\triangle\mathbf{W}_1$, $\mathbf{W}_2 \leftarrow \mathbf{W}_2 + \mu_2\triangle\mathbf{W}_2$ and $b \leftarrow b + \eta\triangle b$
14: **until** convergence

Figure 4.9. JDML algorithm.

The gradients are calculated following the on-line gradient descent recipe. Since the architecture of JDML is the same as DML after the first layer, the gradients are found by:

$$\triangle b = \frac{dJ}{db} = \frac{dJ}{df}\frac{df}{dz}\frac{dz}{db} \tag{4.11}$$

$$= (\frac{r - f}{f(1 - f)})(f(1 - f))(1) = r - f$$

$$\triangle \mathbf{W}_2 = \frac{dJ}{d\mathbf{W}_2} = \frac{dJ}{df}\frac{df}{dz}\frac{dz}{d\mathbf{W}_2}$$

$$= (\frac{r-f}{f(1-f)})(f(1-f))\frac{d}{d\mathbf{W}_2}(\mathbf{x}^T\mathbf{W}_2^T\mathbf{W}_2\mathbf{W}_1\mathbf{o}) \tag{4.12}$$

$$= (r-f)\mathbf{W}_2(\mathbf{x}\mathbf{o}^T\mathbf{W}_1^T + \mathbf{W}_1\mathbf{o}\mathbf{x}^T)$$

$$\triangle \mathbf{W}_1 = \frac{dJ}{d\mathbf{W}_1} = \frac{dJ}{df}\frac{df}{dz}\frac{dz}{d\mathbf{W}_1}$$

$$= (\frac{r-f}{f(1-f)})(f(1-f))\frac{d}{d\mathbf{W}_1}(\mathbf{x}^T\mathbf{W}_2^T\mathbf{W}_2\mathbf{W}_1\mathbf{o}) \tag{4.13}$$

$$= (r-f)(\mathbf{W}_2^T\mathbf{W}_2\mathbf{x}\mathbf{o}^T)$$

where $z = \mathbf{x}^T\mathbf{W}_2^T\mathbf{W}_2\mathbf{W}_1\mathbf{o} + b$ and $f(\mathbf{x}_t, \mathbf{o}_t)$ is denoted as f for the sake of simplicity.

4.4.1.2. JDML as clustering. We can also consider JDML as the task of finding the centroids of phoneme classes with respect to sigma distance $d_\sigma(\mathbf{x}, \mathbf{y})$ (4.5). Similar to the k-means clustering in Euclidean space, we take the derivative of the cost function and find the optimum. But this time, the distance measure is sigma distance (instead of Euclidean distance) and the cost is total cross entropy (instead of mean squared error (MSE)). If we denote the class centroids as $\mathbf{m}_k$, $k = 1 \cdots K$, and the class labels $r_{t,k} \stackrel{\Delta}{=} \delta(\text{class}(\mathbf{x}_t) = k)$, $k = 1 \cdots K$, $t = 1 \cdots N$.

$$J_{\text{clustering}}(\mathbf{m}_k, r_{t,k}) = \\ -\sum_t \sum_k \Big[r_{t,k} log(f_{t,k}) + (1 - r_{t,k}) log(1 - f_{t,k}) \Big] \tag{4.14}$$

where $f_{t,k} = \sigma(\mathbf{x}_t^T\mathbf{W}^T\mathbf{W}\mathbf{m}_k + b)$

If we take the derivative of J with respect to $\mathbf{m}_k$ and equate to zero, we get

$$\begin{aligned}
\frac{dJ}{d\mathbf{m}_k} = 0 &= -\sum_t (r_{t,k} - f_{t,k})\mathbf{W}^T\mathbf{W}\mathbf{x}_t \\
\sum_t r_{t,k}\mathbf{x}_t &= \sum_t f_{t,k}\mathbf{x}_t \\
\sum_{\forall \mathbf{x}_t \in C_k} \mathbf{x}_t &= \sum_t f_{t,k}\mathbf{x}_t
\end{aligned} \tag{4.15}$$

This result makes sense, in that the optimal centroids are obtained at locations where $f_{t,k} = 1$ if $\mathbf{x}_t$ belongs to class C_k and zero otherwise. Since there is no closed-form solution, we approach with gradient descent. The iteration of $\mathbf{m}_k$ is

$$\begin{aligned}
\mathbf{m}_k &= \mathbf{m}_k - \eta\frac{dJ}{d\mathbf{m}_k} \\
\mathbf{m}_k &= \mathbf{m}_k + \eta(\sum_t (r_{t,k} - f_{t,k})\mathbf{W}^T\mathbf{W}\mathbf{x}_t)
\end{aligned} \tag{4.16}$$

Equation 4.16 shows us that the class centroids $\mathbf{m}_k$ are the weighted and projected sum of all training samples. Weights are decided by the error; essentially, friends are added and foes are subtracted. This approach differs from the Euclidean distance in another aspect; while in Euclidean k-means, only the samples of the pertinent class are averaged, here all samples are taken into account in the calculation.

The update results obtained by the two approaches (representation learning and clustering) yield the same mathematical result. Equations 4.13 and 4.16 denote the same operations since the update on $\mathbf{W}_1$ takes place only on the pertinent column of the matrix (because of the one-hot vector on the outer product). Here the columns of $\mathbf{W}_1$ become the centroids of classes with respect to sigma distance ($\mathbf{m}_k$). One advantage of this approach is that we can update the sigma distance parameters ($\mathbf{W}_2$ and b) and the class centroids ($\mathbf{W}_1$) at the same time.

4.4.2. KWS with JDML

As stated earlier, with JDML we aim to model the query representation for each phoneme, and the distance metric for the corresponding set of phonemes jointly. Hence the query model is obtained by concatenating the centroids learned from JDML (columns of $\mathbf{W}_1$) and in sDTW we use the jointly learned sigma distance parameters ($\mathbf{W}_2$ and b). The flowchart of the final KWS system with JDML is given in Figure 4.10. The experiments on KWS with JDML was published in [22] and in extended form in [23]. The results are presented in Chapter 6 along with other experiments for comparison with other methods of query modeling and distance metrics.

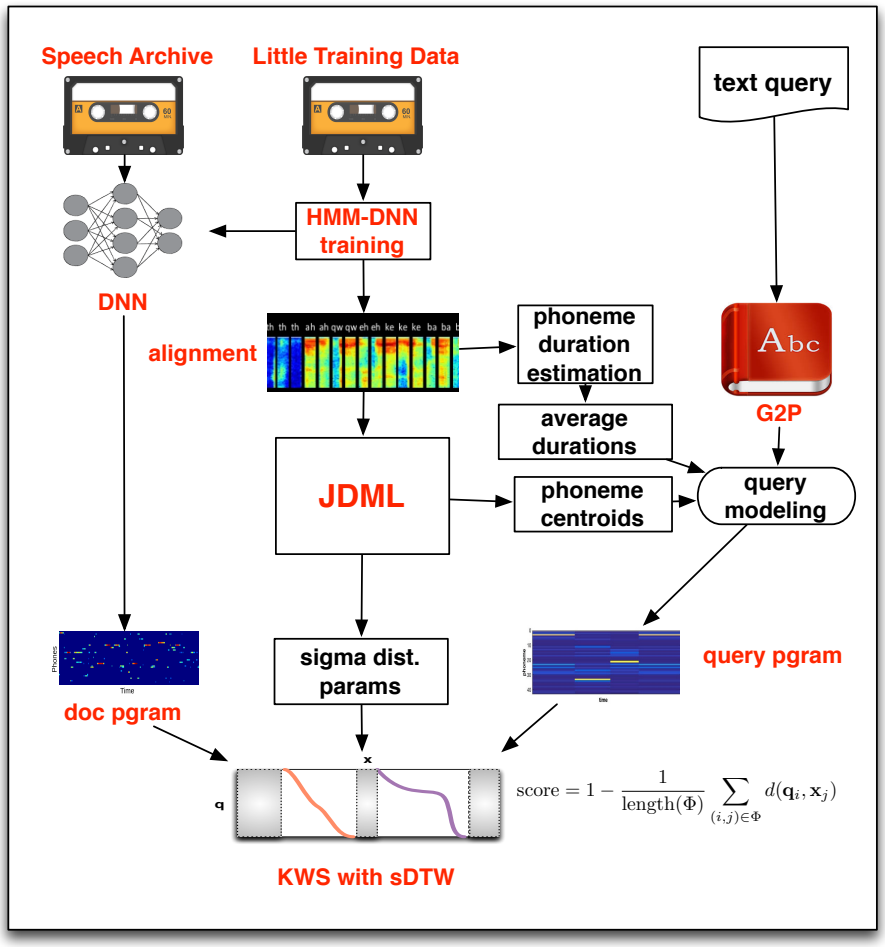

Figure 4.10. sDTW-based KWS flowchart with JDML.

5. NOVEL SCORE NORMALIZATION AND SYSTEM COMBINATION TECHNIQUES

Score normalization and combination for KWS is a very crucial part of the system development. In this chapter, we present the importance of score normalization and introduce the novel score normalization techniques that work hand in hand with the proposed KWS technique. Furthermore, we propose a time efficient system combination schematic and argue that it can be utilized in combination of other heterogeneous detection systems.

5.1. Normalization of Scores for a KWS Task

When a query is given by the user, it is searched within the document and a set of hypotheses is returned each with a relevance score based on the detection methodology of the KWS system. For LVCSR-based KWS systems, these scores are joint log-likelihoods of the keyword obtained from the lattices. In the proposed posteriorgram-based systems, the scores are obtained from the accumulated distances along the alignment path. Then, the system applies a threshold to this set of hypotheses' scores and labels the ones above this threshold as YES.

When another query is entered, the set of hypotheses will inevitably have a different range of scores. Naturally, the score distribution of the KWS system differs significantly for different keywords. For example, longer keywords will tend to have lower scores and keywords having certain phonemes tend to have higher/lower scores compared to others. This is important because we are allowed to use only one threshold value and use it for all keywords to decide YES/NO labels Therefore, scaling of these scores to a similar range is of great importance. This procedure is referred to as normalization.

The histograms of scores obtained by sDTW for 5 sample Turkish keywords are shown in Figure 5.1. These keywords were chosen to have different linguistic behaviors so that the observation of histogram discrepancies and the normalization effects can be made easier. In this set of 5 keywords, we observe the score behavior of short ("emre") vs long ("yatak odası") keywords as well as keywords with different phonemic ingredients, e.g. "ses kaydı" vs "doğru düzgün". We tend to expect that short keywords generally have higher scores and lower variance than the long ones. Furthermore, not only the length, but also the phonemic attributes of the keyword effect the distribution shape. We can see how the score distributions vary for different keywords in Figure 5.1.

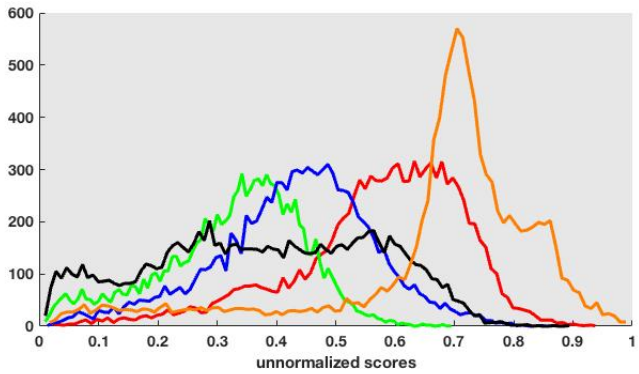

Figure 5.1. Unnormalized score distributions of hypotheses for 5 different keywords over the same speech document.

5.1.1. Keyword Search Based Score Normalization Techniques

There are several score normalization techniques in the literature that aim to rescale the scores to fit in the same range and have the same relative meaning over the set of keywords.

5.1.1.1. Sum-to-One (STO) Normalization. STO normalization [105], which is also widely used in text retrieval systems, aims to approximate the set of hypotheses for each keyword to simulate a probability mass function. It is simply obtained by dividing the score of each hypothesis with the sum of the scores for the corresponding query. If we call the vector of scores for each keyword $\mathbf{s}$, the STO-normalized score vector (of

hypotheses) are obtained by

$$\mathbf{s}_{STO} = \frac{\mathbf{s}}{\sum_i \mathbf{s}(i)} \tag{5.1}$$

5.1.1.2. Keyword Specific Thesholding (KST). One of the most commonly used normalization techniques for KWS is keyword specific thresholding (KST) [106]. In KST normalization, the primary goal is to find the threshold value to maximize the TWV metric given in Equation 2.13. However, the N_{target} parameter of TWV, i.e. the number of target occurrences of a particular keyword is not known to the system developer. An estimate of the number of target occurrences in the document is obtained by summing the system scores, under the assumption that the raw score of a detection is the probability of it being correct, and this estimate is used in the maximization. However, since the posteriorgram based KWS systems can practically return scores for every subsequence of the document, such an estimate, and the KST normalization by extension, is not feasible.

5.1.2. Distribution Based Score Normalization Techniques

Given our sDTW-based KWS scheme, we can practically hypothesize a similarity score for any subsequence of the document. Therefore, in contrast to the LVCSR-based KWS systems, we can make use of the distribution-based score normalization techniques that exist in the literature of other tasks. Furthermore, we propose novel distribution-based score normalization techniques, which yield better results than the existing ones.

5.1.2.1. Histogram Equalization(HE). HE normalization aims to rescale the score vector to have the same relevance range in $[0, 1]$. With this, the most relevant hypothesis (possibly a HIT) would have a score of 1 and the most irrelevant hypothesis would have a score of 0 for all queries. HE is widely used in image enhancement applications and it is expected to help in score normalization for keywords with a narrow dynamic

range of hypotheses scores. HE normalized system scores can be obtained by:

$$\mathbf{s}_{HE} = \frac{\mathbf{s} - \min(\mathbf{s})}{\max(\mathbf{s}) - \min(\mathbf{s})} \tag{5.2}$$

5.1.2.2. Gaussian Normalization(z-norm). z-norm [107] assumes that the scores for each keyword are sums of many independent random variables and hence they approximate to the Gaussian distribution. It can also be seen on the histograms in Figure 5.1 that the distributions are bell-shaped. z-norm aims to normalize them so that each have 0 mean and unity variance

$$\mathbf{s}_z = \frac{\mathbf{s} - \mu(\mathbf{s})}{\sigma(\mathbf{s})} \tag{5.3}$$

5.1.2.3. The Mode Normalization (m-norm). m-norm is proposed in [108] for a QbE-STD task, with the understanding that mean normalization of the z-norm is very susceptible to outliers. Since the hypotheses of interest are actually the outliers (HITs), the normalization is conducted on the mode of the distribution. Furthermore, it has been observed that the distributions are asymmetric and tend to have longer left tails than the right tails. The left tails correspond to irrelevant hypotheses, and the steep right tails correspond to the relevant hypotheses, which are of greater interest. Therefore the standard deviation is calculated among those above the mode of the distribution.

$$\mathbf{s}_m = \frac{\mathbf{s} - \text{mode}(\mathbf{s})}{\sigma(\mathbf{s} > \text{mode}(\mathbf{s}))} \tag{5.4}$$

5.1.3. Novel Score Normalization Techniques for sDTW-based KWS

5.1.3.1. Pruned STO Normalization. It is obvious that if all possible outputs of the system were to be normalized to give sum to one, the normalized scores would become very small, making it impossible to decide a fine-enough well-operating threshold. Hence we need to prune the results for some criterion. Hence, we modify the

STO normalization to have a more keyword-specific behavior by pruning all outputs less than a certain percentage of the highest scoring output of that specific keyword

$$\mathbf{s}_{pSTO}(i) = \begin{cases} 0, & \text{if} \quad \mathbf{s}(i) < p.\max(\mathbf{s}) \\ \dfrac{\mathbf{s}(i)}{\sum\limits_{\mathbf{s}(i) \geq p.\max} \mathbf{s}(i)}, & \text{if} \quad \mathbf{s}(i) \geq p.\max(\mathbf{s}) \end{cases} \tag{5.5}$$

As the global (keyword-independent) parameter p, the vicinity of 0.95% of the best hypothesis for each keyword was found to be the best performing one.

5.1.3.2. Median Normalization (b-norm). Inspired by the claims of m-norm that the score distribution for each keyword is an asymmetric bell-shape, we propose that normalization by zeroing the medians would be more efficient since it is not only less susceptible to outliers, but also fixing the number of hypothesis in the statistics calculation comes in handy. Hence, we propose calculating the standard deviation of the hypotheses greater than the median.

$$\mathbf{s}_b = \frac{\mathbf{s} - \text{median}(\mathbf{s})}{\sigma(\mathbf{s} > \text{median}(\mathbf{s}))} \tag{5.6}$$

5.1.3.3. Alternative Mode Normalization (m2-norm). The distinction that separates z-norm, m-norm and b-norm from the STO and HE normalization techniques is their statistical interpretation. In these distribution-based normalization techniques, a sort of "*typicality*" of the distribution is found and all of those scores lower than this typicality are suppressed. In z-norm, this typicality is taken to be the mean, where in m-norm and b-norm, the typicalities are mode and median, respectively.[5] . Furthermore, the

[5]The term typicality should not be confused with the typical set in information theory. By typicality we mean the statistics that defines a typical member of the distribution

scores higher than the typicality are normalized by the standard deviation calculated over them. It can be argued that in m-norm the standard deviation normalization on "good enough" scores. Therefore we propose that in m2-norm, this "high enough" criterion to be higher than one standard deviation above the mode. The m2-norm normalized scores are calculated with the following formula:

$$\mathbf{s}_{m2} = \frac{\mathbf{s} - \text{mode}(\mathbf{s})}{\sigma(\mathbf{s} > (\text{mode}(\mathbf{s}) + \sigma(\mathbf{s} > \text{mode}(\mathbf{s}))))} \tag{5.7}$$

5.1.3.4. Alternative Median Normalization (b2-norm). Similar to the m2-norm, we apply the normalization using the typicality value as the median. In the next chapter where the experimental results are submitted, it is shown that the alternative normalization techniques (m2-norm and b2-norm) yield better results than the other techniques. b2-norm normalized scores are calculated with the following formula:

$$\mathbf{s}_{b2} = \frac{\mathbf{s} - \text{median}(\mathbf{s})}{\sigma(\mathbf{s} > (\text{median}(\mathbf{s}) + \sigma(\mathbf{s} > \text{median}(\mathbf{s}))))} \tag{5.8}$$

The distributions of the normalized scores upon several normalization techniques are given in Figure 5.2. These proposed novel score normalization techniques are published in [24].

5.2. Fusion of Heterogeneous Systems

Fusion of heterogeneous systems has been studied and used in many detection and verification applications [109, 110]. The main idea is to utilize superiorities of one system to compensate for the impotencies of others in different areas. The fusion can take place at the search part, using different structures and levels, yet generally the fusion is applied to the system scores of different systems. One general fusion methodology is called majority voting. In majority voting, when the majority of the systems have decisions as YES for the same hypothesis, the hypothesis is marked as and the weighted average of their scores is used as a single output of the fusion system.

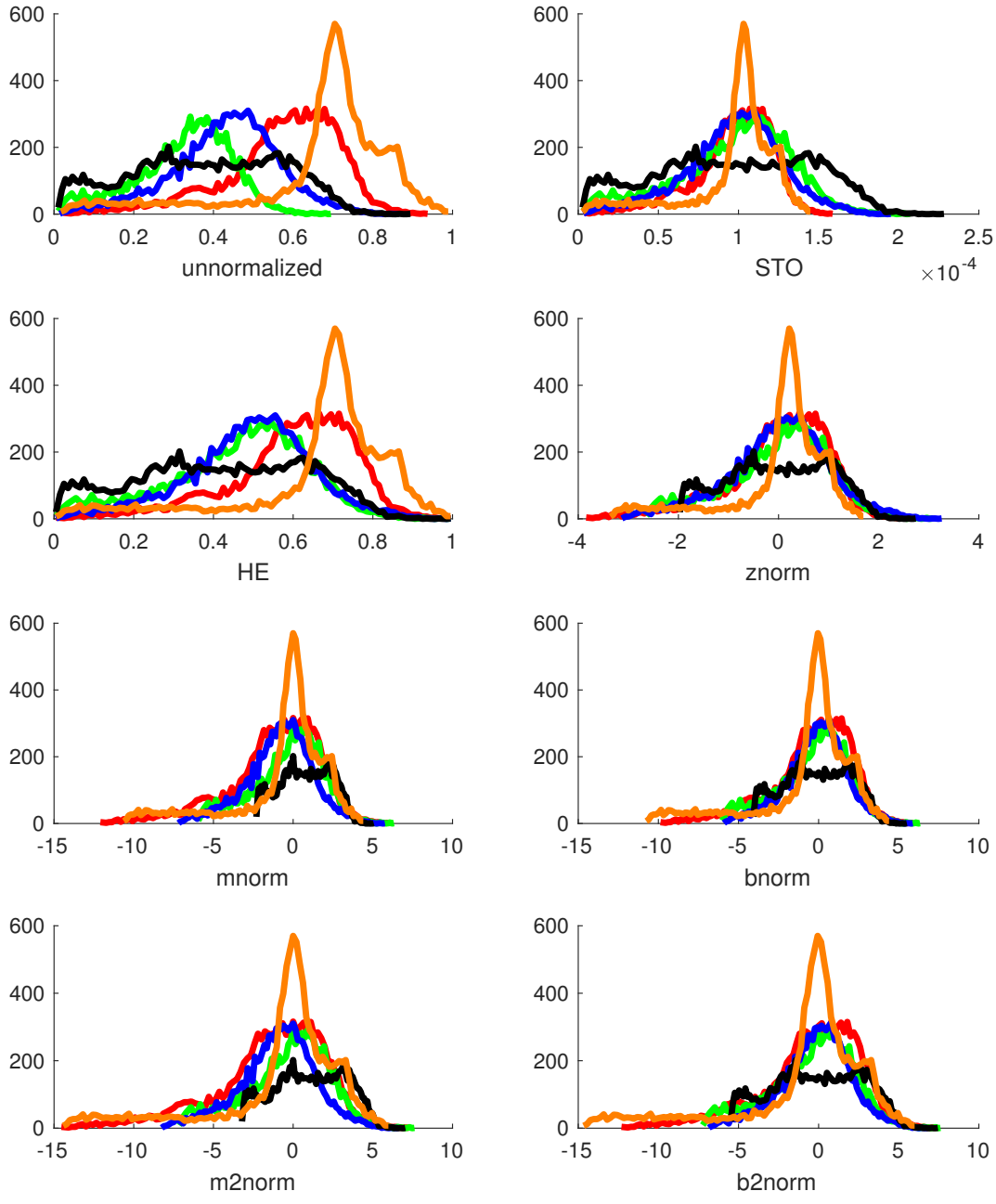

Figure 5.2. Score distributions of 5 different keywords upon several normalization techniques.

This methodology is very useful for elimination of the false alarms.

The current system fusion methodologies implemented for KWS [38] follows a two stage recipe. First, the scores per system and per keyword are STO normalized so that when hypotheses are matched with other systems' hypotheses they would have the same relative meaning. In the second stage, where the merging take place, a series of loops are needed for all systems, for all keywords and for all hypotheses to see the overlapping hypotheses to implement the majority voting. However, this schematic turned out to be inefficient for the combination of LVCSR and sDTW based systems. The reason is that the application of STO normalization on top of the b-norm simply eliminates its efficiency of the b-norm and makes the proposed systems just STO normalized.

In order to utilize the dynamic power of the new normalization techniques, the combination procedure is vectorized and a time efficient algorithm was implemented. In KWS each system output is a list that consists of the locations and scores of each hypotheses per keyword as shown in Equation 5.9 and visualized in Figure 5.3.

$$\begin{gathered} sys(i) = \Big\{ kw\_id(j) : \{file(k), start(k), end(k), score(k)\} \Big\} \\ i = 1 \cdots \#systems, \quad j = 1 \cdots \#keywords, \quad k = 1 \cdots \#hits(i,j) \end{gathered} \tag{5.9}$$

As stated above, the combination procedure follows a series of loops over the hits of the systems and checks to see if there is an overlap and combines the scores of the overlapping hypotheses after STO normalization. In this thesis work, we implemented a time efficient combination scheme in expense of an extended memory usage. Each system $sys(i)$ in Equation 5.9 is now represented with a sparse 3-D matrix $\mathbf{SYS}\{i\}$ filled with the rule given in Equation 5.10.

$$\mathbf{SYS}\{i\} \in \mathcal{R}^{K \mathrm{x} U \mathrm{x} F}, \qquad \mathbf{SYS}\{i\}(j, file(k), start(k) : end(k)) = score(k) \tag{5.10}$$

KW_1				
1	FILE_ID	start	end	score
2	FILE_ID	start	end	score
3	FILE_ID	start	end	score
4	FILE_ID	start	end	score
.	FILE_ID	start	end	score
.	FILE_ID	start	end	score
.				
KW_2				
1	FILE_ID	start	end	score
2	FILE_ID	start	end	score
3	FILE_ID	start	end	score
4	FILE_ID	start	end	score
.	FILE_ID	start	end	score
.	FILE_ID	start	end	score
.				
KW_3				
1	FILE_ID	start	end	score
2	FILE_ID	start	end	score
3	FILE_ID	start	end	score
4	FILE_ID	start	end	score
.	FILE_ID	start	end	score
.	FILE_ID	start	end	score
.				

Figure 5.3. An example output of a KWS system.

where K is the number of keywords, U is the number of discrete utterances (files) we search in, and F is the maximum number of frames in the utterances. The matrix is filled with the score values on the segments the hypotheses suggest, and zeros everywhere else. The filling of this sparse matrix is visualized in Figure 5.4.

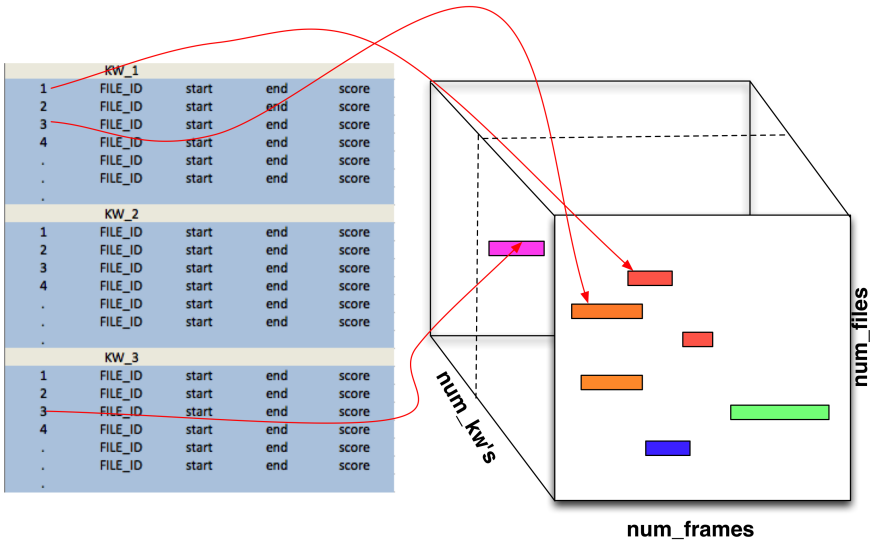

Figure 5.4. Methodology of modeling the system output as a sparse cubic structure.

In this scheme, each slice of the cube on K direction is a hit-list matrix for a specific system and can be added directly with the corresponding matrices of other systems. Before the combination, the KST normalized scores of the LVCSR based

system is multiplied with the maximum obtained score of the proposed systems. Since the KST normalized scores have range $[0, 1]$, a score close to 1 should have the b-norm or m-norm score of a hit for which our systems are very confident. In other words, it is desirable to "trust" the LVCSR based systems for their hypotheses for the in-vocabulary words. Hence, the ranges of the two matrices are equalized by simply multiplying the LVCSR based system with the global maximum of the sDTW based system matrix.

$$\mathbf{SYS}_{norm}\{LVCSR\} = max(max(max(\mathbf{SYS}\{sDTW\}))) \text{ x } \mathbf{SYS}\{LVCSR\} \quad (5.11)$$

Now that the whole scheme is vectorized and equalized, the overlap search is simply done via taking the weighted sum of the sparse system matrices as shown in Equation 5.12. For the experiments of this thesis we took equal weights for the systems. This procedure is visualized in Figure 5.5.

$$\mathbf{Fusion} = \frac{1}{N}\sum_{i=1}^{N}\mathbf{SYS}\{i\} \quad (5.12)$$

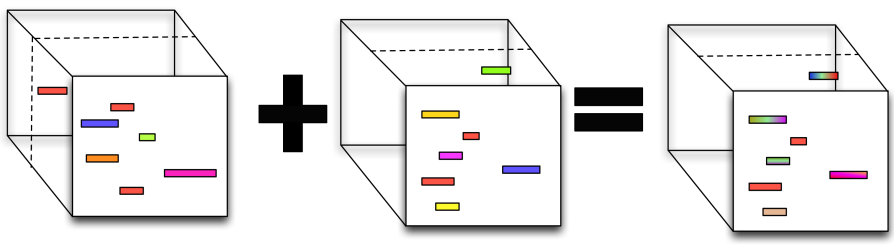

Figure 5.5. Procedure to obtain the fusion matrix cube from the two systems.

The fusion system matrix, which is a sum of sparse matrices, will again be a sparse matrix. The ultimate list of system hypotheses upon the fusion is then extracted from this matrix by finding non-zero segments. Due to averaging, non-overlapping

hypotheses will be suppressed. These also correspond to hypotheses where the minority rooted for. For other hypotheses, with majority of the votes, the overlapping part will have a relatively high score. Due to the fact that different systems will report different start and end locations for the same hypothesis, the scores for those will vary along frames on the combination matrix. The colors of the separated areas in Figure 5.4 correspond to different scores described by Equation 5.10. Hence the fusion matrix on Figure 5.4 may have varying range of score values for each separated hypothesis. During the extraction of the hypotheses list from the matrix, their mean is taken for each separated hypothesis in the sparse cube, so that a measure of the overlap ratio is also integrated in the proposed combination methodology. This procedure is illustrated in Figure 5.6.

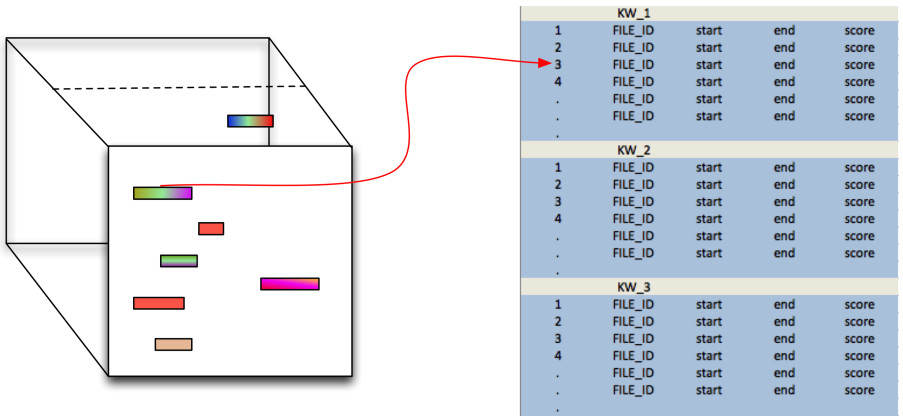

Figure 5.6. The system output obtained from the fusion cube.

6. EXPERIMENTAL RESULTS

So far in this thesis, we explained the mathematics and system definitions of the currently used techniques in the literature, as well as the descriptions of the novel techniques being contributed to the literature. We deliberately deferred the submission of all experimental results to this chapter so that a more efficient comparison between all methodologies would be possible. The proposed posteriorgram-based KWS system was tested on 3 different low resource languages on 2 different data-sets each. We also compared our system performances with the state of the art techniques to show the improvements of our additions. In the experiments, one of the sets for each language is used as development data to determine system parameters and the other set is used to evaluate the system performance to observe robustness.

6.1. Datasets

The experiments were conducted on Turkish [6] , Pashto [7] and Zulu [8] limited language pack (LLP) datasets provided by IARPA Babel Program [12]. A table and segmentation of the datasets are shown on Table 6.1.

The 10-hour transcribed training data is used both for the DNN and the DML/ JDML training. The phoneme-level alignments on the training data, obtained using the Viterbi algorithm, were used in the estimation of phoneme statistics to be used in query modeling. System development experiments for KWS are conducted on the 10-hour development data for each language. We refer to this set as dev-set. After running the search on the dev-set and learning the the KWS parameters (such as optimal decision threshold values, score normalization method, pruning thresholds etc.), we tested our systems on a part of the evaluation data (evalpart1), which was also provided by the Babel Program. We refer to this set as eval-set.

[6] babel105b-v0.4 (dev:kwlist, eval:kwlist2)
[7] babel104b-v0.4bY (dev:kwlist3, eval:kwlist4)
[8] babel206b-v0.1e (dev:kwlist3, eval:kwlist4)

Table 6.1. Dataset descriptions for experiments.

Language	Dataset	Audio-Length	Keyword List		
			IV	OOV	All
Turkish	training	10 hours(transcribed)	-	-	-
	dev-set	5 hours	219	88	307
	eval-set	10 hours	1955	1216	3171
Pashto	training	10 hours(transcribed)	-	-	-
	dev-set	5 hours	1465	600	2065
	eval-set	10 hours	3232	971	4203
Zulu	training	10 hours(transcribed)	-	-	-
	dev-set	5 hours	1194	806	2000
	eval-set	10 hours	2198	1112	3310

6.2. System set-up

6.2.1. The Baseline System

The baseline uses an LVCSR based KWS system with a DNN acoustic model, as described in [111]. From the word lattices generated by the Kaldi speech recognition toolkit [97], a WFST based search is conducted on the keyword indexes. The phonemic transcriptions of the OOV keywords are obtained using the Sequitur grapheme-to-phoneme (G2P) system [99] and the OOV search is conducted using the proxy keyword method [58] using word proxies. The baseline system uses KST normalization of scores [106].

6.2.2. Posteriorgram Based KWS Systems

For development, we tested 6 different posteriorgram-based KWS systems designed using various query models and distance measures. The posteriorgram-based methods denote our novel search methodology with pseudo query modeling and sDTW based search on document posteriorgram. Each query modeling technique and each distance metric can denote a different system. The descriptions of these proposed

systems are given in the list below:

- CB: cosine distance on binary query model,
- CA: cosine distance on average query model,
- LB: log-cosine distance on binary query model,
- LA: log-cosine distance on average query model,
- DML: sigma distance on average query model,
- JDML: simultaneous query modeling and distance metric learning system

Starting from the Turkish dev-set, as we progressed on the system development, we eliminated the poor performing systems and continued with the more fruitful ones. The comparisons of the proposed system against the CB, CA, LA and, LB are useful to observe the efficacy of the proposed distance metric learning methodology, while the comparison of JDML the DML systems depict the contribution of the simultaneous query modeling for the new distance measure.

6.3. Individual System Results

In the individual development of the proposed KWS systems, we conducted the search using all of the systems and observed the MTWV, OTWV and STWV using STO [106] normalization. MTWV shows the systems' ability to discriminate correct hits from false alarms, while OTWV shows the potential for further score normalization and STWV signifies the detection power. The initial results can be seen in Figure 6.1.

First off, we see that all of the sDTW-based systems yield non-zero MTWV, which means such an artificial query modeling and an application of a QbE-STD-like search in KWS task is feasible. We also observe that DML and JDML yield higher performances over all evaluation metrics compared to other systems, using simpler query models and the distance metrics used in QbE-STD. This means that learning a specifically tailored distance metric improves the KWS performance on all metrics. Another notable observation was that OTWVs of the DML based systems are significantly higher than those of other systems. Hence, as the next step, we tested the

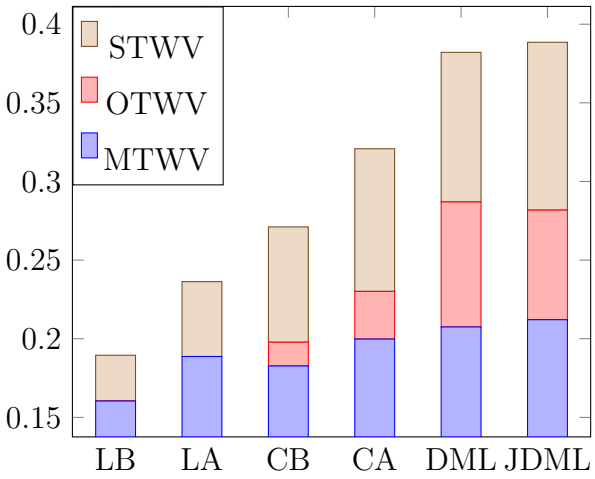

Figure 6.1. Individual system results on the Turkish dev-set with STO.

systems using various normalization techniques and continued the system development with CA, CB, DML and JDML systems.

6.3.1. Tests on Score Normalization Techniques

MTWV metric is calculated on a single best global threshold applied on all of the keywords. Since each keyword has inevitably different acoustic attributes and characteristics, better precision and recall performances are achieved if we use different threshold values for each keyword. However, for system development, this is not allowed. The difference between MTWV (single global threshold value) and OTWV (different threshold values for each keyword) shows the necessity for a normalization of system scores. At this point, to observe the efficiency of different normalization techniques, we applied various normalization techniques, such as histogram equalization [107], z-norm [112], m-norm [108] and the proposed b-norm, m2-norm and b2-norm [24] on the individual system scores. One of the most commonly used normalization techniques for KWS is KST normalization. KST is based on the assumption that the raw score of a detection is the probability of it being correct. An estimate of the number of true occurrences in the document is obtained by summing the system scores. Since the posteriorgram based KWS systems can practically return scores for every subsequence of the document, such an estimate, and the KST normalization by extension, is not feasible. We observe that the proposed JDML system yields consistently better results

than all other systems across the normalization techniques. The MTWVs after various normalizations can be seen in Figure 6.2.

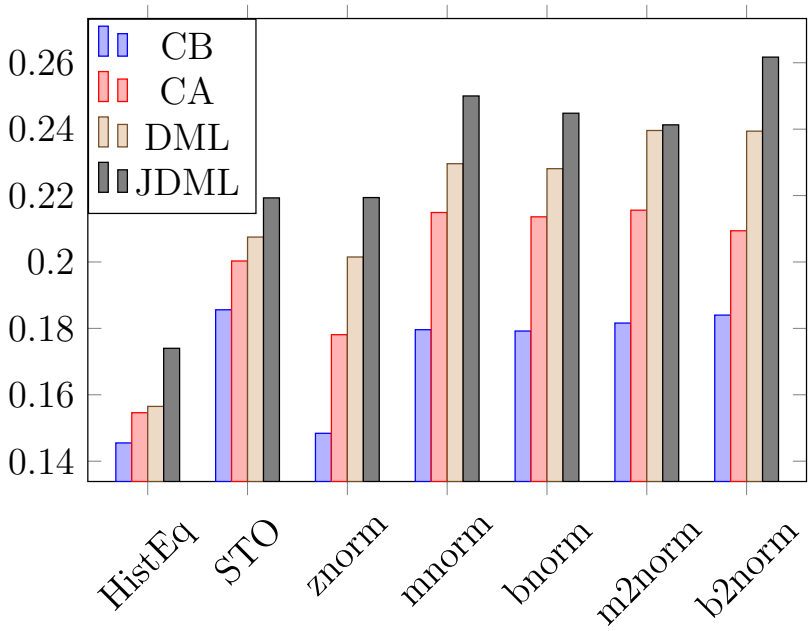

Figure 6.2. MTWV performances of individual systems on Turkish dev-set under several normalizations

6.4. System Fusion Results

As stated earlier, one of the most significant conclusions of the QbE-STD campaign and the Babel program was that the calibration and fusion of heterogeneous systems improve the overall performance. The proposed posteriorgram-based KWS adopts a very different approach to the problem than the conventional LVCSR-based KWS scheme, hence the main intention was to improve the baseline performance in combination. Just like the individual system performances, the fusion of systems performed the best under the b2-norm normalization. As introduced in the literature review, $1 - C_{nxe}^{min}$ is a metric that evaluates the system scores, not only the system decisions. Hence, at this point we, tested our systems' contributions based on not only MTWV, but also the $1-C_{nxe}^{min}$. The combination MTWV and $1-C_{nxe}^{min}$ scores obtained on the Turkish development set show a similar improvement trend as can be seen in Figure 6.3.

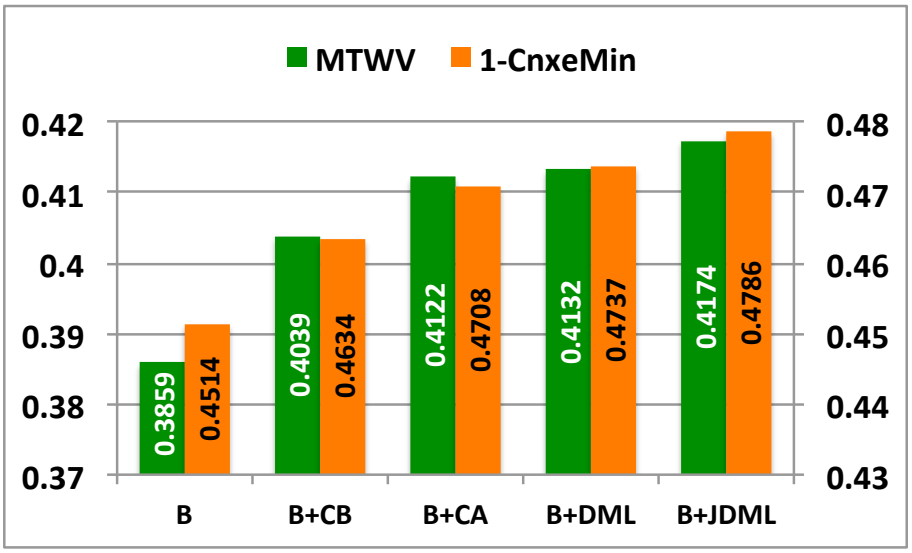

Figure 6.3. MTWV and $1 - C_{nxe}^{min}$ fusion results on Turkish dev-set.

The improvements on MTWV and OTWV for IV and OOV terms separately are shown on Tables 6.2, 6.3 and 6.4 for Turkish, Pashto and Zulu, respectively. Although the improvement on IV terms after fusion is quite modest; on OOV terms, the proposed system outperforms the proxy keyword approach individually and provides significant improvements upon fusion.

Table 6.2. Turkish dev-set system fusion results.

		B	JDML	B+JDML	Gain(%)
MTWV	ALL	0.3859	0.2617	0.4174	8.1
	IV	0.4657	0.2723	0.4871	4.6
	OOV	0.1880	0.2421	0.2538	35.0
OTWV	ALL	0.4169	0.4175	0.5488	31.6
	IV	0.4896	0.4200	0.5899	20.5
	OOV	0.2317	0.4110	0.4440	91.6

The experiments conducted on the eval-set with a different keyword list proved the system's robustness. Using the optimal threshold values that produced the MTWV in the dev-set, we calculated the ATWV on the eval-set. The eval-set performances for each language are shown in Tables 6.5, 6.6 and 6.7, for Turkish, Pashto and Zulu, respectively. ATWV metric can summarize the actual test performance of the proposed

Table 6.3. Pashto dev-set system fusion results.

		B	JDML	B+JDML	Gain(%)
MTWV	ALL	0.2107	0.1101	0.2272	7.8
	IV	0.2527	0.1009	0.2577	2.0
	OOV	0.0842	0.1442	0.1591	89.0
OTWV	ALL	0.3278	0.2305	0.3761	14.7
	IV	0.3774	0.2144	0.3895	3.2
	OOV	0.1786	0.2790	0.3355	87.9

Table 6.4. Zulu dev-set system fusion results.

		B	JDML	B+JDML	Gain(%)
MTWV	ALL	0.2268	0.1816	0.3062	35.0
	IV	0.3174	0.1776	0.3383	6.6
	OOV	0.0915	0.2566	0.2597	183.7
OTWV	ALL	0.3302	0.3409	0.4619	39.9
	IV	0.4622	0.3380	0.4746	2.7
	OOV	0.1333	0.3780	0.4428	232.2

technique. The improvement over all terms on ATWV metric is 13.9% relative and, more significantly, the ATWV improvement for the OOV terms reaches 154.5%.

6.4.1. OOV Performance of the Proposed System

The proposed posteriorgram-based KWS systems aim to assist the LVCSR-based systems, especially on the OOV terms. The experiments showed that not only did it improve the state of the art proxy keyword-based baseline ATWV by 154.5% relative upon combination, but it also outperformed the baseline as a standalone system on all experiments and all metrics for OOV terms. In Figure 6.4, we plot the baseline and the proposed system's MTWV and ATWV numbers obtained from the three languages' dev-set and eval-set experiments as markers on a scatter plot. Each point on this scatter

Table 6.5. Turkish eval-set system fusion results.

		B	JDML	B+JDML	Gain(%)
ATWV	ALL	0.2848	0.1586	0.3219	13.0
	IV	0.3595	0.1600	0.3791	5.4
	OOV	0.0955	0.1551	0.1770	85.4
MTWV	ALL	0.2861	0.1646	0.3286	14.8
	IV	0.3725	0.1693	0.3862	3.7
	OOV	0.0955	0.1608	0.1881	97.0
OTWV	ALL	0.4126	0.3266	0.4997	21.1
	IV	0.4955	0.3225	0.5422	9.4
	OOV	0.2028	0.3370	0.3919	93.3

plot is an experiment (i.e. Pashto dev-set, Zulu eval-set etc.) and its corresponding IV vs OOV ATWV numbers. When the OOV performances are considered, we can observe the proposed system's superiority to the baseline (looking at the locations of the markers on projection to the y-axis). We also fit lines for both methods on the scatter plot and see that the line for the JDML system has a slope that almost equals to 1, where the baseline is far from that. We claim that a desired performance would be independent of the vocabulary and such a system should have similar IV and OOV performances. The proposed system yields such a performance given the experimental results.

6.4.2. Comparison with Similar Work

As expected, LVCSR-based methods perform better than the proposed approach on IV words due to the inclusion of language model. As stated earlier, our methodologies aimed primarily to help OOV retrieval. When we compare the proposed system's performance, with the other OOV retrieval techniques in literature, we see that the proposed method provides significant contributions due to its originality. In our survey of the existing approaches, we found the work on PPMs [61] to be the most comparable

Table 6.6. Pashto eval-set system fusion results.

		B	JDML	B+JDML	Gain(%)
ATWV	ALL	0.2458	0.1148	0.2588	5.3
	IV	0.2775	0.1126	0.2791	0.6
	OOV	0.0473	0.1289	0.1315	177.9
MTWV	ALL	0.2459	0.1168	0.2597	5.6
	IV	0.2775	0.1143	0.2846	2.6
	OOV	0.0629	0.1353	0.1325	110.5
OTWV	ALL	0.3857	0.2544	0.4229	9.6
	IV	0.4220	0.2499	0.4408	4.5
	OOV	0.1581	0.2824	0.3102	96.2

to ours, not only because their approach does not make use of LVCSR systems, like us, but also because the authors worked with the same baseline Kaldi DNN architecture and OOV-handling method that we did. The authors were able to obtain a significant 50.5% average relative ATWV improvement on OOV terms over the languages they worked on, upon fusion with the LVCSR-based baseline. On the Zulu dev-set experiment, for which they reported a 111% ($0.09 \rightarrow 0.19$) relative gain on the baseline, we were able to obtain a 183.7% improvement on MTWV. It should be noted that the authors reported the ATWV scores based on a KWS system whose hyper-parameters are learned from a subset of the dev-set. Table 6.8 presents a thorough comparison with the other OOV handling methodologies reported results on the matching languages. On this table, the work labeled as "proxy keyword" [58] is our baseline which use acoustically similar IV terms, for searching OOV terms. The "confusion modeling" work [59] follows a similar method with different level of proxy usage. In the work denoted "web crawling" [56] aims to search web for automatically obtained and filtered words to extend the language model, so that the OOV terms may exist on the lattice. [9]

[9]Disclaimer: Since neither the version of Kaldi nor the keyword list used in the aforementioned work is provided, it was impossible to perform experiments under identical conditions.

Table 6.7. Zulu eval-set system fusion results.

		B	JDML	B+JDML	Gain(%)
ATWV	ALL	0.2459	0.1648	0.3033	23.3
	IV	0.3132	0.1631	0.3452	10.2
	OOV	0.0630	0.1693	0.1892	200.3
MTWV	ALL	0.2462	0.1695	0.3056	24.1
	IV	0.3132	0.1702	0.3458	10.4
	OOV	0.0651	0.1775	0.2056	216.0
OTWV	ALL	0.3106	0.3328	0.4608	48.4
	IV	0.3800	0.3259	0.4898	28.9
	OOV	0.1219	0.3516	0.3820	213.4

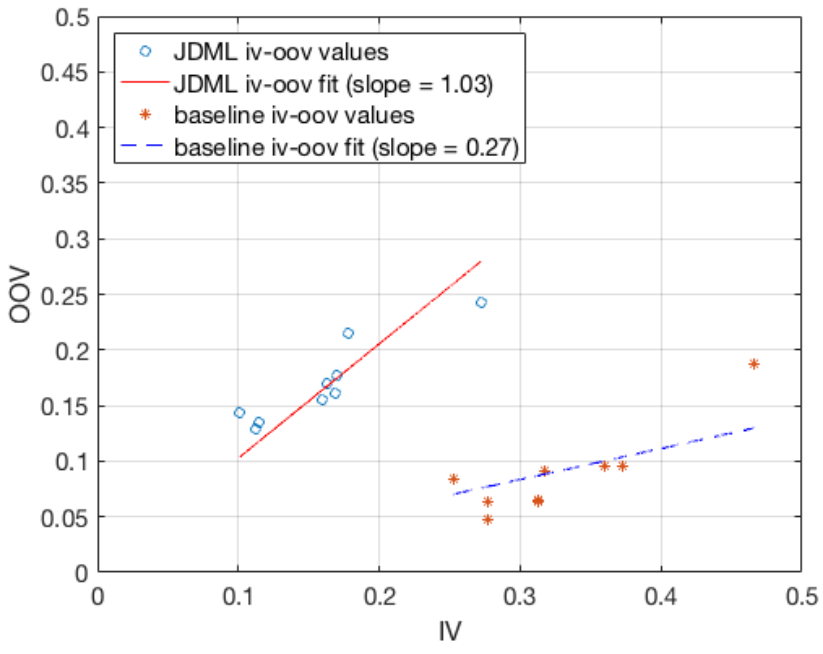

Figure 6.4. The IV vs OOV performance comparison of the baseline system and the proposed system.

Table 6.8. OOV performance comparison with similar systems in literature.

Method	Turkish	Pashto	Zulu
confusion modeling [59]	0.115	0.033	
proxy keyword [58]	0.188	0.084	0.091
web crawling [56]	0.210	0.060	-
PPM [61]	-	-	0.140
PPM+proxy [61]	-	-	0.190
JDML	0.2421	0.1442	0.2566
JDML+proxy	0.2538	0.1591	0.2597

7. FURTHER WORK: SEQUENCE MODELING FOR POSTERIORGRAM BASED KEYWORD SEARCH

As a further study, we investigated the usage of recurrent neural networks (RNN) to the proposed approach in two different scenarios:

- Usage of RNNs to incorporate phoneme level language information to enhance the search document posteriorgram
- Usage of RNNs to model simultaneous state and phone confusion information in query "generation"

7.1. RNNs for Document Posteriorgram Enhancement

The schematics of our proposed approach does not make use of any language model information neither in the feature extraction (posteriorgrams and query modeling) nor in the search. This can be considered as a strength of the proposed approach in terms of language independence and language-robustness. During the posteriorgram generation, the frame level cross entropy cost is minimized using the HMM-DNN model. However, we investigated further enhancing the document posteriorgrams using RNNs so that the previous frame activations are also used to estimate the current frame. This is the first time we incorporate temporal information into our modeling.

For this, we used a 2-layer LSTM to estimate a prediction of the current frame using the short term dependencies. The labels are used from the forced alignments to make the posteriorgram more sparse. The enhancement network can be seen in Figure 7.1. Each LSTM cell in the model was implemented as follows:

$$
\begin{aligned}
\mathbf{f}_t &= \sigma(\mathbf{W}_f[\mathbf{c}_{t-1}, \mathbf{h}_{t-1}, \mathbf{x}_t] + \mathbf{b}_f) \\
\mathbf{i}_t &= \sigma(\mathbf{W}_i[\mathbf{c}_{t-1}, \mathbf{h}_{t-1}, \mathbf{x}_t] + \mathbf{b}_i) \\
\mathbf{o}_t &= \sigma(\mathbf{W}_o[\mathbf{c}_t, \mathbf{h}_{t-1}, \mathbf{x}_t] + \mathbf{b}_o) \\
\mathbf{c}_t &= \mathbf{f}_t \star \mathbf{c}_{t-1} + \mathbf{i}_t \star tanh(\mathbf{W}_c[\mathbf{h}_{t-1}, \mathbf{x}_t] + \mathbf{b}_c) \\
\mathbf{h}_t &= \mathbf{o}_t \star tanh(\mathbf{c}_t)
\end{aligned}
\tag{7.1}
$$

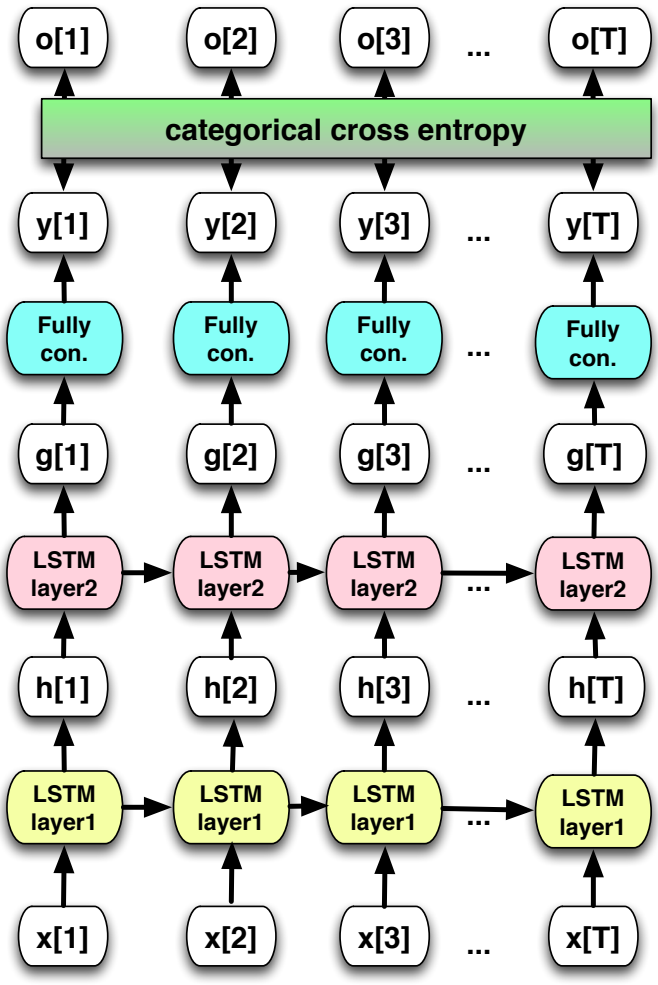

Figure 7.1. Document posteriorgram enhancement with LSTMs. Arrows denote functional dependency whereas the colors denote shared parameters.

The outputs are calculated on a fully connected layer with soft-max activations to estimate the alignment label. The objective function was categorical cross entropy

between the LSTM output and the one-hot vectors from the Viterbi alignment. In order to decide when to stop training, we separated 10% of the training set as validation and observed the convergence. The drop in the cost for the two sets can be seen in Figure 7.2.

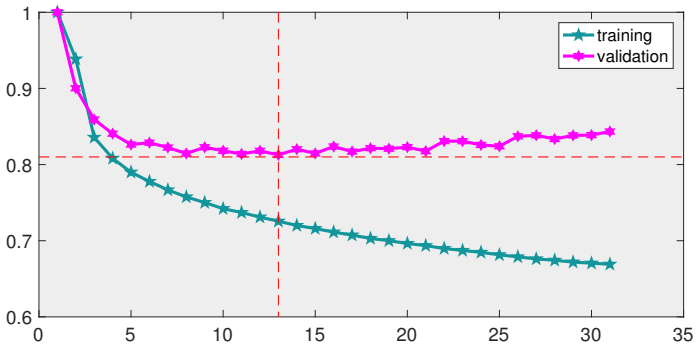

Figure 7.2. Cost behavior on LSTM training.

After the network has trained until convergence, the search document posteriorgram was input to the model in chunks of constant length for which the LSTM was trained to look back. The outputs of the LSTM network was recombined to generate the somewhat "sparsified" document posteriorgram.

To operate hand-in-hand with the new training procedure, we used the categorical cross entropy measure as the distance metric in the sDTW based search. The queries were generated using the binary query model. On the scores obtained from the accumulated distance paths, b2-normalization is applied. The flowchart of the KWS system with the document enhancement can be seen in Figure 7.3.

7.2. RNNs for Query Generation

In the second experiment, we investigated the usage of RNNs for query generation. For this, a single layer RNN is trained, following the model:

Given an input sequence $\mathbf{O} = \{\mathbf{o}_1, \cdots, \mathbf{o}_T\}$, which are the one-hot vectors corresponding to the phonetic indexes of the keyword, the hidden vector sequence $\mathbf{H} =$

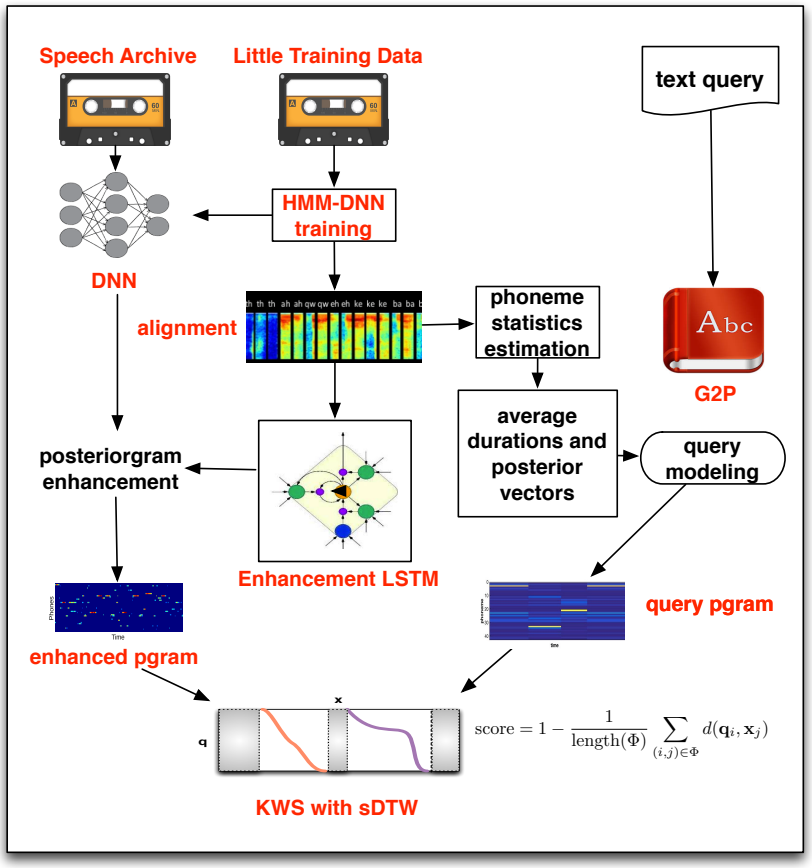

Figure 7.3. Flowchart of the KWS system utilizing LSTM-based document posteriorgram enhancement.

$\{\mathbf{h}_1, \cdots, \mathbf{h}_T\}$, and output vector sequence $\mathbf{Y} = \{\mathbf{y}_1, \cdots, \mathbf{y}_T\}$ are calculated by iterating the following equations from $t = 1 \rightarrow T$:

$$\mathbf{h}_t = \mathbf{W}_{oh}\mathbf{o}_t + \mathbf{W}_{hh}\mathbf{h}_{t-1}, \qquad \text{and} \qquad \mathbf{o}_t = \text{softmax}(\mathbf{h}_t) \tag{7.2}$$

With this, we aim to model two things while generating the query:

- $\mathbf{W}_{oh}$ models the between phone confusions that the decoder has
- $\mathbf{W}_{hh}$ models the state and phone transition confusions

After training the generator model until convergence we use the concatenated one-hot vectors corresponding to the phoneme indexes for each query to generate artificial query posteriorgrams. This way we also end up modeling the phoneme variations inside

the durations of the phonemes. Also, we model the inter-phoneme confusions, which we addressed by the average query model. The KWS flowchart utilizing the query generation using the generative RNN is shown in Figure 7.5.

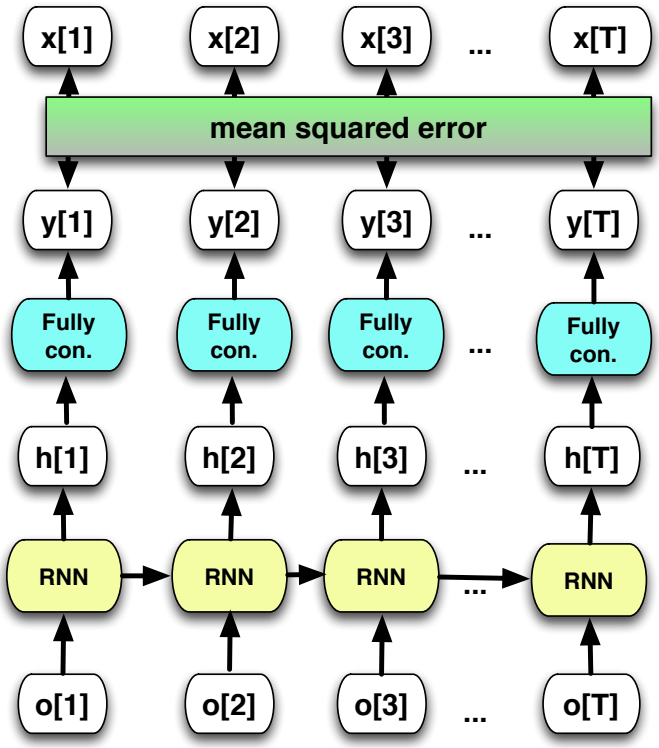

Figure 7.4. RNN-based query generation methodology.

The experimental results on Turkish dev-set can be seen on Table 7.1. On the table, the experiments were annotated as follows:

- JDML: distance metric and query model representations are jointly learned and the parameters are used in search, as stated in this thesis
- SP: sparsified document posteriorgram, binary query model and cross entropy distance measure in search
- GR: queries are obtained by the generative RNN and cross entropy distance measure in search.

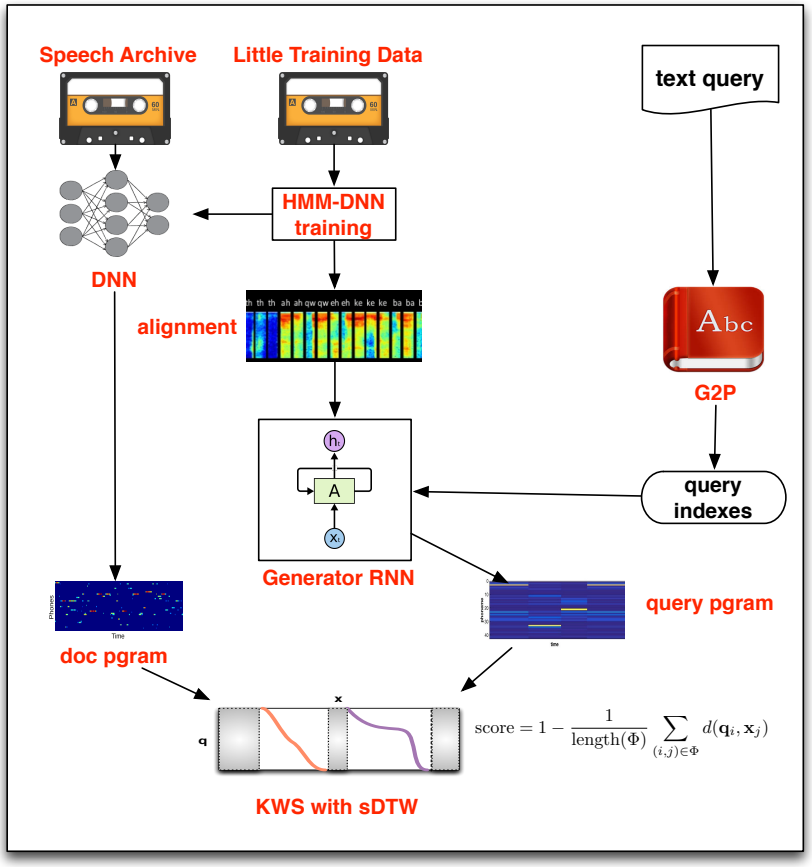

Figure 7.5. Inclusion of generator RNN in posteriorgram based KWS.

Table 7.1. Turkish dev-set posteriorgram sparsifier and query generator results.

		JDML	SP	GR
MTWV	ALL	0.2617	0.2741	0.2652
	IV	0.2723	0.2738	0.2901
	OOV	0.2421	0.2854	0.2076
OTWV	ALL	0.4175	0.4157	0.4194
	IV	0.4200	0.4140	0.4320
	OOV	0.4110	0.4199	0.3871
STWV	ALL	0.8152	0.7422	0.7214
	IV	0.8332	0.7536	0.7389
	OOV	0.7695	0.7132	0.6769

8. CONCLUSIONS

In this thesis, we addressed the problems of keyword search, caused by the scarcity of sources in low resource languages. We mentioned that, for these languages, reliable LVCSR systems cannot be built, and therefore, any KWS system that operates using the outputs of LVCSR systems, will inherently fail to retrieve terms of interest with high accuracy. We also pointed out that many of the query terms will in practice be out of the vocabulary of LVCSR systems, trained with low resources. We claimed that a methodology of a KWS system, which does not use LVCSR systems, should be studied to help retrieval of OOV terms. In search for solutions for these problems, we utilized the techniques of a pertinent task QbE-STD. In QbE-STD, or in any QbE task, the query is provided in the same format with the search document. A template matching-based search is conducted to retrieve the similar parts of the document.

In order to extend the successful techniques of QbE-STD to KWS, we modeled the text query as sequences of feature vectors, to match the format of the document. Due to its speaker independence characteristic, we used phoneme-level posteriorgrams to model the document, and artificially modeled the text query with phoneme posterior averages corresponding to the phonemic transcription of the keyword. We have seen that, the new scheme of pseudo query modeling and template matching based search actually works and helps the LVCSR based systems in combination, especially for the OOV terms. This methodology is desirable since no usage of language model is involved and it treats the IV terms and OOV terms with comparable performances.

We also investigate learning a more efficient frame-level distance metric to be used in the sDTW-based search, using a part of the training data. We have proposed a novel Siamese neural network-based distance metric learning methodology that would both model the decoder confusions between similar phonemes and provide a more desirable behavior on the DTW flow. We have seen that, not only did the new distance metric discriminate samples of different classes better than other known metrics, but it also worked better in sDTW-based KWS.

Upon learning a better distance metric to work in the posteriorgram representation space we work, we investigated learning proper query model frame representations that would work hand in hand with the new distance metric, instead of using just first order statistics we obtain for each phoneme. We extended the DML network, to simultaneously learn the new distance metric parameters and the proper frame representations for the artificially modeled queries. We have justified the new model using two approaches : 1. representation learning, 2. clustering. We showed that, approaching the JDML methodology as a representation learning for the top layers and clustering for a given distance metric yields the same mathematical results. We showed that, the usage of jointly learned representations and the distance metric works better than our initial proposed system-DML, which was already, again in this thesis, shown to be better than other distance metrics.

Given the same initial resources as the conventional LVCSR-based systems, a search document, a limited training corpus and a list of keywords, we built a complete KWS system, using a methodology of some other task, i.e. QbE-STD. We were able to return hypotheses of similarity for every subsequence of the document given a query. In order for these hypotheses to be labeled as relevant or irrelevant based on a single threshold, a normalization of the scores for hypotheses are needed. The contemporary score normalization techniques for the LVCSR-based KWS systems proved inefficient for our new scheme. For this reason, as a complementary measure, we worked on novel normalization techniques for our new system. We see that, the proposed novel normalizations performs better, making MTWVs closer to OTWVs, than other techniques. These normalization methods are not only suitable for the proposed sDTW-based KWS system, but they may also be used in other verification and detection systems.

Due to the originality and novelty of the search technique proposed in this thesis, it is expected to provide improvements in retrieval, when the results are combined with the other systems in literature. To be able to harvest the efficiency we obtained with the new normalization techniques, we prosed a new system fusion methodology that would take into account the overlap ratios as well. Furthermore, this new system fusion method combines multiple systems without the need for extra loops over new systems

and new hypotheses, in the cost of some expended memory usage.

The experiments conducted on the three low resource languages, Turkish, Pashto and Zulu, using two different datasets showed that the proposed methodologies fulfill the goals of set for this thesis and provide significant improvements on two different evaluation metric. Experiments on the eval-sets for all languages proved the systems' robustness to datasets. On the evaluation datasets alone, the improvement over all terms on ATWV metric is 13.9% relative and, more significantly, the ATWV improvement for the OOV terms reaches 154.5%. When we compare the proposed systems' results with respect to IV-OOV specific performance, we observed that the proposed system almost always doubles the performance of the LVCSR+proxy based system on OOV terms. We also observe that the proposed system can be called independent of the vocabulary and has similar IV and OOV performances.

REFERENCES

1. Borges, J. L., *The Library of Babel*, Collected Fictions (Penguin), 1999.

2. Haidt, J., *The Happiness Hypothesis: Finding Modern Truth in Ancient Wisdom*, Basic Books, 2006.

3. "VistaWide", `https://www.vistawide.com/languages/language-statistics.htm`, accessed at December 2017.

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

5. Huang, X., A. Acero, H.-W. Hon and R. Reddy, *Spoken language processing: A guide to theory, algorithm, and system development*, Vol. 95, Prentice hall PTR Upper Saddle River, 2001.

6. "GMR Transcription", `https://www.gmrtranscription.com/prices.aspx`, accessed at December 2017.

7. "OpenKWS14 Keyword Search Evaluation Plan", `http://www.nist.gov/itl/iad/mig/upload/KWS14-evalplan-v11.pdf`, accessed at December 2017.

8. Rodriguez-Fuentes, L. J. and M. Penagarikano, *MediaEval 2013 Spoken Web Search Task: System Performance Measures*, Tech. Rep. TR-2013-1, Department of Electricity and Electronics, University of the Basque Country, 2013.

9. "NIST 2017 (Pilot) Speech Analytic Technologies Evaluation", `https://www.nist.gov/itl/iad/mig/nist-2017-pilot-speech-analytic-technologies-evaluation`, accessed at December 2017.

10. Versteegh, M., R. Thiolliere, T. Schatz, X.-N. Cao, X. Anguera, A. Jansen and E. Dupoux, "The zero resource speech challenge 2015.", *Interspeech*, pp. 3169–3173, 2015.

11. Jansen, A., E. Dupoux, S. Goldwater, M. Johnson, S. Khudanpur, K. Church, N. Feldman, H. Hermansky, F. Metze, R. Rose *et al.*, "A summary of the 2012 JHU CLSP workshop on zero resource speech technologies and models of early language acquisition", *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2013.

12. Harper, M., "IARPA Babel program", `https://www. iarpa. gov/index. php/research-programs/babel`, 2014, accessed at December 2017.

13. "USC Shoah Foundation", `http://sfi.usc.edu/about/institute`, accessed at December 2017.

14. Bazzi, I., *Modelling out-of-vocabulary words for robust speech recognition*, Ph.D. Thesis, Massachusetts Institute of Technology, 2002.

15. Szöke, I., *Hybrid word-subword spoken term detection*, Ph.D. Thesis, Faculty of Information Technology, BUT, 2010.

16. Chen, G., S. Khudanpur, D. Povey, J. Trmal, D. Yarowsky and O. Yilmaz, "Quantifying the value of pronunciation lexicons for keyword search in lowresource languages", *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 8560–8564, 2013.

17. Sarı, L., B. Gündoğdu and M. Saraçlar, "Posteriorgram based approaches in keyword search", *Signal Processing and Communications Applications Conference (SIU), 2015 23th*, pp. 1183–1186, IEEE, 2015.

18. Sarı, L., B. Gündoğdu and M. Saraçlar, "Fusion of LVCSR and Posteriorgram Based Keyword Search", *Interspeech*, pp. 824–828, 2015.

19. Gündoğdu, B., L. Sarı, G. Çetinkaya and M. Saraçlar, "Template-based Keyword Search with pseudo posteriorgrams", *2016 IEEE 24th Signal Processing and Communication Application Conference (SIU)*, pp. 973–976, 2016.

20. Gündoğdu, B., L. Sarı, G. Çetinkaya and M. Saraçlar, "Best Student Presentation Award", *2016 24th Signal Processing and Communications Applications Conference (SIU)*, 2016.

21. Gündoğdu, B. and M. Saraçlar, "Distance Metric Learning for Posteriorgram Based Keyword Search", *The 42nd IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) 5-9 March 2017, New Orleans, USA*, pp. 5660–5664, 2017.

22. Gündoğdu, B. and M. Saraçlar, "Similarity Learning Based Query Modeling for Keyword Search", *Interspeech*, pp. 3617–3621, 2017.

23. Gündoğdu, B., B. Yusuf and M. Saraçlar, "Joint Learning of Distance Metric and Query Model for Posteriorgram-Based Keyword Search", *IEEE Journal of Selected Topics in Signal Processing*, Vol. 11, No. 8, pp. 1318–1328, 2017.

24. Gündoğdu, B. and M. Saraçlar, "Novel score normalization techniques for posteriorgram based keyword search", *2017 IEEE 25th Signal Processing and Communication Application Conference (SIU)*, pp. 1–4, 2017.

25. Gündoğdu, B. and M. Saraçlar, "Similarity Measure Optimization for Target Detection: A Case Study for Detection of Keywords in Telephone Conversations", *IGI Global (under review)*, 2018.

26. Hirschberg, J., M. Bacchiani, D. Hindle, P. Isenhour, A. Rosenberg, L. Stark, L. Stead, S. Whittaker and G. Zamchick, "SCANMail: Browsing and searching speech data by content", *Seventh European Conference on Speech Communication and Technology*, 2001.

27. Kubala, F., S. Colbath, D. Liu and J. Makhoul, "Rough'n'Ready: a meeting recorder and browser", *ACM Computing Surveys (CSUR)*, Vol. 31, No. 2es, p. 7, 1999.

28. Siegler, M. A., U. Jain, B. Raj and R. M. Stern, "Automatic segmentation, classification and clustering of broadcast news audio", *Proc. DARPA speech recognition workshop*, 1997.

29. 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", *IEEE Transactions on multimedia*, Vol. 4, No. 1, pp. 88–96, 2002.

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

31. Weintraub, M., "LVCSR log-likelihood ratio scoring for keyword spotting", *IEEE International Conference on Acoustics, Speech, and Signal Processing(ICASSP)*, Vol. 1, pp. 297–300, 1995.

32. Frinken, V., A. Fischer, R. Manmatha and H. Bunke, "A novel word spotting method based on recurrent neural networks", *IEEE Transactions on pattern analysis and machine intelligence*, Vol. 34, No. 2, pp. 211–224, 2012.

33. Weintraub, M., "Keyword-spotting using SRI's DECIPHER large-vocabulary speech-recognition system", *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pp. 463–466, 1993.

34. Chen, G., C. Parada and G. Heigold, "Small-footprint keyword spotting using deep neural networks", *Acoustics, Speech and Signal Processing (ICASSP), 2014 IEEE International Conference on*, pp. 4087–4091, IEEE, 2014.

35. Kupiec, J., D. Kimber and V. Balasubramanian, "Speech-based retrieval using se-

mantic co-occurrence filtering", *Proceedings of the workshop on Human Language Technology*, pp. 373–377, Association for Computational Linguistics, 1994.

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

37. Voorhees, E. M. *et al.*, "The TREC-8 Question Answering Track Report.", *Trec*, Vol. 99, pp. 77–82, 1999.

38. Fiscus, J. G., J. Ajot, J. S. Garofolo and G. Doddingtion, "Results of the 2006 spoken term detection evaluation", *Proceedings of ACM SIGIR Workshop on Searching Spontaneous Conversational*, pp. 51–55, 2007.

39. 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.

40. Can, D. and M. Saraçlar, "Lattice indexing for spoken term detection", *IEEE Transactions on Audio, Speech, and Language Processing*, Vol. 19, No. 8, pp. 2338–2347, 2011.

41. Saraçlar, M. and R. Sproat, "Lattice-based search for spoken utterance retrieval", *HLT-NAACL 2004: Main Proceedings*, Vol. 51, pp. 129–136, 2004.

42. Wang, D., J. Frankel, J. Tejedor and S. King, "A comparison of phone and grapheme-based spoken term detection", *IEEE International Conference on Acoustics, Speech and Signal Processing, (ICASSP)*, pp. 4969–4972, 2008.

43. Garcia, A. and H. Gish, "Keyword spotting of arbitrary words using minimal speech resources", *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Vol. 1, pp. 949–952, 2006.

44. Lee, S.-w., K. Tanaka and Y. Itoh, "Combining multiple subword representations for open-vocabulary spoken document retrieval", *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Vol. 1, pp. 505–508, 2005.

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

46. 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.

47. Parlak, S. and M. Saraçlar, "Spoken term detection for Turkish broadcast news", *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASP)*, pp. 5244–5247, IEEE, 2008.

48. Chia, T. K., K. C. Sim, H. Li and H. T. Ng, "A lattice-based approach to query-by-example spoken document retrieval", *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, pp. 363–370, ACM, 2008.

49. Parlak, S. and M. Saraçlar, "Spoken Information Retrieval for Turkish Broadcast News", *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*, pp. 782–783, 2009.

50. Chelba, C., T. J. Hazen and M. Saraçlar, "Retrieval and browsing of spoken content", *IEEE Signal Processing Magazine*, 2007.

51. He, Y., P. Baumann, H. Fang, B. Hutchinson, A. Jaech, M. Ostendorf, E. Fosler-Lussier and J. Pierrehumbert, "Using pronunciation-based morphological subword units to improve OOV handling in keyword search", *IEEE/ACM Transactions on Audio, Speech and Language Processing (TASLP)*, Vol. 24, No. 1, pp. 79–92, 2016.

52. Mamou, J., B. Ramabhadran and O. Siohan, "Vocabulary independent spoken term detection", *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, pp. 615–622, ACM, 2007.

53. Karakos, D. and R. Schwartz, "Subword and phonetic search for detecting out-of-vocabulary keywords", *Interspeech*, pp. 2469–2473, 2014.

54. Parlak, S. and M. Saraçlar, "Performance analysis and improvement of Turkish broadcast news retrieval", *IEEE Transactions on Audio, Speech, and Language Processing*, Vol. 20, No. 3, pp. 731–741, 2012.

55. Burget, L., "Hybrid word-subword decoding for spoken term detection", *Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 42–48, 2008.

56. Gandhe, A., L. Qin, F. Metze, A. Rudnicky, I. Lane and M. Eck, "Using web text to improve keyword spotting in speech", *IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*, pp. 428–433, 2013.

57. Mendels, G., E. Cooper, V. Soto, J. Hirschberg, M. J. Gales, K. M. Knill, A. Ragni and H. Wang, "Improving speech recognition and keyword search for low resource languages using web data.", *Interspeech*, pp. 829–833, 2015.

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

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

60. Chen, G. *et al.*, *Low Resource High Accuracy Keyword Spotting*, Ph.D. Thesis, Johns Hopkins University, 2016.

61. Liu, C., A. Jansen, G. Chen, K. Kintzley, J. Trmal and S. Khudanpur, "Low-resource open vocabulary keyword search using point process models", *Interspeech*, pp. 2789–2793, 2014.

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

63. Cui, J., B. Kingsbury, B. Ramabhadran, G. Saon, T. Sercu, K. Audhkhasi, A. Sethy, M. Nußbaum-Thom and A. Rosenberg, "Knowledge distillation across ensembles of multilingual models for low-resource languages", *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2017, New Orleans, LA, USA, March 5-9, 2017*, pp. 4825–4829, 2017.

64. Cui, X., V. Goel and B. Kingsbury, "Data augmentation for deep neural network acoustic modeling", *IEEE/ACM Transactions on Audio, Speech and Language Processing (TASLP)*, Vol. 23, No. 9, pp. 1469–1477, 2015.

65. Povey, D. and P. Woodland, "Minimum Phone Error and I-smoothing for improved discriminative training", *IEEE International Conference on Acoustics Speech and Signal Processing*, pp. 105–108, 2002.

66. Weng, C., B.-h. F. Juang and D. Povey, "Discriminative Training Using Non-uniform Criteria for Keyword Spotting on Spontaneous Speech", *Interspeech*, pp. 300–312, 2012.

67. Chan, C.-a. and L.-s. Lee, "Integrating frame-based and segment-based dynamic time warping for unsupervised spoken term detection with spoken queries", *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 5652–5655, 2011.

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

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

70. 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.

71. Szoke, I., L. J. Rodriguez-Fuentes, A. Buzo, X. Anguera, F. Metze, J. Proenca, M. Lojka and X. Xiao, "Query by example search on speech at mediaeval 2015", *Working Notes Proceedings of the MediaEval 2015 Workshop, September 14-15, 2015, Wurzen, Germany*, 2015.

72. Szöke, I., M. Skácel and L. Burget, "BUT QUESST 2014 system description", *Proceedings of CEUR Workshop*, pp. 1–2, 2014.

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

74. Anguera, X., "Speaker independent discriminant feature extraction for acoustic pattern-matching", *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 485–488, 2012.

75. Anguera, X. and M. Ferrarons, "Memory efficient subsequence DTW for query-by-example spoken term detection", *IEEE International Conference on Multimedia and Expo (ICME)*, pp. 1–6, 2013.

76. Chan, C.-a. and L.-s. Lee, "Unsupervised spoken-term detection with spoken queries using segment-based dynamic time warping", *Interspeech*, 2010.

77. Gupta, V., J. Ajmera, A. Kumar and A. Verma, "A language independent approach to audio search", *Interspeech*, 2011.

78. Hazen, T. J., W. Shen and C. White, "Query-by-example spoken term detection using phonetic posteriorgram templates", *IEEE Workshop on Automatic Speech Recognition & Understanding*, pp. 421–426, 2009.

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

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

81. Rabiner, L. R. and B.-H. Juang, *Fundamentals of speech recognition*, PTR Prentice Hall, 1993.

82. Tzanetakis, G., A. Ermolinskyi and P. Cook, "Pitch histograms in audio and symbolic music information retrieval", *Journal of New Music Research*, Vol. 32, No. 2, pp. 143–152, 2003.

83. Müller, M., "Dynamic time warping", *Information retrieval for music and motion*, pp. 69–84, 2007.

84. Brown, M. and L. Rabiner, "An adaptive, ordered, graph search technique for dynamic time warping for isolated word recognition", *IEEE Transactions on acoustics, speech, and signal processing*, Vol. 30, No. 4, pp. 535–544, 1982.

85. Park, A. S. and J. R. Glass, "Unsupervised Pattern Discovery in Speech", *IEEE Transactions on Audio, Speech, and Language Processing*, Vol. 16, pp. 186–197, 2008.

86. Muscariello, A., G. Gravier and F. Bimbot, "Audio keyword extraction by unsupervised word discovery", *Interspeech*, pp. 1–4, 2009.

87. Anguera, X., R. Macrae and N. Oliver, "Partial sequence matching using an unbounded dynamic time warping algorithm", *IEEE International Conference on Acoustics Speech and Signal Processing (ICASSP)*, pp. 3582–3585, 2010.

88. Jansen, A., K. Church and H. Hermansky, "Towards Spoken Term Discovery At Scale With Zero Resources", *Interspeech*, pp. 1676–1679, 2010.

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

90. Keogh, E. and C. A. Ratanamahatana, "Exact indexing of dynamic time warping", *Knowledge and information systems*, Vol. 7, No. 3, pp. 358–386, 2005.

91. Zhang, Y. and J. R. Glass, "A Piecewise Aggregate Approximation Lower-Bound Estimate for Posteriorgram-Based Dynamic Time Warping.", *Interspeech*, 2011.

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

93. Anguera, X., "Information retrieval-based dynamic time warping.", *Interspeech*, 2013.

94. Çetinkaya, G., B. Gündoğdu and M. Saraçlar, "Pre-filtered dynamic time warping for posteriorgram based keyword search", *IEEE Spoken Language Technology*

Workshop (SLT), pp. 376–382, 2016.

95. Fiscus, J., J. Ajot and G. Doddington, "The spoken term detection (STD) 2006 evaluation plan", *NIST USA*, 2006.

96. Anguera, X., L.-J. Rodriguez-Fuentes, A. Buzo, F. Metze, I. Szöke and M. Penagarikano, "QUESST2014: evaluating query-by-example speech search in a zero-resource setting with real-life queries", *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 5833–5837, 2015.

97. Povey, D., A. Ghoshal, G. Boulianne, L. Burget, O. Glembek, N. Goel, M. Hannemann, P. Motlicek, Y. Qian, P. Schwarz, J. Silovsky, G. Stemmer and K. Vesely, "The Kaldi Speech Recognition Toolkit", *IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*, 2011.

98. Zhang, X., J. Trmal, D. Povey and S. Khudanpur, "Improving deep neural network acoustic models using generalized maxout networks", *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 215–219, 2014.

99. Bisani, M. and H. Ney, "Joint-sequence models for grapheme-to-phoneme conversion", *Speech communication*, Vol. 50, No. 5, pp. 434–451, 2008.

100. Mueller, M., "Dynmic Time Warping", *Information Retrieval for Music and Motion*, 2007.

101. Anguera, X., L. J. Rodriguez-Fuentes, I. Szoke, A. Buzo and F. Metze, "Query-by-Example Spoken Term Detection Evaluation on Low-Resource Languages", *Proceedings of the International Workshop on Spoken Language Technologies for Underresourced Languages (SLTU)*, Vol. 24, p. 31, 2014.

102. Bromley, J., J. W. Bentz, L. Bottou, I. Guyon, Y. LeCun, C. Moore, E. Säckinger and R. Shah, "Signature verification using a "Siamese" time delay neural net-

work", *International Journal of Pattern Recognition and Artificial Intelligence*, Vol. 7, No. 04, pp. 669–688, 1993.

103. Chopra, S., R. Hadsell and Y. LeCun, "Learning a similarity metric discriminatively, with application to face verification", *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, Vol. 1, pp. 539–546, 2005.

104. Čiginas, A. and D. Pumputis, "Gini's mean difference and variance as measures of finite populations scales", *Lithuanian Mathematical Journal*, Vol. 55, No. 3, pp. 312–330, 2015.

105. 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", *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 8272–8276, 2013.

106. Wang, Y. and F. Metze, "An in-depth comparison of keyword specific thresholding and sum-to-one score normalization", *Interspeech*, 2014.

107. Montague, M. and J. A. Aslam, "Relevance score normalization for metasearch", *Proceedings of the tenth international conference on Information and knowledge management*, pp. 427–433, ACM, 2001.

108. Szöke, I., L. Bürget, F. Grézl, J. H. Černockỳ and L. Ondel, "Calibration and fusion of query-by-example systems—But SWS 2013", *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 7849–7853, 2014.

109. Brummer, N., J. Cernocky, M. Karafiát, D. A. van Leeuwen, P. Matejka, P. Schwarz, A. Strasheim *et al.*, "Fusion of heterogeneous speaker recognition systems in the STBU submission for the NIST speaker recognition evaluation 2006", *IEEE Transactions on Audio, Speech, and Language Processing*, Vol. 15,

No. 7, pp. 2072–2084, 2007.

110. Abad, A., L. J. Rodriguez-Fuentes, M. Penagarikano, A. Varona and G. Bordel, "On the calibration and fusion of heterogeneous spoken term detection systems.", *Interspeech*, pp. 20–24, 2013.

111. Trmal, J., G. Chen, D. Povey, S. Khudanpur, P. Ghahremani, X. Zhang, V. Manohar, C. Liu, A. Jansen, D. Klakow *et al.*, "A keyword search system using open source software", *IEEE Spoken Language Technology Workshop (SLT)*, pp. 530–535, 2014.

112. Gracia, C., X. Anguera and X. Binefa, "Combining temporal and spectral information for Query-by-Example Spoken Term Detection", *Proceedings of the 22nd European Signal Processing Conference (EUSIPCO)*, pp. 1487–1491, IEEE, 2014.