Language Independent Query by Example Spoken Term Detection

présentée le 4 Juin 2019
à la Faculté des Sciences et Techniques de l'Ingénieur
Programme Doctoral en Génie Électrique (EDEE)
Laboratoire LIDIAP (Idiap Research Institute)
École Polytechnique Fédérale de Lausanne

pour l'obtention du grade de Docteur ès Sciences
par

Dhananjay Ram

acceptée sur proposition du jury:

Prof. Jean-Philippe Thiran, président du jury
Prof. Hervé Bourlard, directeur de thèse
Dr. Jean-Marc Vesin, rapporteur
Prof. Jan Cernocký, rapporteur
Dr. Xavier Anguera, rapporteur
Lausanne, EPFL, 2019

This thesis is dedicated to
my loving parents and grandparents

Abstract

Language independent query by example spoken term detection (QbE-STD) is the problem of retrieving audio documents from an archive, which contain a spoken query provided by a user. This is usually casted as a hypothesis testing and pattern matching problem, also referred to as a "zero-resource task" since no specific training or lexical information is required to represent the spoken query. Thus it enables multilingual search on unconstrained speech without requiring a full speech recognition system. State-of-the-art solutions typically rely on Dynamic Time Warping (DTW) based template matching using phone posteriors features estimated by Deep Neural Networks (DNN).

In this thesis, we aim at exploiting the low-dimensional subspace structure of speech signal, resulting from the constrained human speech production process. We exploit this subspace structure to improve over the state-of-the-art to (1) generate better phone or phonological posterior features, and (2) to improve the matching algorithm. To enhance phone posteriors, we learn the underlying phonetic subspaces in an unsupervised way, and use the sub-phonetic attributes to extract the phonological components in a supervised manner. To improve the matching algorithm, we model the subspaces of the spoken query using its phone posterior representation. The resulting model is used to compute distances between the subspaces of the query and the phone posteriors of each audio document. These distances are then used to detect occurrences of the spoken query, while also regularizing the DTW to improve the detection scores.

In addition to optimizing different components of the state-of-the-art system, we propose a novel DNN-based QbE-STD system to provide an end-to-end learning framework. Towards that end, we replace the DTW based matching with a Convolutional Neural Network (CNN) architecture. We also learn multilingual features, aimed at obtaining language independent representation. Finally, we integrate the feature learning and CNN-based matching to jointly train and further improve the QbE-STD performance.

We perform experiments using the challenging AMI meeting corpus (English), as well as multilingual datasets such as Spoken Web Search 2013 and Query by Example Search on Speech Task 2014, and show significant improvements over a very competitive state-of-the-art system.

Keywords: *Deep neural network (DNN), phone posteriors, phonological posteriors, subspace detection, subspace regularization, convolutional neural network (CNN), query by example, spoken term detection, dynamic time warping (DTW)*

Résumé

La détection de mot clés avec une requête par l'exemple (QbE-STD), indépendante de la langue, consiste à identifier quels documents audio dans une archive contiennent une requête prononcée par un utilisateur. Ceci est généralement formulé comme un test d'hypothèse et un problème d'appariement de modèles, également appelé "tâche zéro ressource", puisque aucune donnée d'apprentissage ou information lexicale spécifique n'est nécessaire pour représenter le signal de parole donné. Ainsi, il permet de faire une recherche dans un signal de parole multilingue sans contrainte, sans nécessiter un système de reconnaissance vocale complet. L'état de l'art se base généralement sur l'association de modèles utilisant la déformation temporelle dynamique (DTW) et sur l'exploitation des paramètres postérieurs de phonèmes monolingues, produits par des réseaux de neurones artificiels profonds (DNN).

Dans cette thèse, nous souhaitons exploiter le fait que les signaux de parole résident dans des sous espaces de faible dimension, en conséquence des contraintes du processus de production de la parole humaine. Nous exploitons la structure de ces sous espaces pour améliorer l'état de l'art, en (1) générant de meilleurs paramètres postérieurs phonétiques ou phonologiques et (2) en améliorant l'algorithme d'appariement. Afin d'améliorer les postérieurs phonétiques, nous apprenons les sous espaces phonétiques sous jacents de façon non supervisée, et utilisons les attributs subphonétiques pour extraire les composants phonologiques de façon supervisée. Pour améliorer l'algorithme d'appariement, nous modélisons les sous espaces de la requête prononcée en exploitant sa représentation postérieure phonétique. Le modèle résultant est utilisé pour calculer les distances entre les sous espaces de la requête et les postérieurs phonétiques de chaque document audio. Ces distances sont ensuites utilisées pour détecter les occurrences de la requête prononcée, tout en régularisant la DTW pour améliorer les scores de détection.

En plus d'optimiser les différentes composantes de l'état de l'art, nous proposons un système QbE-STD entièrement basé sur des DNN, dans l'optique de fournir une structure d'apprentissage de bout en bout. Pour ce faire, nous remplaçons l'appariement basé sur la DTW par une architecture innovante basée sur les réseaux de neurones convolutifs (CNN). Afin d'obtenir une représentation indépendante de la langue, le modèle apprend également des paramètres multilingues. Enfin, nous intégrons l'apprentissage de caractéristiques et l'appariement basé sur les CNN dans un apprentissage joint, qui permet d'améliorer encore les performances de la QbE-STD.

Nous conduisons nos expériences sur le corpus difficile *AMI meeting* (Anglais), ainsi que sur de multiples bases de données multilingues, telles que *Spoken Web Search 2013* et *Query by Example Search on Speech Task 2014*, et démontrons des améliorations significatives sur un système de l'état de l'art très compétitif.

Mots clefs: *réseaux de neurones profonds (DNN), postérieurs phonétiques, postérieurs phonologiques, détection des sous espaces, régularisation des sous espaces, réseaux de neurones convolutifs (CNN), requête par l'exemple, la détection de mot clés, déformation temporelle dynamique*

Acknowledgements

I would like express gratitude to my thesis adviser Prof. Hervé Bourlard for giving me the opportunity to pursue PhD. I thank him for his guidance, insights and motivational comments throughout my PhD. He gave me the freedom to pursue my research ideas and was always patient to discuss it with me. I would like to thank Dr. Petr Motlicek for giving me the opportunity for an internship at Idiap which introduced me to the great research work of the speech group and Idiap in general. This internship motivated me to pursue my PhD at Idiap. A special thank goes to Dr. Afsaneh Asaei, who mentored me in the initial years of my PhD. I would also like to thank my thesis jury members: Prof. Jean-Phillipe Thiran, Prof. Jan Cernocky, Dr. Xavier Anguera, and Dr. Jean-Marc Vesin for evaluating my thesis and providing insightful comments.

I sincerely thank Swiss National Science Foundation (SNSF) for supporting my research through two different projects: A-MUSE (200020-144281) and PHASER-QUAD (200020-169398). My gratitude also goes to research teams at Amazon Alexa and Microsoft Research for giving me internship opportunities. These internships introduced me to the industrial research and taught me how to use ideas in different products for real-life usage.

I would like to thank the secretariat and the help-desk at Idiap for providing a very professional work environment and helping me with technical and non-technical resources. The most important part of work and life in Martigny have been the wonderful friends I made over the last 5 years with whom I shared countless tea, coffee, wine, dinner, ski trips, hikes etc. I thank Subhadeep, Pranay, Skanda, Blaise, Pierre-Edouard, Gulcan, Suraj, Vinayak, Bastian, Weipeng, Sibo, Banri, Apoorv, Julian, Nikos, Nicholas, Teja, Angel, Mathew, and many more friends for making Martigny and Switzerland an amazing place to live. A special thanks to Lesly whose constant support and motivation made everything look easier.

Finally, I am utmost grateful to my whole family, especially my parents for their constant love, support, guidance, blessings, and freedom they provided me to pursue my interests; and I am immensely thankful to my brothers Sanjay and Ajay for all their encouragement and love.

Martigny, July 2019 Dhananjay

Contents

List of Figures

List of Tables

List of Algorithms

1 Introduction

Internet has become an integral part of our life due to its capability to provide desired information in a fast and reliable manner. This information is searched through huge databases by primarily relying on text. However, the growing use of internet and social media leads to massive amount non-textual data (e.g. audio, video etc.) constantly being uploaded. The search through this kind of data still depends on its textual description which may not be always available or it is insufficient for representing the complete contents of data. Therefore, text based retrieval algorithms give very limited search results. Moreover, it is desirable to search through those contents using speech as a natural and generic medium of communication.

Spoken term detection (STD) aims at solving this problem by relying on a user generated spoken query to search through a database and retrieve all audio documents containing the query as shown in Figure 1.1. Traditionally, STD is performed by cascading an automatic speech recognition (ASR) system with text based retrieval techniques (Lee and Pan, 2009; Chia et al., 2008; Mandal et al., 2013; Shen et al., 2009). In this approach, spoken queries as well as the test utterances are converted into a sequence of words or symbols and information retrieval methods are applied to detect the queries. Performance of a STD system is largely dependent on the accuracy of the underlying ASR system (Shen et al., 2009; Parada et al., 2009). However, building a good ASR system requires large quantity of annotated data and very few languages have such resources. Thus, STD systems are not useful for most of the languages.

This motivated us to focus on language independent spoken term detection (LI-STD), also known as query by example spoken term detection (QbE-STD) (Rodriguez-Fuentes et al., 2014; Szoke et al., 2015). In this scenario, no training data is provided, making it a zero-resource task. Thus, the data can be generated in any language with no constraints on vocabulary, pronunciation lexicon, accents etc. It is essentially a pattern matching problem in the context of speech data where the targeted pattern is the information encoded using speech signal and presented to the system as a spoken query. The solution can be very useful in searching through multilingual audio archives which consists of data from the news channels, radio broadcasts, internet, social media etc.

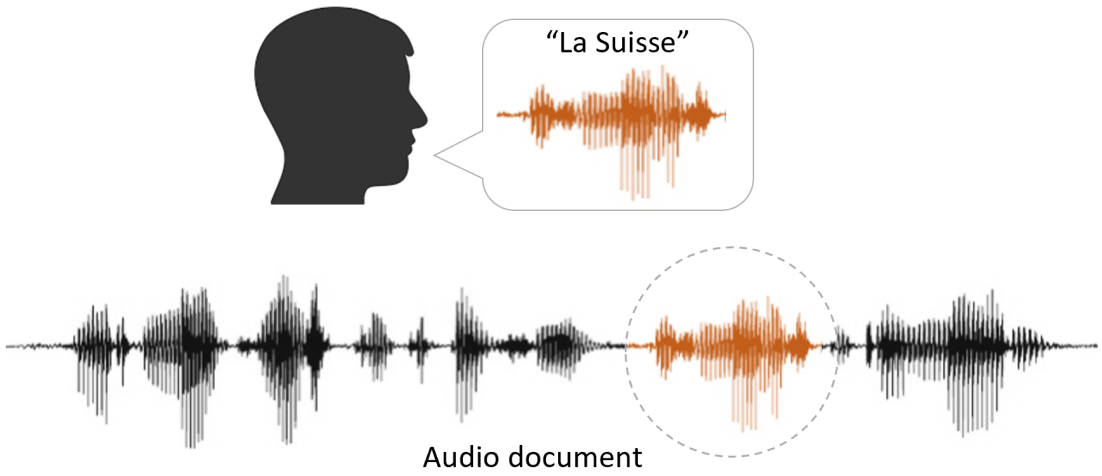

Figure 1.1: Query-by-Example Spoken Term Detection

Current approaches for QbE-STD are largely dominated by template matching techniques for their superior performance to the statistical methods in zero-resource conditions (Hazen et al., 2009; Mandal et al., 2013; Anguera et al., 2014; Rodriguez-Fuentes et al., 2014). These approaches mainly consist of two steps: feature extraction and template matching. Phone posterior features (posterior probabilities of a set of phonetic classes) estimated at the output of a deep neural network (DNN) have been very successful (Rodriguez-Fuentes et al., 2014) for this purpose. A Dynamic Time Warping (DTW) algorithm (Rabiner et al., 1978) is generally used to find the degree of similarity between a query and a test utterance. Although state-of-the-art performance is achieved with this approach, the systems are still far from being usable in real life scenario.

In this thesis, we propose several ways to exploit the low-dimensional subspace structure of speech for QbE-STD. We utilize this at both stages of the system to improve over the state-of-the-art by (1) generating better phone or phonological posterior features, and (2) improving the matching algorithm to compute better likelihood score. In addition, we introduce a neural network based end-to-end learning framework to replace the DTW and jointly learn the acoustic representation and matching algorithm to further improve the performance.

The rest of this chapter is organized as follows. In Section 1.1, we discuss our motivation behind the approaches proposed in this thesis. We summarize the contributions of this thesis in Section 1.2. Finally, the chapter-wise outline is presented in Section 1.3.

1.1 Motivation

Spoken utterances are composed of a sequence of words which in turn consist of phones and sub-phonetic components. These linguistic components are generated using the constrained articulatory mechanism of human speech production process which results in speech data

lying on non-linear manifolds (Deng, 2004; King et al., 2007). These manifolds can be modeled as a union of low-dimensional subspaces, and sparse representation is shown to be a useful technique to model these subspaces (Sainath et al., 2011; Gemmeke et al., 2011). This property of speech has been exploited to perform large vocabulary as well as noise robust speech recognition (Sainath et al., 2011; Gemmeke et al., 2011). In this thesis, we aim at exploiting this low-dimensional structure to improve the QbE-STD system.

We developed novel techniques to extract better features as well as improve the DTW based template matching using the subspace properties of speech. This work is based on the training of Deep Neural Networks (DNNs) on several languages to extract multiple monolingual phonetic features. In spite of achieving significant improvements over a strong state-of-the-art baseline, we observed that all systems still suffer from language mismatch when testing on a language that has not been seen during training. This motivated us to extract language independent representations for high-performance QbE-STD even in the case of unseen languages. In this scenario, we focus on DNN-based learning due to its strong modeling capability and success in several other problems e.g. speech recognition (Hinton et al., 2012; Graves et al., 2013), image classification (Simonyan and Zisserman, 2014; He et al., 2016), machine translation (Bahdanau et al., 2014; Miculicich et al., 2018) etc.

In spite of using better features, the DTW-based pattern matching algorithm suffers from its own limitations. For instance, it is not well suited in cases of local mismatches between a test sequence and its associated query, which can result in very low matching scores. Hence, we developed an innovative framework, based on Convolutional Neural Networks (CNN) to be able to properly cope with those cases.

In all of the above work, the two stages (feature extraction and pattern matching) of our QbE-STD system are still optimized independently of each other. Hence, in the spirit of current trends in DNN-based pattern recognition, we finally decided to investigate an "end-to-end" neural network architecture to enable joint optimization of those two stages simultaneously.

1.2 Contributions

The goal of this thesis is to investigate novel approaches to exploit low-dimensional subspace structure of speech signal as well as neural network based techniques to improve over the state of the art QbE-STD system. The contributions of this thesis can be summarized as follows:

- **Sparse recovery of phonetic subspaces**: We propose a novel approach to obtain phonetic subspace features to improve QbE-STD performance. The subspaces are learned using sparse modeling framework and exploited to enhance phone posteriors obtained at the output of Deep Neural networks (DNN). Alternatively, we consider sub-phonetic attributes (phonological information) to model the phonetic subspaces using DNN, resulting in a new set of features. We demonstrate that these two sets of features provide complementary information and a distance fusion method is proposed to integrate that

information to further improve the performance.

- **Spoken term retrieval as a subspace detection problem:** We show that the subspace structure can be used to find queries in a test utterance by casting the problem as subspace detection instead of DTW based template matching. We also show that DTW based template matching can benefit from the subspace detection approach. This is achieved by (i) regularizing the distances for DTW using subspace detection score and (ii) improving the DTW matching score using subspace matching score.

- **QbE-STD based on Convolution Neural Network (CNN)**: We introduce an innovative DNN-based learning framework to replace the DTW based template matching. Inspired from the success in image classification, we use Convolution Neural Networks (CNN) to detect spoken terms. The distance matrix previously used for DTW is considered as an "image" which contains somewhere a good match (quasi-diagonal) pattern if a query occurs in a test utterance, hence casting the query detection problem as a binary classification problem.

- **End-to-End CNN-based QbE-STD**: We finally extend the above CNN based query matching approach by integrating it with a feature extraction network and training the whole architecture in an end-to-end manner. The feature extraction network is implemented using multilingual bottleneck network to obtain language independent representation which in turn improves the CNN's ability to detect a query. We also show that CNN based matching network can be used as a loss function to obtain better language independent features.

- **Experimental validation**: State-of-the-art DTW based QbE-STD system is implemented as a very strong baseline. All the proposed approaches are evaluated against the baseline system on different challenging datasets, including the AMI meeting corpus (English), as well as multilingual datasets such as Spoken Web Search 2013 and Query by Example Search on Speech Task 2014, demonstrating in each case significant improvements.

1.3 Thesis Outline

We present the organization of this thesis by briefly describing the main goal of each of its constituent chapters.

Chapter 2: We present the key components of a QbE-STD system along with different types of acoustic feature representation. We also discuss different databases used for training and evaluation, and several evaluation metric for comparison purposes. Then the state-of-the-art QbE-STD system is described which is used as our baseline system.

Chapter 3: We aim at exploiting the low-dimensional subspace structure of speech to obtain better representation for DTW based template matching. A data-driven and a knowledge based approach is proposed for this purpose and a distance fusion techniques is employed to

integrate those features.

Chapter 4: We propose to improve the matching algorithm by modeling the query subspaces and using those models to compute subspace detection scores. These scores are then used for subspace regularized DTW as well as subspace based re-scoring of DTW. Both approaches are shown to provide significant improvement over the baseline.

Chapter 5: We present a multilingual bottleneck network to extract language independent features. These features are used for DTW based template matching as well as a novel CNN based matching for QbE-STD. The CNN matching network is integrated with the bottleneck feature extractor into a single architecture to jointly train and optimize the network for significant performance improvement.

Chapter 6: We discuss the conclusions of this thesis and provide some possible directions for future work.

2 Background on Query-by-Example Spoken Term Detection

Contents

2.1 Introduction

In this chapter, we provide a brief background on the task of query-by-example spoken term detection (QbE-STD). We summarize several techniques proposed in the literature towards solving QbE-STD in Section 2.2. The details of different acoustic feature vectors used in this thesis to represent the speech signal is provided in Section 2.3. Then we describe different monolingual and multilingual databases used to train and evaluate the proposed systems in Section 2.4. Several evaluation metrics are used to thoroughly examine and compare our system with the state-of-the-art as summarized in Section 2.5. Finally, we present a brief description of our baseline system in Section 2.6.

2.2 Related Works

In this section, we summarize different techniques proposed for QbE-STD. The first set of methods consists of a two step approach: feature extraction and template matching as discussed earlier. The spoken queries as well as test utterances can be represented using mel frequency cepstral coefficient (MFCC) or perceptual linear prediction (PLP) based spectral features. These spectral features were initially investigated for template matching task (Sakoe and Chiba, 1978). However these features were outperformed by posterior features, which can be estimated from models trained in both supervised and unsupervised manner (Hazen et al., 2009; Rodriguez-Fuentes et al., 2014; Zhang and Glass, 2009). Gaussian mixture model (GMM) based posteriors are estimated from a GMM trained in an unsupervised manner where the feature dimensions correspond to posterior probabilities of different Gaussian components in the model (Zhang and Glass, 2009; Park and Glass, 2008). On the other hand, a deep Boltzman machine (DBM) trained in unsupervised as well as semi-supervised manner can be used to extract posterior features. The unsupervised training of DBM can capture hierarchical structural information from unlabeled data. In (Zhang et al., 2012), the authors first train a DBM using unlabeled data and then fine tune it using small amount of labeled data. In another approach, GMM based posteriors were used as labels for the DBM training (Zhang et al., 2012). Posteriors from DBM in both cases perform better than GMM posteriors for QbE-STD.

The supervised approach to extract posterior features primarily relies on training a DNN using labeled data. In case of zero resource languages, the DNN is first trained using data from different well resourced languages where the labels can indicate mono-phones, context dependent phones or senones (Hazen et al., 2009; Rodriguez-Fuentes et al., 2014). The DNN is then used to extract posterior features to perform template matching for QbE-STD. In this approach, the posteriors are interpreted as a characterization of instantaneous content of the speech signal, irrespective of the underlying language (Rodriguez-Fuentes et al., 2014). DNNs with bottleneck layer have also been trained in a similar multilingual setting to compute bottleneck features for QbE-STD (Szoke et al., 2014; Chen et al., 2017).

Features extracted from the spoken query and test utterance are used to compute a frame-level distance matrix and a DTW algorithm is used to find the degree of similarity between them. Standard DTW algorithm performs an end-to-end comparison between two temporal sequences, making it difficult to use for QbE-STD because the query can occur anywhere in the test utterance as a sub-sequence. In segmental DTW (Park and Glass, 2008), the distance matrix is segmented into overlapping diagonal bands where the width of the band indicates temporal distortion allowed for matching. But the width of each band limits its capability to deal with signals of widely varying speaking rate. Slope-constrained DTW (Zhang and Glass, 2009) was proposed to deal with this problem by penalizing the slope of warping path which maps the spoken query within a test utterance. It limits the number of frames to be mapped in the test audio corresponding to a frame in the query and vice versa. In sub-sequence DTW (Müller, 2007), the cost of insertion is forced to be 0 in the beginning and end of a query, which enables the warping path to begin and end at any point in the test audio and finds a sub-sequence best matching the query.

More recent approaches are aimed at minimizing the computational cost or memory footprints of the DTW based search techniques (Anguera, 2013; Anguera and Ferrarons, 2013; Asaei et al., 2018). Information retrieval based DTW (Anguera, 2013) proposes to index the frames of test utterance and uses hashing techniques to reduce the search space. On the other hand, memory efficient DTW (Anguera and Ferrarons, 2013) proposed an improvement over subsequence DTW by using a lookup table for faster backtracking and an alternative normalization of the warping path. Alternative to DTW, subspace detection based approach relying on frame level detection scores have been proposed to make the search faster (Ram et al., 2015, 2016). Additionally, model based approaches have been proposed to deal with acoustic and speaker mismatch conditions. These methods depend on unsupervised acoustic unit discovery, followed by the use of hidden Markov models (HMM) to model those units. These HMMs are then used to find symbolic representation of the query and test utterance, and symbolic search techniques are used to retrieve the query.

2.3 Feature Vectors

In this section, we describe different feature vectors used in this thesis to represent speech signal.

2.3.1 Mel Frequency Cepstral Coeffcients

In speech processing, Mel Frequency Cepstral Coeffcients, commonly known as MFCC feature (Davis and Mermelstein, 1980) is a short-term power spectrum based representation of speech signal. These features are widely used in automatic speech recognition and speaker recognition systems, which are extracted by windowing the speech signal into short segments. Generally, a sliding window of 25ms with a shift of 10ms is used. The short segments are assumed to be stationary and is referred to as a frame. Fast Fourier transform (FFT) is applied on

the sequence of frames to compute the corresponding power spectrum. This step is followed by a filter-bank analysis to obtain the energies in various frequency regions. This is achieved by using non-linearly placed triangular filters (usually 40). The filters are linearly spread in the mel domain which is chosen to mimic the human auditory perception. The final step is to compute discrete cosine transform of log of the filter-bank energies. It is required to de-correlate the filter-bank energies. Typically for speech recognition and related tasks, the first 13 coefficients are used. These coefficients are concatenated with their delta (Δ) and double-delta ($\Delta\Delta$) features in order to incorporate the trajectory information of features. Let the MFCC feature vectors for an utterance is represented by, $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_T]$. The corresponding delta features are computed using linear regression over a window of W as follows:

$$\Delta_t = \frac{\sum_{w=1}^{W} w(\mathbf{x}_{t+w} - \mathbf{x}_{t-w})}{2\sum_{w=1}^{W} w^2} \tag{2.1}$$

where, Δ_t is the delta feature vector for the t-th frame of an utterance $\mathbf{X}$. The double-delta features can be obtained by successively applying to Equation (2.1).

2.3.2 Posterior Feature

Posterior feature (e.g. mono-phone, tri-phone) is a vector representation consisting of class conditional posterior probabilities given a short window of the acoustic features (e.g. MFCC features, filter-bank features). Deep feed-forward Neural Networks (DNN) are typically used to estimate these features, however Convolutional Neural Network (CNN) or Long Short Term Memory networks (LSTM) can also be used (Bourlard and Morgan, 1994; Hinton et al., 2012). These posteriors features have been shown to be very effective for ASR systems which motivated the researchers to use them for template matching tasks (Hazen et al., 2009; Rodriguez-Fuentes et al., 2014).

Mono-phone based posterior features have been primarily used for QbE-STD (Rodriguez-Fuentes et al., 2014; Ram et al., 2016). The setup for extracting these posterior features is illustrated in Figure 2.1. In the first step, MFCC features are extracted from the speech signal as described in previous section. Those MFCC features along with some acoustic context (left and right) are used as input to train a DNN for estimating the phone posterior probabilities. MFCC features of test data are then forward passed through the trained DNN to compute corresponding phone posterior vectors.

2.3.3 Bottleneck Feature

Bottleneck features (Hinton and Salakhutdinov, 2006; Yu and Seltzer, 2011; Veselỳ et al., 2012) are low-dimensional representation of data generally obtained from a hidden bottleneck layer of a DNN. This bottleneck layer has a smaller number of hidden units compared to the size of other layers. The smaller size layer constrains the information flow through the network

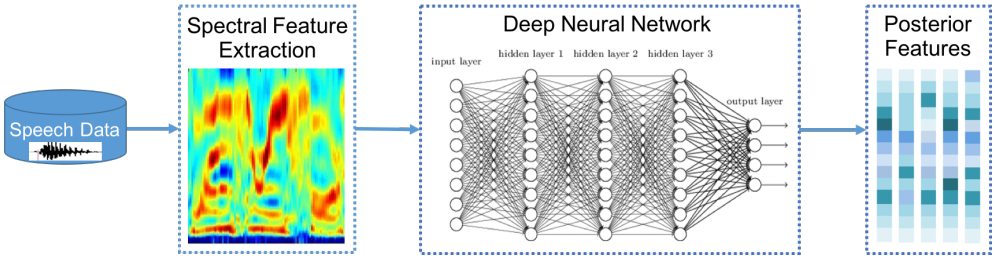

Figure 2.1: Posterior feature extraction using a deep neural network. First, Mel Frequency Cepstral Coefficient (MFCC) based features are extracted over a sliding window. These features, together with their acoustic context, are then fed to a DNN to estimate phone conditional posterior probabilities.

which enables the network to focus on the information that is necessary to optimize the final objective. Bottleneck features have been commonly used in both auto-encoders (Hinton and Salakhutdinov, 2006) as well as DNNs for classification (Yu and Seltzer, 2011). Language independent bottleneck features can be obtained using multi-lingual objective function (Veselỳ et al., 2012).

The bottleneck features can be extracted by following a very similar setup presented in Figure 2.1. The primary difference is that the network has a bottleneck layer and the features come out of the bottleneck layer instead of the output layer.

2.4 Databases

We have used 4 different databases to train and evaluate different systems proposed in this work. We present a brief description of these datasets in the following.

2.4.1 AMI meeting corpus

The AMI (Augmented Multi-party Interaction) meeting corpus (McCowan et al., 2005) was collected to support multi-disciplinary research. The dataset was recorded in several rooms in different locations equipped with a variety of microphones, video cameras, presentation slide capture devices etc. The corpus contains real meeting recordings as well as scenario driven meetings which were designed to promote a wide range of realistic behavior. The meeting rooms were setup to record both close-talking and far-field audio. The close-talk speech was recorded using individual headset microphone (IHM) with wireless setup, whereas the far-field audio was recorded using four or eight miniature omni-directional microphone arrays with wired setup. We use IHM recordings as well as single distant microphone (SDM) recordings with mic-id 1 to evaluate the performance of our system for both types of recordings.

Both datasets are partitioned into three groups[1] to train the corresponding DNN. The partition

[1] http://groups.inf.ed.ac.uk/ami/corpus/datasets.shtml

Table 2.1: Number of different types of queries available in both sets in SWS 2013, partitioned according to the number of examples per query.

Query Set	Examples per query		
	1	3	10
Development	311	100	94
Evaluation	310	100	93

consists of about 81 hours of speech for training and about 9 hours for each of the development and evaluation. Although, the meeting language was English, many participants were non-native speakers. In addition, the recordings contain considerable amount of overlapping speech (competing speakers).

2.4.2 Spoken Web Search (SWS) 2013

We consider the Spoken Web Search (SWS) database from MediaEval 2013 benchmarking initiative (Anguera et al., 2013) for training and evaluation of our QbE-STD system. This database was the result of a joint effort between several institution to provide a challenging new dataset for QbE-STD task. It consists of 20 hours of audio recordings (10762 utterances) as search space form 9 different low-resourced languages: Albanian, Basque, Czech, non-native English, Isixhosa, Isizulu, Romanian, Sepedi and Setswana. The data was collected in varying acoustic conditions and in different amounts from each language. Some recordings were obtained from in-room microphones while others were obtained through street recordings with cellphones. The recordings also had variable sampling rate. Thus, all data has been converted to 8KHz/ 16bit WAV files.

The dataset also contains two sets queries for use in the development and evaluation. There are 505 queries in the development set and 503 queries in the evaluation set. Each set consists of 3 types of queries depending on the number of examples available per query: 1, 3 and 10 examples. The number of queries available in each category is shown in Table 2.1. We also present the number of queries per language in both development and evaluation set in Table 2.2

2.4.3 Query by Example Search on Speech Task (QUESST) 2014

Query by Example Search on Speech Task (QUESST) database (Anguera et al., 2014) is part of the MediaEval 2014 challenge similar to SWS 2013 database from previous year. The search corpus consists of ~23 hours of audio recordings (12492 utterances) in 6 different low-resourced languages: Albanian, Basque, Czech, non-native English, Romanian and Slovak. Similar to SWS 2013, this dataset was also recorded in varying acoustic conditions with variable sampling rate. The dataset consists of 560 development queries and 555 evaluation queries which were separately recorded than the search corpus. Unlike SWS 2013 datatset, all queries

Table 2.2: Database content (number of queries, utterances in search space, type of speech) disaggregated per language in SWS 2013

Language	Search Space (minutes / utterances)	Number of queries		Type of speech
		Development	Evaluation	
Albanian	127 / 968	50	50	read
Basque	192 / 1,841	100	100	broadcast / read
Czech	252 / 3,667	94	93	conversational
Isixhosa	65 / 395	25	25	read
Isizulu	59 / 395	25	25	read
NNEnglish	141 / 434	61	60	lecture
Romanian	244 / 2,272	100	100	read
Sepedi	69 / 395	25	25	read
Setswana	51 / 395	25	25	read
Total	1,196 / 10,762	505	503	mixed

have only one example available. We present the number of queries per language in both development and evaluation set in Table 2.3

The primary difference from SWS 2013 task is that, QUESST 2014 considers two types of 'approximate matching' along with the 'exact matching'. These 3 types of matching are defined as follows:

- **Type 1 (Exact):** The occurrences which exactly match the lexical representation of the query are considered as detections. E.g. the query 'brown bear' would match the utterance 'I see a brown bear'.
- **Type 2 (Variant):** In this case, query occurrences that has slight lexical variations at the start or end of a query are considered as detections. Only queries having more than 5 phoneme (250 ms) are used for this task, and the matching part was required to be much greater than the non-matching part. E.g. the query 'researcher' would match an utterance containing 'research' and vice-versa.
- **Type 3 (Reordering/Filler):** In case of a multi-word query, a detection is required to contain all words in the query, however the words may appear in different order, possibly with some filler content between those words. E.g. the query 'white snow' would match an utterance containing either 'snow is white', 'whitest snow' or 'whiter than snow'. Note, the queries in this case are spoken continuously and the detections are allowed to contain a large number of filler content between words.

2.4.4 GlobalPhone Corpus

GlobalPhone is a multilingual speech database developed by the Karlsruhe Institute of Technology (KIT) (Schultz et al., 2013). It consists of high quality recordings of read speech with

Table 2.3: Database content (number of queries, utterances in search space, type of speech) disaggregated per language in QUESST 2014

Language	Search Space (mins./# utt)	Number and subset of queries		Type of speech
		Dev (T1/T2/T3)	Eval (T1/T2/T3)	
Albanian	127 / 968	50 (20/13/16)	50 (18/13/17)	read
Basque	192 / 1,841	70 (16/33/21)	70 (30/19/21)	broadcast
Czech	237 / 2,653	100 (77/24/27)	100 (73/27/32)	conversational
NNEnglish	273 / 2,438	138 (46/46/46)	138 (46/46/46)	lecture
Romanian	244 / 2,272	100 (46/21/31)	100 (43/27/30)	TEDx
Slovak	312 / 2,320	102 (102/53/14)	97 (97/47/10)	parliamentary
Total	1,385 / 12,492	560 (307/190/155)	555 (307/179/156)	mixed

Table 2.4: Partition (in hours) of the GlobalPhone languages used in this work

Language	Train	Dev	Eval	# Phones
French (FR)	22.8	2.1	2.0	38
German (GE)	14.9	2.0	1.5	41
Portuguese (PT)	22.8	1.6	1.8	45
Spanish (ES)	17.6	2.1	1.7	40
Russian (RU)	19.8	2.5	2.4	48

corresponding transcription and pronunciation dictionaries in 20 different languages. It was designed to be uniform across languages in terms of audio quality (type of microphone, noise condition, channel), the collection scenario (task, setup, speaking style), phone set conventions (IPA-based naming of phone) etc. In this work, we use 5 languages to train and estimate multilingual bottleneck features for QbE-STD experiments. The partition for training, development and evaluation set of these language and the corresponding number of phones are shown in Table 2.4.

2.5 Evaluation Metrics

The performance of a QbE-STD system is evaluated using statistical classification theory (Bishop, 2006). We use three different metrics to evaluate our systems: (i) detection error trade-off curve (DET), (ii) maximum term weighted value ($MTWV$) and (iii) minimum normalized cross entropy (C_{nxe}^{min}). First two are based on system hard decision whereas the third one is an information theoretic measure as we will discuss in the following sections.

2.5.1 Detection Error Trade-off (DET) curve

Detection error trade-off curve was proposed to represent the performance of a detection task (e.g. speaker verification, language recognition etc.) that involves a trade-off between error

types: missed detections and false alarms (Martin et al., 1997). These errors for our problem are defined as follows:

- **Missed detection rate:** The missed detection rate ($P_{miss}(q,\theta)$) for a query, q and a threshold, θ is defined as:

$$P_{miss}(q,\theta) = \frac{N_{miss}(q,\theta)}{N_{act}(q)} \tag{2.2}$$

  where, $N_{miss}(q,\theta)$ is the number of occurrences of a query, q not detected by a system for a given threshold, θ and $N_{act}(q)$ is the total number of occurrences of the query, q in the whole search space.

- **False alarm rate:** The false alarm rate ($P_{fa}(q,\theta)$) for a query, q and a threshold, θ is defined as:

$$P_{fa}(q,\theta) = \frac{N_{fa}(q,\theta)}{N - N_{act}(q)} \tag{2.3}$$

  where, $N_{fa}(q,\theta)$ is the number of false detections of a query, q by a system for a given threshold, θ and N is the total number audio documents in the search space.

Then, we can compute the average error rate for all the queries $q \in \mathbf{Q}$ (assuming, all the queries are equally likely) as follows:

$$P_{miss}(\theta) = \frac{1}{|\mathbf{Q}|} \sum_{\forall q \in \mathbf{Q}} P_{miss}(q,\theta) \tag{2.4}$$

$$P_{fa}(\theta) = \frac{1}{|\mathbf{Q}|} \sum_{\forall q \in \mathbf{Q}} P_{fa}(q,\theta) \tag{2.5}$$

These two errors are plotted against each other for various thresholds to obtain a DET curve. The Y-axis represents the missed detection rate ($P_{miss}(\theta)$) while the X-axis represents the false alarm rate ($P_{fa}(\theta)$). Also the axes are non-linear in order to have better comparison among multiple system. Figure 2.2 shows a typical a DET curve comparing two QbE-STD systems. System-2 gives lower miss probability for any false alarm probability in the given range. Clearly System-2 is a better system

2.5.2 Term Weighted Value

The *Term-Weighted Value* (TWV) (Rodriguez-Fuentes and Penagarikano, 2013) is defined as a weighted combination of the miss and false alarm error rates, averaged over the set of all

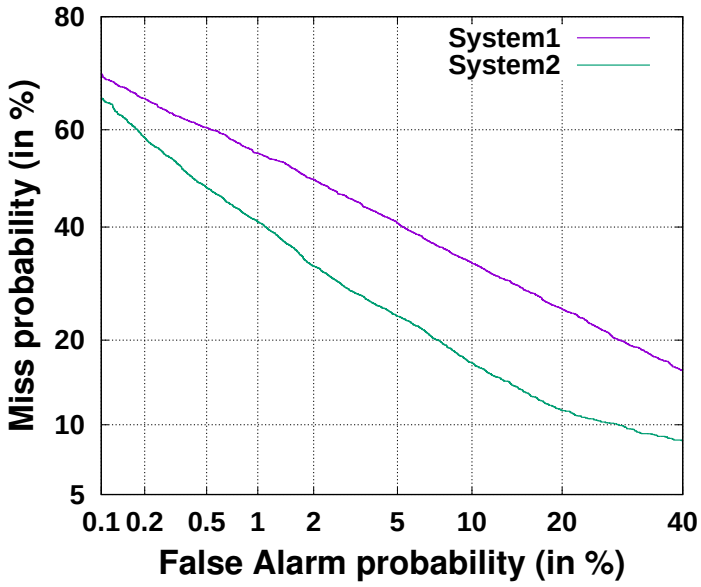

Figure 2.2: DET curves comparing two QbE-STD systems.

queries, as follows:

$$TWV(\theta) = 1 - \frac{1}{|\mathbf{Q}|} \sum_{\forall q \in \mathbf{Q}} (P_{miss}(q,\theta) + \beta . P_{fa}(q,\theta)) \tag{2.6}$$

$$= 1 - (P_{miss}(\theta) + \beta . P_{fa}(\theta)) \tag{2.7}$$

The weight factor $\beta > 0$ is defined as:

$$\beta = \frac{C_{fa} . (1 - P_{target})}{C_{miss} . P_{target}} \tag{2.8}$$

where, $C_{miss} > 0$ and $C_{fa} > 0$ are the costs of miss and false alarm errors respectively. $P_{target} \in [0,1]$ is the prior probability of a target being present in a search utterance. It is assumed to be constant for all queries. The value of $TWV(\theta)$ ranges from $-\beta$ to 1. $TWV(\theta) = 1$ indicates a perfect system, $TWV(\theta) = 0$ indicates a simple system which always makes the decision 'No' (i.e. rejects all the trials), whereas $TWV(\theta) = -\beta$ indicates the worst possible system. We use *Maximum Term Weighted Value* ($MTWV$) to evaluate our QbE-STD systems which is defined as follows:

$$MTWV = \max_{\theta \in \Theta} TWV(\theta) \tag{2.9}$$

The $MTWV$ can be achieved using a well calibrated system.

2.5.3 Normalized Cross Entropy

The output scores of a QbE-STD system for query-search utterance pairs (also called trials) can be viewed as more informative if they represent log-likelihood ratios between the two hypothesis: query is present or absent. Assuming a system $\mathcal{S}$ computes log-likelihood ratios llr_t for a set of trials $t = (q, x) \in T(\mathcal{S})$ corresponding to a query q and a search utterance x, the goodness of those scores can be measured by logarithmic cost function (Rodríguez-Fuentes et al., 2012):

$$C_{\log}(llr_t) = -\log P(\mathcal{G}_t|llr_t) \tag{2.10}$$

where, $\mathcal{G}_t \in \{true, false\}$ is the ground-truth of a trial t. When the system favors the ground-truth, $P(\mathcal{G}_t|llr_t) \approx 1$ so the cost $C_{\log}(llr_t) \approx 0$. In the opposite scenario $P(\mathcal{G}_t|llr_t) \approx 1$, thus $C_{\log}(llr_t) \gg 0$. Now, we can average the cost over all the trials and divide it by log2 to obtain the empirical cross entropy (in information bits) (Rodriguez-Fuentes and Penagarikano, 2013):

$$C_{xe} = \frac{1}{\log 2}\left(\frac{P_{tar}}{|T_{true}(\mathcal{S})|}\sum_{t\in T_{true}(\mathcal{S})} C_{\log}(llr_t) + \frac{1-P_{tar}}{|T_{false}(\mathcal{S})|}\sum_{t\in T_{false}(\mathcal{S})} C_{\log}(llr_t)\right) \tag{2.11}$$

The empirical cross entropy can be normalized by comparing it with a trivial system which outputs non-informative scores (i.e. $llr_t = 0$). The empirical cross entropy of such a system if often called the prior entropy and is computed as:

$$C_{xe}^{prior} = \frac{1}{\log 2}\left(P_{tar}.\log\frac{1}{P_{tar}} + (1-P_{tar}).\log\frac{1}{(1-P_{tar})}\right) \tag{2.12}$$

Finally, the normalized cross entropy is defined as follows:

$$C_{nxe} = \frac{C_{xe}}{C_{xe}^{prior}} \tag{2.13}$$

Normalized cross entropy provides the knowledge that a QbE-STD system has on the ground truth. To be more precise, it computes the information which is not provided by the detection scores generated by a given system. A perfect system produces $C_{nxe} \approx 0$, whereas a non-informative system gives $C_{nxe} = 1$.

The system scores can be calibrated using the following reversible transform:

$$\hat{llr}_t = \alpha.llr_t + \beta \tag{2.14}$$

where, α and β are the calibration parameters, which can be used to minimize the normalized cross entropy:

$$C_{nxe}^{min} = \min_{\alpha,\beta}\left(\hat{C}_{nxe}\right) \tag{2.15}$$

We use C_{nxe}^{min} to evaluate the proposed systems and compare them with the baseline system.

2.5.4 Test of Statistical Significance

We perform Student's t-test to measure statistical significance of the improvements obtained in both $MTWV$ and C_{nxe}^{min} scores by our proposed systems. To perform this test, we compute the scores ($MTWV$ or C_{nxe}^{min} whichever applicable) per query and these scores are considered as samples for a paired-samples t-test. In order to indicate improvement by our systems, the test is one-tailed t-test and the corresponding p-values are indicated with the results.

2.6 Baseline System

The posterior-based QbE-STD system proposed in (Rodriguez-Fuentes et al., 2014) is used as our baseline system. It was the best system in MediaEval challenge 2013 (Anguera et al., 2013) for the task of Spoken Web Search (SWS). The basic framework of the system is presented in this section. It consists of four main modules: (1) posterior feature extraction, (2) speech activity detection, (3) query template construction and (4) DTW based template matching

2.6.1 Posterior Feature Extraction

Phone posterior features are used to detect queries in test utterances. These phone posteriors are estimated by forward passing the corresponding MFCC features through pre-trained neural networks as discussed in Section 2.3.2. These posterior features can be considered as a characterization of instantaneous content of the speech signal independent of the underlying language (Rodriguez-Fuentes et al., 2014). The posteriors are filtered using a speech activity detector before performing QbE-STD experiments as discussed in the following.

2.6.2 Speech Activity Detection

We perform speech activity detection (SAD) to remove the silence and noisy frames from test utterances as well as queries. The SAD relies on the output of the phone recognizers to perform this task. It calculates the probability of no voice activity by summing up the probabilities corresponding to silence and non-speech units in the posterior vector. If for any frame, this probability is highest, the frame is considered silence/noisy and rejected from the corresponding audio. Also, if there are less than 10 frames in an audio file, it is not considered for the experiments to reduce the false alarm rate and computational complexity. Finally, the dimensions corresponding to silence and non-speech units are removed from the posterior vectors as these are unlikely to help in the query matching task (Rodriguez-Fuentes and Penagarikano, 2013).

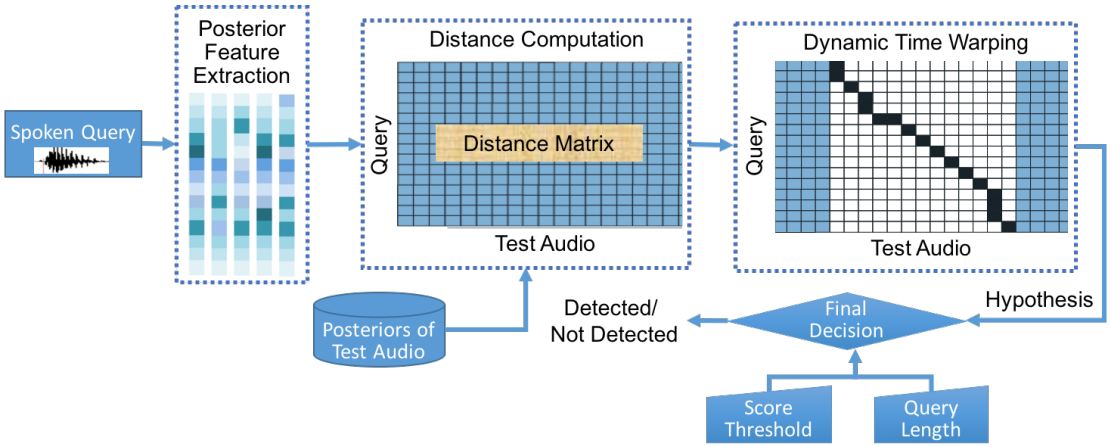

Figure 2.3: Block diagram of the baseline system. After extracting phone posterior features to calculate the normalized distance matrix between query and test utterance, we apply DTW to obtain a sub-sequence matching the query. If the length of the hypothesis is smaller than half the query length, it is discarded to reduce false alarm rate. Otherwise, its score is compared to a threshold to yield a final decision.

2.6.3 Query Template Construction

The posterior features of a query are used to construct a template for DTW based matching. If there is only one example provided for a query, the corresponding posteriors are used as the reference template for performing DTW. If multiple examples are provided for a query, we compute an average template from posteriors of those examples using DTW. In that case, we first select the example with highest number of posteriors (i.e. the longest one) as reference. We then use traditional DTW algorithm (Sakoe and Chiba, 1978) to obtain posteriors-level alignment of the rest of the examples with the reference. The mapped posteriors are averaged together to generate the posteriors of the reference template (Rodriguez-Fuentes et al., 2014; Chen et al., 2015). Finally, this template is used to find the query in test utterances as discussed in the following section.

2.6.4 Template Matching

The template matching algorithm presented in (Rodriguez-Fuentes et al., 2014) is similar to the slope-constrained DTW (Hazen et al., 2009) with some important differences. First, a distance matrix is calculated between each pair of frames of the query and test utterance using logarithm of the cosine distance. We denote the matrix of all posterior feature vectors corresponding to a spoken query and a test utterance by $\mathbf{Q} = [\,\mathbf{q}_1, \mathbf{q}_2, \ldots, \mathbf{q}_m\,]$ and $\mathbf{T} = [\,\mathbf{t}_1, \mathbf{t}_2, \ldots, \mathbf{t}_n\,]$ respectively, where m and n are the corresponding temporal lengths. The distance between each pair of vectors is calculated as follows,

$$d(\mathbf{q}_i, \mathbf{t}_j) = -\log \frac{\mathbf{q}_i \cdot \mathbf{t}_j}{\|\mathbf{q}_i\| \cdot \|\mathbf{t}_i\|} \tag{2.16}$$

The distances are further normalized with respect to the test utterance, **T** using the following test normalization technique.

$$d_{norm}(\mathbf{q}_i, \mathbf{t}_j) = \frac{d(\mathbf{q}_i, \mathbf{t}_j) - d_{min}(i)}{d_{max}(i) - d_{min}(i)} \tag{2.17}$$

where, $d_{min}(i) = \min_{j=1,\ldots,n} d(\mathbf{q}_i, \mathbf{t}_j)$ and $d_{max}(i) = \max_{j=1,\ldots,n} d(\mathbf{q}_i, \mathbf{t}_j)$
This results in a distance matrix comprising values between 0 and 1. Note that, this is a kind of test normalization technique corresponding to each feature vector of the query and was found to be very useful for achieving good performance in SWS 2013.

The query **Q** is detected in the test utterance **T** by minimizing the average distance in a crossing path of the matrix d_{norm}. This crossing path can start at any frame $k_1 \in [1, n]$ of **T**, traverse a continuous segment that is optimally aligned to **Q** (with L vector alignments), and ends at any frame $k_2 \in [k_1, n]$. The average distance in this crossing path is:

$$d_{avg}(\mathbf{Q}, \mathbf{T}) = \frac{1}{L} \sum_{l=1}^{L} d_{norm}(\mathbf{q}_{i_l}, \mathbf{t}_{j_l}) \tag{2.18}$$

where i_l and j_l are indices of the vectors of **Q** and **T** in the alignment, for $l = 1, 2, \ldots, L$, $i_1 = 1$, $i_L = m$, $j_1 = k_1$ and $j_L = k_2$. To find the optimal crossing path, two matrices are defined: $\mathbf{A}(i, j)$ for storing accumulated distance of the partial crossing path ending at (i, j) and $\mathbf{\Lambda}(i, j)$ indicating the length of that path, so that $\mathbf{A}(i, j)/\mathbf{\Lambda}(i, j)$ is the average distance. These matrices are initialized as follows:

$$\begin{aligned} \mathbf{A}(i, 1) &= \sum_{k=1}^{i} d_{norm}(\mathbf{q}_k, \mathbf{t}_1) \\ \mathbf{\Lambda}(i, 1) &= i \end{aligned} \tag{2.19}$$

for $i = 1, ..., m$. The minimization operation runs from $j = 2$ to $j = n$, and for each j, from $i = 1$ to $i = m$ by utilizing a dynamic programming algorithm, as follows:
$i = 1$:

$$\begin{aligned} \mathbf{A}(1, j) &= d_{norm}(\mathbf{q}_1, \mathbf{t}_j) \\ \mathbf{\Lambda}(1, j) &= 1 \end{aligned} \tag{2.20}$$

$i > 1$:

$$\Theta = \{(i, j-1), (i-1, j), (i-1, j-1)\}$$

$$(x_{opt}, y_{opt}) = \arg \min_{(x,y)\in\Theta} \frac{\mathbf{A}(x, y) + \delta(x \neq m).d_{norm}(\mathbf{q}_i, \mathbf{t}_j)}{\mathbf{\Lambda}(x, y) + \delta(x \neq m)} \tag{2.21}$$

$$\begin{aligned} \mathbf{A}(i, j) &= \mathbf{A}(x_{opt}, y_{opt}) + \delta(x_{opt} \neq m).d_{norm}(\mathbf{q}_i, \mathbf{t}_j) \\ \mathbf{\Lambda}(i, j) &= \mathbf{\Lambda}(x_{opt}, y_{opt}) + \delta(x_{opt} \neq m) \end{aligned} \tag{2.22}$$

where,

$$\delta(c) = \begin{cases} 1, & if \quad c = True \\ 0, & if \quad c = False \end{cases} \tag{2.23}$$

The function $\delta(x_{opt} \neq m)$ is introduced to account for the special case: $i = m$. It occurs when the best path to (m, j) comes from $(m, j-1)$. It implies that the crossing path has already ended at previous j-th frame of the test utterance and no more distance accumulation is required. Note that $i = 1$ (as shown in (2.20)) is taken as the starting point with no accumulated distances from past. It only considers the distance for the current frames: $d_{norm}(\mathbf{q}_1, \mathbf{t}_j)$. The detection score is computed as $1 - d_{avg}(\mathbf{Q}, \mathbf{T})$ which ranges from 0 to 1. Here, 1 indicates a perfect match and 0 indicates no match. The frames k_1 and k_2 in the test utterance $\mathbf{T}$ refer to the start and end of the detection hypothesis that matches best with the query $\mathbf{Q}$. If the length of a hypothesis is less than half of the query length, it is discarded since small portions of the test utterance can match well with query segments and produce a high likelihood score. Finally, the score of a hypothesis is compared with a pre-defined threshold to decide the occurrence of the query. A block diagram of this system is presented in Figure 2.3 to find a spoken query in a test utterance.

3 Phonetic Subspace Features for QbE-STD

Contents

This chapter is based on the following paper:

> Dhananjay Ram, Afsaneh Asaei, and Hervé Bourlard. Phonetic subspace features for improved query by example spoken term detection. *Speech Communication*, 103:27–36, 2018a

3.1 Introduction

In the last chapter, we presented the state-of-the-art QbE-STD system consisting of two main stages: (1) extraction of feature vectors from the spoken query and the test utterance, and (2) alignment of the query and test features using DTW. Here we propose to exploit the low-dimensional subspace structure of speech in the first stage to obtain better features before using DTW based template matching. In this way, we develop a framework to utilize the subspace information (in the first stage) as well as temporal information of speech signal (in the second stage using DTW) for query detection. To achieve this goal, we propose a data-driven and a knowledge-based approach to obtain better representation of speech and a fusion technique to combine information from different kinds of representations as discussed below.

(i) *Phone Posteriors (Section 3.2)*: We propose to use sparse modeling as an unsupervised data-driven method to characterize the low-dimensional structures of sub-phonetic components (Elhamifar and Vidal, 2013; Rish and Grabarnik, 2014). To that end, we model the underlying phonetic subspaces using dictionary learning for sparse coding. The dictionaries are used to obtain sparse representation of the phone posteriors and we project them onto the phonetic subspaces through reconstruction. This approach leads to subspace enhanced phone posteriors such that the query and test posteriors are represented on a common subspace and reduces the effect of unstructured phonetic variations.

(ii) *Phonological Posteriors (Section 3.3)*: Alternative to the data-driven sparse modeling approach, we utilize linguistic knowledge for identifying the sub-phonetic attributes or phonological features (Chomsky and Halle, 1968). The phonological features are recognized as the atomic components of phone construction. The linguists define a binary mapping between the phone and phonological categories. We exploit DNN in probabilistic characterization of the phonological features, referred to as the *phonological posteriors* (Cernak et al., 2017). Due to the sub-phonetic nature of these features, they are less language dependent (Lee and Siniscalchi, 2013; Sahraeian et al., 2015) and can be helpful for a zero resource task like QbE-STD.

(iii) *Distance fusion (Section 3.4)*: The proposed representations are exploited for QbE-STD using the DTW method presented in (Rodriguez-Fuentes et al., 2014) (see Section 2.6 for details). To integrate the information from multiple feature representations, we propose to update the distance matrix for DTW by fusing the distances between the query and test utterance obtained from different kinds of feature representations. In contrast to (Wang et al., 2013), we use non-uniform weights which are optimized using development queries.

The proposed methods are evaluated on two subsets of AMI database (IHM and SDM) with challenging conditions as presented in Section 3.6. The improvements obtained by our

approach over the baseline system indicate the significance of subspace structure of speech for QbE-STD.

3.2 Phone Posteriors

In this section, we present a data-driven approach for modeling the underlying low-dimensional subspaces which constitute different phonetic units in a language. These models are then used to enhance the phone posterior features by reducing the effect of unstructured noise present in the data.

3.2.1 Sparse Subspace Modeling

We consider sparse coding as a data-driven unsupervised technique to characterize the subspace structure of phone posteriors. Earlier studies have shown that the phone posterior vectors belong to a union of low-dimensional subspaces (Ram et al., 2015, 2016, 2018b; Dighe et al., 2016a). Any data point in these subspaces can be efficiently reconstructed using a sparse linear combination of other points in that space. This property is referred to as the self-expressiveness (Elhamifar and Vidal, 2013) of data. In practice, an over-complete set of basis vectors, called dictionary is learned from the training data to model the underlying subspaces. It is learned in a manner such that each training vector can be reconstructed as a sparse linear combination of its columns. The columns of a dictionary are known as the atoms forming the molecular structure of the phone posteriors.

Formally speaking, any data point $\mathbf{y}_t$ belonging to the space of phone k can be expressed as a sparse linear combination of the atoms present in the corresponding dictionary as $\mathbf{y}_t = \mathbf{D}_k \boldsymbol{\alpha}_t$ where $\mathbf{D}_k \in \mathbb{R}^{K \times M_k}$ consists of the over-complete basis vectors (atoms) used to model the subspaces of phone k and $\boldsymbol{\alpha}_t$ is the sparse weight vector indicating the significance of each atom to construct the posterior vector. Here, M_k is the number of atoms in the k-th dictionary and K is the dimension of each atom as well as the number of phone classes.

In order to obtain a sparse representation of a posterior vector, we require dictionaries modeling the underlying subspaces. For this purpose, we learn phone-specific dictionaries using the training posteriors. The data used to train the DNN for phone posterior estimation are used here (after forward passing through the DNN) to train these dictionaries. Let us consider a set of T_k training posterior vectors, $\mathbf{Y}_k = \{\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_{T_k}\}$ belonging to phone class k. Their sparse representations are denoted by $\mathbf{A}_k = \{\boldsymbol{\alpha}_1, \boldsymbol{\alpha}_2, \ldots, \boldsymbol{\alpha}_{T_k}\}$. The objective function for dictionary learning is expressed as

$$\mathbf{D}_k = \arg\min_{\mathbf{D}, \mathbf{A}_k} \frac{1}{T_k} \sum_{t=1}^{T_k} \left(\frac{1}{2} \left\| \mathbf{y}_t - \mathbf{D}\,\boldsymbol{\alpha}_t \right\|_2^2 + \lambda \|\boldsymbol{\alpha}_t\|_1 \right) \tag{3.1}$$

where λ is the regularization parameter. The first term in this expression represents the reconstruction error. The second term denotes the ℓ_1-norm of $\boldsymbol{\alpha}$ defined as $\|\boldsymbol{\alpha}\|_1 = \sum_i |\alpha_i|$

Algorithm 1 Dictionary Learning for Sparse Modeling of Phone k

Require: : $\mathbf{Y} = \{\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_{T_k}\}, \lambda, \mathbf{D}^{(0)}$ (initialization)

1: **for** $t = 1$ `to` T_k **do**

2: Sparse representation of $\mathbf{y}_t$ to determine $\boldsymbol{\alpha}_t$:
$\boldsymbol{\alpha}_t = \arg\min_{\boldsymbol{\alpha}} \left\{ \frac{1}{2} \left\| \mathbf{y}_t - \mathbf{D}^{(t-1)} \boldsymbol{\alpha} \right\|_2^2 + \lambda \|\boldsymbol{\alpha}\|_1 \right\}$

3: Updating $\mathbf{D}^{(t)}$ with $\mathbf{D}^{(t-1)}$ as warm restart:
$\mathbf{D}^{(t)} = \arg\min_{\mathbf{D}} \left\{ \frac{1}{t} \sum_{i=1}^{t} (\frac{1}{2} \left\| \mathbf{y}_i - \mathbf{D}\boldsymbol{\alpha}_i \right\|_2^2 + \lambda \|\boldsymbol{\alpha}_i\|_1) \right\}$

4: **end for**

5: **return** $\mathbf{D}_k = \mathbf{D}^{(T_k)}$

which quantifies the level of sparsity of $\boldsymbol{\alpha}_t$. The joint optimization of this objective function with respect to both $\mathbf{D}$ and $\mathbf{A}_k$ simultaneously is non-convex, however it can be solved as a convex objective by optimizing for one while keeping the other fixed (Mairal et al., 2010).

In this work, we have used the fast online algorithm proposed in (Mairal et al., 2010) which was found to be effective for sparse modeling of the phone posterior (Dighe et al., 2016a; Ram et al., 2016). This algorithm is based on stochastic gradient descent optimization and is summarized in Algorithm 1; it alternates between a step of sparse representation for the current training feature $\mathbf{y}_t$ and then optimizes the previous estimate of dictionary $\mathbf{D}^{(t-1)}$ to determine the new estimate $\mathbf{D}^{(t)}$ using stochastic gradient descent.

To learn the phone-specific subspaces, individual dictionaries are learned for each phonetic class separately using the corresponding training data. These phone-specific dictionaries are then used to construct the subspace enhanced posteriors based on the procedure explained in the following section.

3.2.2 Subspace Enhanced Phone Posteriors

The dictionaries learned for sparse modeling characterize the phonetic subspaces using a large amount of training posteriors. Sparse representation of posterior vectors obtained using those dictionaries enables projection of the posteriors to the space of training data. This projection enhances the posterior vectors by better matching the structure of phonetic subspaces modeled using training data and has been successfully used in speech recognition (Sainath et al., 2011; Dighe et al., 2016c). In this work, we use a similar approach to enhance phone posterior and use it for the task of QbE-STD.

To enhance the phone posteriors, we first compute sparse representation of a posterior vector using the dictionaries. This sparse representation is then used to obtain the enhanced posterior by multiplying it with the corresponding dictionary. The process of obtaining the sparse representation is different for training and test posteriors because the phonetic class is known for training posteriors, whereas it is unknown for test posteriors.

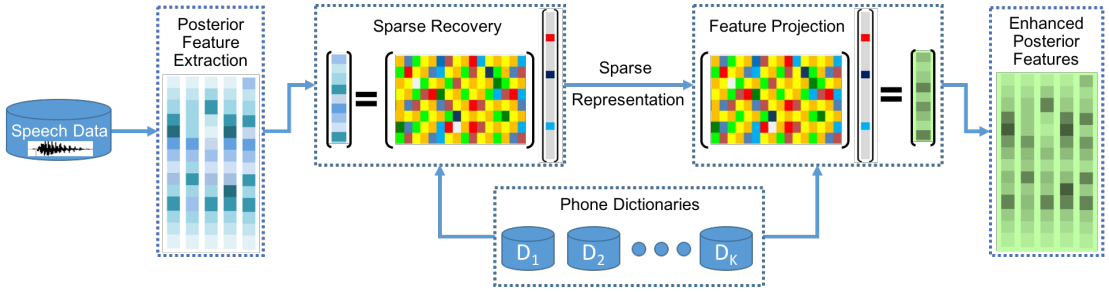

Figure 3.1: Block diagram to obtain subspace enhanced phone posteriors using sparse representation based reconstruction: Phone posterior features are extracted from speech signal. Sparse representations for these feature vectors are obtained using phone dictionaries. These sparse representations are then multiplied to the corresponding dictionary matrix in order to compute the enhanced phone posteriors.

1. *Enhancement of Training Posteriors*: Given a training posterior vector $\mathbf{x}_t$ from phone class k and a dictionary $\mathbf{D}_k$, we can obtain its sparse representation by solving the following optimization problem (Tibshirani, 1996),

$$\boldsymbol{\alpha}_t = \arg\min_{\boldsymbol{\alpha}} \left\{ \frac{1}{2} \|\mathbf{x}_t - \mathbf{D}_k \boldsymbol{\alpha}\|_2^2 + \lambda \|\boldsymbol{\alpha}\|_1 \right\} \tag{3.2}$$

   The enhanced posterior vector is computed by multiplying the sparse representation with the corresponding dictionary matrix as $\mathbf{D}_k \boldsymbol{\alpha}_t$.

2. *Enhancement of Test Posteriors*: In case of test posteriors, the underlying phonetic class k is unknown. Thus, a global dictionary is constructed by concatenating the individual phone dictionaries as $\mathfrak{D} = [\mathbf{D}_1 \quad \mathbf{D}_2 \quad \ldots \quad \mathbf{D}_K]$. Due to the partitioning in construction of $\mathfrak{D}$, there is an inherent block structure underlying the space of $\mathfrak{D}$. This structure is exploited using group sparse recovery algorithm where block sparsity is encouraged during the optimization (Sprechmann et al., 2011). Sparse representation of a test posterior $\mathbf{x}_t$ is then obtained by solving the following optimization problem,

$$\boldsymbol{\beta}_t = \arg\min_{\boldsymbol{\beta}} \left\{ \frac{1}{2} \left\|\mathbf{x}_t - \mathfrak{D}\boldsymbol{\beta}\right\|_2^2 + \lambda f_{\mathfrak{D}}(\boldsymbol{\beta}) \right\} \tag{3.3}$$

   where $f_{\mathfrak{D}}$ is the Group-Lasso regularizer (Sprechmann et al., 2011) defined as: $f_{\mathfrak{D}} = \sum_{i=1}^{K} \left\|\boldsymbol{\beta}_{\{\mathbf{D}_i\}}\right\|_2$ and $\boldsymbol{\beta}_{\{\mathbf{D}_i\}}$ indicates the sparse representation coefficients corresponding to a sub-dictionary $\mathbf{D}_i$ inside $\mathfrak{D}$. The function $f_{\mathfrak{D}}$ can be interpreted as a generalization of the ℓ_1 regularization used in (3.1) and (3.2), where each atom of the dictionary is considered to be a sub-dictionary. The group sparsity regularizer, $f_{\mathfrak{D}}$ forces the atoms of the dictionary to be activated in groups.

   The quantity $\mathfrak{D}\boldsymbol{\beta}_t$ yields a projection of the test posterior onto the phonetic subspace using sparse representation. The posterior vectors thus share common subspaces, which mitigates the effect of unstructured noise (Dighe et al., 2016c). The posteriors of

test utterances used as search space for QbE-STD experiments are enhanced using the technique discussed above. A block diagram to obtain the subspace enhanced phone posteriors is depicted in Figure 3.1. We study the effectiveness of this approach for QbE-STD in Section 3.6.

3.3 Phonological Posteriors

In this section, we describe a linguistic knowledge-based approach to identify the sub-phonetic attributes composing different phonetic units. We further present a setup to exploit this knowledge for extracting new feature vectors (other than phone posteriors) to be used for QbE-STD.

3.3.1 Phonological Subspaces

The linguistic theory states that the elements of phonological structures can be represented as a vector of sub-phonetic, binary-valued attributes (Chomsky and Halle, 1968; Clements, 1985). Each phone attribute represents the minimal distinction between groups of phonemes that share some set of articulatory, perceptual, and/or acoustic properties. The attributes thus cover both articulator-free and articulator-bound distinctions.

Articulator-free attributes describe the high-level properties that can be used to specify the broad classes of speech sounds. These attributes determine the details of the sub-phonetic variations, including whether a periodic source is used and whether nasal effect is occurred. For instance, the articulator-free attribute [son] distinguishes sonorant sounds ([+son]), produced with a largely open vocal tract (e.g. vowels), from obstruent sounds such as fricatives, produced with a tract constriction.

Articulator-bound attributes describe the articulatory configurations of the vocal tract. For instance, the feature [high] distinguishes between those vowels produced with the tongue close to the roof of the mouth (e.g., the /t/ in beat) from those that are not.

The articulator-bound and articulator-free sub-phonetic attributes possess a natural hierarchical structure (McCarthy, 1988). Articulator-free distinctions are more perceptually salient and their acoustic correlates are less context dependent whereas the articulator-bound features make finer distinctions between individual or small groups of phonemes (Jansen and Niyogi, 2013). Every component of the phonological features characterizes a subspace such that the phonemes are formed through the composition of the underlying phonological components (Cernak et al., 2016, 2017; Asaei et al., 2017). Here, we use DNNs to generate a probabilistic representation of the phonological features, as described in the following section.

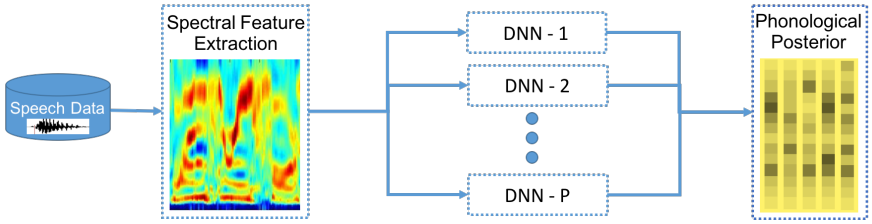

Figure 3.2: Phonological posterior estimation using a bank of deep neural networks (DNN): First, MFCC based spectral features are extracted from windowed speech. These features are then fed with some acoustic context (left and right) to each DNN modeling a specific phonological class to estimate the corresponding class conditional posterior probabilities. These probabilities are concatenated to form posterior feature vector.

3.3.2 Phonological Posterior Estimation

A vector representing the class conditional posterior probabilities of all phonological classes is referred to as phonological posteriors. They are considered to be capable of providing a language independent representation of speech, and any sound can be decomposed into a subset of phonological classes. In contrast, for sparse modeling approach, there is no linguistic knowledge guiding the discovery of the underlying subspaces. Hence, both representations may bear complimentary information on the sub-phonetic structure of the speech representation.

The setup to extract phonological posteriors (Cernak et al., 2016, 2017) from speech signal is depicted in Figure 3.2, which is similar to the phone posterior estimation method as described in Section 2.3.2. In contrast to the system presented in Figure 2.1, one DNN is trained per phonological class (see Section 3.5.3 for more details). This is due to the fact that multiple phonological classes can be active per speech frame, unlike the case of phone posteriors where only one class is active per frame. Once the DNNs are trained, MFCC features are forward passed through each of them to estimate the corresponding phonological class probability. These probabilities are then concatenated to obtain the phonological posterior vectors.

So, we have presented two different approaches to capture the subspace information of speech data in Sections 3.2 and 3.3 respectively. Our first approach is aimed at enhancing the phone posteriors using the subspace information from a data-driven method. In a similar manner, we need a method to integrate the information obtained from speech data using phonological posteriors with that of the phone posteriors. Hence, we propose a new approach to fuse information from different feature representations of same speech signal as described in the following section.

3.4 Distance Fusion

In this section, our goal is to develop a mechanism to integrate information from multiple feature representations of speech data for spoken query detection which we use to fuse the phone and phonological posteriors. Different types of feature vectors have different

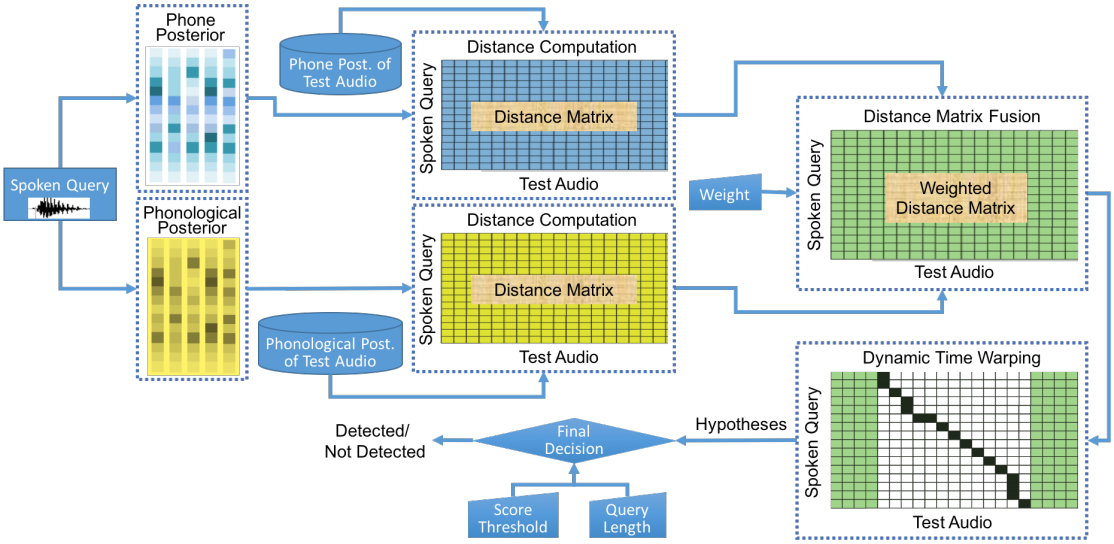

Figure 3.3: Block diagram of distance fusion for DTW system using phone and phonological posteriors: Both phone and phonological posterior based representations are obtained for spoken query and test utterance to compute corresponding normalized distance matrices. We fuse these distance matrices by taking a weighted combination of each element from corresponding distance matrices. DTW algorithm (Rodriguez-Fuentes et al., 2014) is then used to obtain a hypothesized sub-sequence matching the query. A hypothesis of length less than half the length of the query is discarded to reduce false alarm. Finally, the score of a valid hypothesis is compared to a threshold to yield a decision.

dimensions and represent different characteristics of speech, making it difficult to fuse them in the domain of feature vectors. Instead, we use the features independently to construct a distance matrix for DTW and fuse these matrices into a single distance matrix which can be used for query detection using the DTW algorithm as explained in Section 2.6.4.

More formally, let $\mathbf{U}^s = \left[\mathbf{u}_1^s, \mathbf{u}_2^s, \ldots, \mathbf{u}_m^s\right]$ denote the feature vectors extracted from a spoken query and $\mathbf{V}^s = \left[\mathbf{v}_1^s, \mathbf{v}_2^s, \ldots, \mathbf{v}_n^s\right]$ the feature vectors of a test utterance; m and n represents the number of frames in the spoken query and test utterance respectively, and $s = 1, 2, \ldots, S$ indicates the source of feature vector where S is the number of different types of features extracted from the same speech. The pairwise distances of any two feature vectors can be calculated using a distance measure:

$$\begin{aligned} \delta_s(i,j) &= \text{distance}\left(\mathbf{u}_i^s, \mathbf{v}_j^s\right) \\ &\forall\ s = 1,2,\ldots,S;\ i = 1,2,\ldots,m \text{ and } j = 1,2,\ldots,n \end{aligned} \tag{3.4}$$

The distance function can be chosen to best match the properties of corresponding feature vectors. In this paper, we consider logarithm of cosine distance as the distance function for its superior performance in posterior-based query detection (Rodriguez-Fuentes et al., 2014). A simple range normalization technique (as shown in (2.17)) is used to have the distance

values between 0 and 1. We fuse the distance matrices (δ_s) for S different sources of feature representations by taking a weighted combination as follows:

$$\Delta(i,j) = \sum_{s=1}^{S} w_s \times \delta_s(i,j) \quad \text{s.t.} \quad \sum_{s=1}^{S} w_s = 1 \\ \forall\ i = 1,2,\ldots,m \text{ and } j = 1,2,\ldots,n \tag{3.5}$$

where the weight parameters w_s are optimized over an independent development set. The weight associated to a feature representation indicates its importance in conjunction with the rest of feature vectors being fused for a particular task (see Section 3.6.3 for our experimental analysis on the choice of w_s). After fusion, we simply follow the process described in Section 2.6.4 to generate QbE-STD hypotheses.

In this work we consider two types of features, namely phone posterior and phonological posterior and show that the fusion technique discussed here improves the detection performance (see Table 3.4 and, Figures 3.4 and 3.5). The block diagram of the system integrating these features (phone and phonological posteriors) is presented in Figure 3.3. Note that the distance fusion method is independent of the type of features and distance measure being used, and it is applicable to other DTW based pattern matching.

3.5 Experimental Setup

We use two subsets of AMI meeting corpus: IHM and SDM (as described in Section 2.4) to perform QbE-STD experiments. In this section, we present a brief description of the query selection process, the setup used for extracting phone and phonological posteriors and the pre-processing steps involved to perform the experiments. Finally, we describe the score normalization and evaluation metrics used to compare different systems proposed in this work.

3.5.1 Query Selection

We select different words from the database to construct the query set for our QbE-STD experiments. We use two different strategies to extract queries for experiments on IHM and SDM set. For experiments on IHM, we extract 200 more frequent words (excluding functional words) including very short words such as 'BUY' to long words such as 'TELEVISION'. In case of SDM queries, we compute term frequency-inverse document frequency (TF-IDF) statistic for all words in the dataset, which indicates the importance of a word to a document in a collection of documents. We consider each meeting recording as a document for computing TF-IDF statistic. We arrange these words with decreasing TF-IDF values and choose top 200 words giving us important, content bearing words. The spoken examples corresponding to these words are extracted from randomly chosen utterances in the training data. These queries are divided into two sets of 100 queries each. These sets become our development and

evaluation query set. The development queries are used to optimize parameters for different systems. The test set of each dataset is used as the search space for queries from corresponding dataset.

3.5.2 Phone Posterior Estimation

We train two different DNNs for IHM and SDM datasets to estimate the corresponding phone posterior vectors. The phone posterior features are estimated from a DNN with MFCC based spectral features as input. These MFCC features are extracted form speech data by following the steps described in Section 2.3.1. To add contextual information, 4 frames of left and right acoustic contexts are appended (total 9 frames) to have a 351 dimensional input vector to the DNN. The DNN consists of 3 hidden layers of 1024 neurons each, to estimate 43 dimensional phone posteriors at the output. There are 39 phones obtained from CMU pronunciation dictionary[1] for lexical modeling. The remaining 4 phones are used to model silence and non-speech sounds. The training labels for the DNN are generated using a GMM-HMM based speech recognizer (Hinton et al., 2012). The recognizer is used to force align the training data to obtain the corresponding phonetic transcription. This whole setup is implemented using the Kaldi toolkit (Povey et al., 2011).

3.5.3 Phonological Posterior Estimation

We have used the open-source phonological vocoding platform presented in (Cernak et al., 2017) to obtain the phonological posteriors. It uses the phonological system of extended Sound Pattern of English (eSPE) for phonological representation. The phonological classes in eSPE are: vowel, fricative, nasal, stop, approximant, coronal, high, dental, glottal, labial, low, mid, retroflex, velar, anterior, back, continuant, round, tense, voiced, silence. The vocoding platform has 21 DNNs corresponding to each phonological class including one class for silence. Each DNN consists of 3 hidden layers of 1024 neurons each. These DNNs were trained on Wall Street Journal (WSJ0 and WSJ1) continuous speech recognition database (Paul and Baker, 1992). The input to the DNNs are MFCC features with a context of 4 frames (both left and right) giving us a 351 dimensional input vector. The output consists of 2 labels indicating whether the phonological class occurs for the segment or not. In other words, each DNN performs binary classification of the target class vs the rest. All the DNNs were randomly initialized and were trained by minimizing cross-entropy loss. The output probabilities from all DNNs are concatenated to form the phonological posterior feature used for QbE-STD.

3.5.4 Speech Activity Detection (SAD)

We have implemented the speech activity detection setup presented in Section 2.6.2 to remove the noisy frames from test utterances as well as queries to perform detection experiments.

[1] http://www.speech.cs.cmu.edu/cgi-bin/cmudict

The frame-level posterior probabilities of silence and non-speech sounds from IHM dataset are used to perform SAD for all the experiments with both datasets.

3.5.5 Score Normalization

We use the posterior features to perform query detection experiment and obtain a likelihood score for each pair of spoken query and test utterance. Following the score normalization technique used in (Rodriguez-Fuentes and Penagarikano, 2013), we normalize the scores to have zero-mean and unit-variance per query. This reduces the variability in scores across different queries and makes them comparable for final evaluation. We have also tried sum-to-1 (STO) normalization and keyword-specific thresholds (KST) (Karakos et al., 2013; Wang and Metze, 2014). However they did not perform better than the mean-variance normalization. Thus the results presented in this work utilize the mean-variance normalization.

3.5.6 Evaluation Metric

We use $MTWV$, C_{nxe}^{min} and DET curves (as described in Section 2.5) to evaluate and compare the performance of different systems. $MTWV$ takes into account the prior probability of query occurrence in the test set as well as the costs of missed detection and false alarm. We consider the cost of missed detection (C_{miss}) to be 100 and the cost of false alarm (C_{fa}) to be 1 for our experiments. We also perform statistical significance test to measure the improvements in $MTWV$ and C_{nxe}^{min} and the corresponding p-values are indicated with the results.

3.6 Experimental Analysis

This section describes different QbE-STD experiments conducted to analyze and evaluate the performance of the proposed approaches exploiting information from various phonetic subspace representations. In all experiments, only one spoken instance of each query is provided, and the test utterances are conversational speech produced by competing speakers.

3.6.1 Phone and Phonological Posteriors

We have used the QbE-STD system discussed in Section 2.6 as our baseline system. We use both phone posteriors and phonological posteriors as feature representation for template matching. This work is the first attempt at using phonological posteriors for QbE-STD. The detection performance using development queries of IHM and SDM dataset is presented in Table 3.1. Clearly, the phonological posteriors perform worse than the phone posteriors for both datasets in a stand-alone system. This can be attributed to the shared sub-phonetic properties of different phonemes. However, we expect that they bear complementary sub-phonetic information guided by the knowledge of linguistics. Hence, we study the performance of the detection system where both features are integrated using the distance fusion technique

Table 3.1: Performance of the DTW based QbE-STD system using phone and phonological posteriors as feature vectors evaluated on the development queries of IHM and SDM dataset from AMI corpus.

Feature Representation	IHM		SDM	
	$MTWV \uparrow$	$C_{nxe}^{min} \downarrow$	$MTWV \uparrow$	$C_{nxe}^{min} \downarrow$
Phone Posterior	0.4646	0.6558	0.1976	0.8083
Phonological Posterior	0.3105	0.8118	0.0276	0.9118

presented in Section 3.4.

3.6.2 Subspace Enhanced Phone Posteriors

We evaluate the data-driven approach to enhance phone posteriors using the subspace structure of speech as discussed in Section 3.2. The phonetic subspaces are characterized via dictionary learning for sparse representation. We use Algorithm 1 to learn phone-specific subspaces (dictionaries) from corresponding training phone posteriors. For this purpose, we collect all posterior vectors corresponding to a phonetic class, and randomly choose 50 posteriors to initialize the dictionary. Rest of the posteriors are used to train the dictionary.

The posteriors from training data are enhanced using the sparse representation obtained from (3.2). The enhancement is achieved by multiplying the sparse representation coefficients with the corresponding dictionary. On the other hand, to enhance the test posteriors, all the dictionaries are concatenated to form a single dictionary of all phones. This dictionary is then used to obtain sparse representation of test posteriors using (3.3). The enhanced posterior is obtained by multiplying the sparse representation with the corresponding dictionary as discussed in Section 3.2.2.

To evaluate this approach, the regularization parameter λ is tuned on the development set. It controls the number of dictionary atoms (sub-phonetic components) in subspace reconstruction, and its optimal value depends on the size of the dictionary. To evaluate the sensitivity of the QbE-STD system to λ, we compare the $MTWV$ and C_{nxe}^{min} scores corresponding to different values measured on the development queries. The results for IHM and SDM datasets are presented in Table 3.2. We can see that $\lambda = 0.2$ and $\lambda = 0.1$ yields the best performance for IHM and SDM set respectively, so these values are chosen for the corresponding experiments using evaluation queries.

3.6.3 Distance Fusion Performance

We analyze the effectiveness of different feature vectors for QbE-STD using the distance fusion technique presented in Section 3.4. We performed two different experiments for both IHM

Table 3.2: Variation of $MTWV$ and C_{nxe}^{min} for Subspce Enhanced Phone Posterior features with varying regularizarion, λ evaluated on development queries of IHM and SDM dataset from AMI corpus

λ	IHM		SDM	
	$MTWV \uparrow$	$C_{nxe}^{min} \downarrow$	$MTWV \uparrow$	$C_{nxe}^{min} \downarrow$
0.05	0.4399	0.6827	0.1923	0.8035
0.1	0.4633	0.6712	**0.2497**	**0.7726**
0.2	**0.4705**	**0.6653**	0.2439	0.7910
0.3	0.4563	0.6812	0.2054	0.8094

and SDM datasets to integrate subspace information of speech data presented in the form of phonological posteriors. The first one (Phone + Phonological) integrates phone posteriors with phonological posteriors whereas the second one (Enhanced Phone + Phonological) combines enhanced phone posteriors (data-driven) with phonological posteriors (knowledge-based). In both cases, we construct distance matrices between the query and test utterance using corresponding feature representation. The two matrices are then fused to form a single distance matrix following (3.5), and is used to perform DTW to detect queries.

In both experiments we merge two matrices, resulting in weights w and $(1-w)$ corresponding to (enhanced) phone and phonological posterior respectively. These weights indicate the significance of the corresponding feature vectors for the detection stage. The results using the development queries of the IHM dataset for different weights are shown in Table 4.4 as an example. The optimal value of $w = 0.6$ for 'Phone + Phonological' indicates a contribution of 0.6 and 0.4 for phone and phonological posteriors respectively to attain the best performance. In 'Enhanced Phone + Phonological', the optimal value of $w = 0.5$ shows equal contribution of enhanced phone and phonological posteriors for best performance. The optimized weight for SDM set is $w = 0.8$ for both 'Phone + Phonological' and 'Enhanced Phone + Phonological' cases. The higher weight indicates smaller contribution from phonological posteriors compared to IHM set. It can be attributed to the worse performance of the phonological posteriors in a stand alone system for SDM set compared to IHM set as indicated in Table 3.1. These optimized weights are used for final assessment on evaluation queries as presented in the following section. As part of our experiments, we have also tried a score fusion technique (Brümmer and De Villiers, 2013) as used in (Rodriguez-Fuentes et al., 2014). However, this resulted in worse QbE-STD performance compared to the baseline system using phone posteriors.

3.6.4 Performance Comparison

In this section, we present the performance of different systems (optimized using development queries) on evaluation queries. The $MTWV$ and C_{nxe}^{min} scores are presented in Table 3.4 for both IHM and SDM datasets, whereas the corresponding DET curves are shown in Figures 3.4

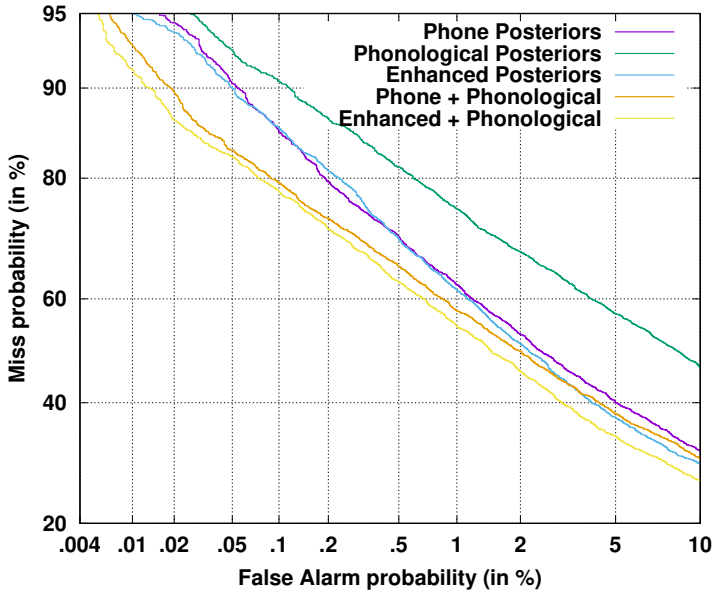

Figure 3.4: DET curves comparing the performance of different posterior features (phone and phonological) for QbE-STD, evaluated on IHM set of AMI corpus.

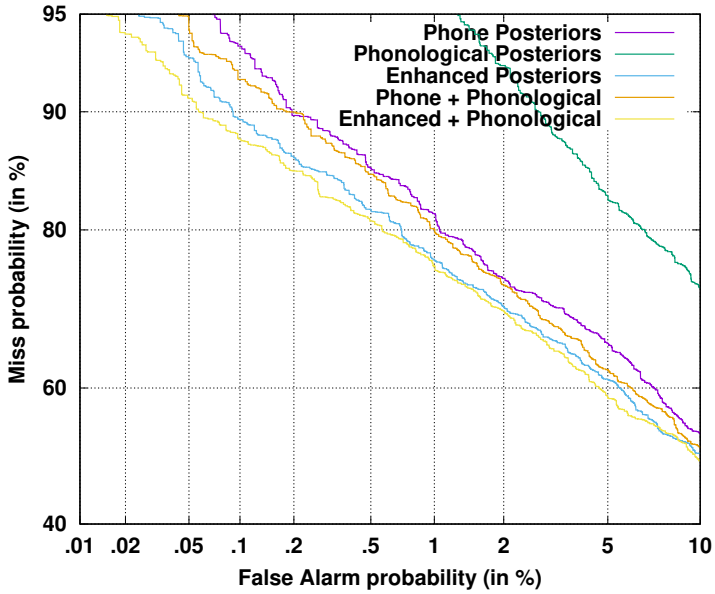

Figure 3.5: DET curves comparing the performance of different posterior features (phone and phonological) for QbE-STD, evaluated on SDM set of AMI corpus.

Table 3.3: Variation of $MTWV$ and C_{nxe}^{min} for fusion of two sets of feature representations (Phone+Phonological and Enhanced Phone+Phonological) evaluated on development queries of the IHM set from AMI corpus. The optimal weight indicates the contribution of corresponding feature representation to attain the best performance.

Fusion Weights	Phone + Phonological		Enhanced Phone + Phonological	
	$MTWV \uparrow$	$C_{nxe}^{min} \downarrow$	$MTWV \uparrow$	$C_{nxe}^{min} \downarrow$
0.4	0.4343	0.6896	0.5197	0.6322
0.5	0.4916	0.6358	**0.5270**	**0.6217**
0.6	**0.4988**	**0.6307**	0.5225	0.6232
0.7	0.4982	0.6323	0.5243	0.6280
0.8	0.4880	0.6359	0.5135	0.6357

Table 3.4: Performance comparison of the DTW based QbE-STD system using different posteriors as feature representations computed on the evaluation queries. Enhanced phone posteriors improve the QbE-STD performance. Also, distance fusion technique is effective in integration of the multiple sources of information through different set of features

Feature Representation	IHM		SDM	
	$MTWV \uparrow$	$C_{nxe}^{min} \downarrow$	$MTWV \uparrow$	$C_{nxe}^{min} \downarrow$
Phone Posterior	0.4758	0.6526	0.2319	0.8398
Phonological Posterior	0.3044	0.7780	0.0459	0.8924
Enhanced Phone Posterior	0.5052*	0.6136*	0.2774*	0.8202*
Phone + Phonological Posterior	0.4969*	0.6287*	0.2555*	0.8288*
Enhanced Phone + Phonological Posterior	**0.5414***	**0.6051***	**0.2944***	**0.8102***

* significant at $p < 0.001$

and 3.5 respectively to indicate the miss probabilities for a given range of false alarm probabilities. The performance using phone posteriors is our baseline system and it gets increasingly better using the subspace enhanced posteriors and distance fusion techniques. It can be observed that the performance of both phone posteriors and enhanced phone posteriors improves while a distance fusion is performed with phonological posteriors. This indicates that, both phone posteriors and enhanced phone posteriors are not able to capture all information present in the speech signal and phonological posteriors are one way of capturing finer details of sub-phonetic components.

3.7 Conclusions

QbE-STD benefits from the advanced features capturing the subspace structure of the speech signal. We exploit the low-dimensional subspace structures of speech through a data-driven and a knowledge-based approach. The data-driven approach relies on sparse modeling of

phonetic posteriors to characterize the sub-phonetic components in an unsupervised manner and we use these models to enhance the phone posteriors. The knowledge-based approach utilizes linguistic information to identify the sub-phonetic units and represent them using phonological posteriors. To integrate information from multiple representations of speech, a distance fusion technique is proposed. We show that the phone posteriors and phonological posteriors represent complementary information to improve the performance of the QbE-STD system. The QbE-STD solution developed in this paper makes no assumption on the underlying language, thus it is applicable to multilingual scenarios involving low-resource or zero-resource languages.

4 Sparse Subspace Modeling for QbE-STD

Contents

This chapter is based on the following papers:

Dhananjay Ram, Afsaneh Asaei, and Hervé Bourlard. Subspace regularized dynamic time warping for spoken query detection. In *Workshop on Signal Processing with Adaptive Sparse Structured Representations (SPARS)*, 2017

Dhananjay Ram, Afsaneh Asaei, and Hervé Bourlard. Sparse subspace modeling for query by example spoken term detection. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 26(6):1126–1139, June 2018b. ISSN 2329-9290. doi: 10.1109/TASLP.2018.2815780

4.1 Introduction

In the last chapter, we exploited the low-dimensional subspace structure of the speech signal to obtain better representation of the query and test utterances for QbE-STD. Here we focus on utilizing the same properties in the matching algorithm to improve the likelihood scores. The present work is motivated by the success of exemplar-based sparse representation in detection and classification tasks (Wright et al., 2009; Chen et al., 2011). Previously, (Ram et al., 2016) cast the query detection problem as subspace detection between query and non-query speech where the corresponding subspaces are modeled through dictionary learning for sparse representation. Given these dictionaries, detection of each frame is performed based on the ratio of the two corresponding sparse representation reconstruction errors, and the frame-level decisions are accumulated by counting the continuous number of frames detected as the query. Although this approach shows a promising direction, it lacks a proper framework to capture the temporal information inherent to speech signal. Also, it relies on the background dictionaries to model non-query speech which is usually not available for QbE-STD.

Building upon the above discussion, this work explores new systems designed to take advantage of both *temporal information* and *subspace structure* of speech. The primary contribution of this chapter is to show the effectiveness of subspace structure of speech data for finding a spoken query in a test utterance. In this context, a query is modeled through dictionary which can be built from single as well as multiple query examples. We present three different ways to achieve our goal, as discussed in the following:

1. *Sparse Subspace Detection (Section 4.3)*: This approach relies on modeling the low-dimensional structure of sub-phonetic components of the query. These subspaces are modeled using dictionaries for sparse coding. The dictionaries are used to obtain a frame level sparse representation which quantifies the errors to reconstruct those frames. We propose to use a dynamic programming technique to obtain possible regions of occurrence of the query in test utterances. This reduces the effect of errors made by the sparse coding algorithm on frame level and captures the sequential information present in the data. It is a much faster technique compared to DTW based methods.

2. *Subspace Regularized DTW (Section 4.4.1)*: In this approach, we use both sparsity and DTW to make a better system, instead of relying solely on either of them to perform the same task. The idea is to consider the frame-level reconstruction errors as subspace based distances. We propose to regularize the distance matrix for DTW using this subspace based distance and perform DTW to detect the query. This regularization helps to take into account the temporal information as well as the subspace structure of speech signal.

3. *Subspace-Based Rescoring of DTW (Section 4.4.2)*: In another approach, we propose to rescore the hypothesized regions obtained from the DTW system using sparsity based

system. This method aims at improving the likelihood score for a hypothesized region using the subspace structure of speech signal.

In all three cases discussed above, we rely on the low-dimensional subspace structure of speech signal for the task of QbE-STD. The systems proposed in this chapter indicate different ways of utilizing this information. These systems are evaluated on two different databases with challenging conditions as we will see in Sections 4.6 and 4.7. The performance improvements provided by the combination of DTW and sparsity based systems show the importance of subspace structure of speech to perform QbE-STD.

4.2 Subspace Modeling

In this section, we describe the modeling of subspaces of query exemplars for sparse representation. We start by describing the sparse representation of posterior feature vectors (as a sparse linear combination of an over-complete dictionary posteriors) before discussing different methods to construct dictionaries for query modeling.

4.2.1 Sparse Representation

When speech is represented in terms of posterior probabilities, the subspace corresponding to each sub-word class is a low-dimensional space (Dighe et al., 2016b; Luyet et al., 2016). Accordingly, a speech utterance comprised of sub-word classes, can be modeled as a union of low-dimensional subspaces. Any data point in a union of low-dimensional subspaces can be efficiently reconstructed by a sparse combination of other points in that space, a property referred to as the self-expressiveness (Elhamifar and Vidal, 2013) of data.

Let $\mathbf{y}_t$ be a posterior feature vector for a speech frame at time t. Each posterior vector $\mathbf{y}_t$ is a K dimensional feature vector where each dimension corresponds to a speech unit. These speech units (associated with DNN outputs) can be phones (context dependent or independent), senones, or any other sub-word unit. Following the self-expressiveness property of data, the feature vector $\mathbf{y}_t$ can be represented as a sparse linear combination of other feature vectors present in the training set, $\{\mathbf{d}_i\}_{i=1}^{N}$ corresponding to the query subspace, i.e.:

$$\begin{aligned}\mathbf{y}_t &\approx \alpha_1\mathbf{d}_1 + \alpha_2\mathbf{d}_2 + \ldots + \alpha_N\mathbf{d}_N \\ &= \underbrace{[\mathbf{d}_1 \quad \mathbf{d}_2 \quad \ldots \quad \mathbf{d}_N]}_{\mathbf{D}} \times \underbrace{[\alpha_1 \quad \alpha_2 \quad \ldots \quad \alpha_N]^\top}_{\boldsymbol{\alpha}_t} \\ &= \mathbf{D}\boldsymbol{\alpha}_t\end{aligned} \tag{4.1}$$

where N is the number of training samples (basis vectors) used to model the query subspace, $\mathbf{D}$ is a matrix of size $K \times N$ which consists of basis vectors $\mathbf{d}_i$ of the query subspace, and $\boldsymbol{\alpha}_t$ is the weight vector indicating the significance of each basis vector in construction of a test posterior. The matrix $\mathbf{D}$ is called a dictionary matrix and the columns of this matrix are called

atoms. The weight vector $\boldsymbol{\alpha}_t$ is sparse, i.e., having very few non-zero entries. The non-zero entries correspond to the underlying low-dimensional subspaces which the test posterior belongs to.

The framework of sparse representation introduced above in (4.1) relies on construction of the dictionary matrix $\mathbf{D}$. Given this dictionary for characterizing the underlying subspaces, the independent subspaces are guaranteed to be identified correctly using sparse representation (Elhamifar and Vidal, 2013). In the following section, we explain how these subspaces can be modeled using dictionary for sparse representation.

4.2.2 Dictionaries for Subspace Modeling

Dictionaries for sparse representation are constructed using an over-complete set of basis vectors obtained from the training examples of corresponding classes. These dictionaries can be modeled primarily in the following two ways:

1. *Concatenation of training examples*: In this method, we take the features of all available training examples for a desired class and concatenate them to build the dictionary (Sainath et al., 2011; Gemmeke et al., 2011). This method is more suitable in a scenario when very few training examples are available for a class. On the other hand, with huge number of training examples, the size of the dictionary can grow very large. This, in turn, can increase the computational complexity of the sparse coding algorithm. A method to extract all the information present in the training data without increasing the size of the dictionary to an undesirable magnitude is thus required.

2. *Learning from training examples*: Dictionary learning refers to the task of learning an over-complete set of basis vectors from the training exemplars such that each training exemplar can be reconstructed as a sparse linear combination of the dictionary vectors (atoms). These dictionaries can be learned by solving an optimization problem which gives the best approximation of training vectors while keeping the degree of sparsity on desirable level as discussed in the following.

   Let us have a set of T training vectors with $\mathbf{Y} = \{\mathbf{y}_1, \mathbf{y}_2, ..., \mathbf{y}_T\}$, and their sparse representations using dictionary $\mathbf{D} \in \mathbb{R}^{K \times M}$ with $\mathbf{A} = \{\boldsymbol{\alpha}_1, \boldsymbol{\alpha}_2, ..., \boldsymbol{\alpha}_T\}$, where K is the dimension of exemplar vectors, and M is the number of dictionary atoms, the objective function for dictionary learning is defined as

$$\arg\min_{\mathbf{D},\mathbf{A}} \frac{1}{T} \sum_{t=1}^{T} \left(\frac{1}{2} \left\| \mathbf{y}_t - \mathbf{D}\,\boldsymbol{\alpha}_t \right\|_2^2 + \lambda \| \boldsymbol{\alpha}_t \|_1 \right) \tag{4.2}$$

   where λ is the regularization parameter. The first term in this expression, quantifies the *reconstruction error*. The second term denotes the ℓ_1-norm of $\boldsymbol{\alpha}$ defined as $\|\boldsymbol{\alpha}\|_1 = \sum_i |\alpha_i|$ which quantifies the sparsity of $\boldsymbol{\alpha}_t$. The joint optimization of this objective function with respect to both $\mathbf{D}$ and $\boldsymbol{\alpha}_t$ simultaneously is non-convex, it can be solved

as a convex objective by optimizing for one while keeping the other fixed (Mairal et al., 2010).

In case of QbE-STD, we represent each query as a class to be modeled using dictionary. The training data for these dictionaries is obtained by extracting posterior features from the spoken instances of corresponding query as discussed in Section 2.3.2. These are the same posterior features used to construct the query templates for DTW in the baseline system. We consider two cases to construct a dictionary depending on the number of examples available for a given query. If there is only one example available per query, the corresponding posterior feature vectors constitute the dictionary. Whereas with multiple examples per query, we either concatenate the posteriors of these examples to construct a dictionary or use these posteriors to learn a dictionary. In case of learning a dictionary, we initialize the dictionary with posteriors of the example having the highest number of frames. Posteriors from rest of the examples are used to learn the dictionary according to (4.2).

4.3 Sparse Subspace Detection (SSD)

Once the query subspaces are modeled, the QbE-STD problem is cast as a subspace detection problem where the reconstruction errors of the respective sparse representations are used to detect the underlying subspaces. Given a test posterior feature vector $\mathbf{y}_t$ and the query dictionary $\mathbf{D}$, the test vector can be represented as a sparse linear combination of dictionary atoms characterizing the query. The sparse representation is obtained by solving the following optimization problem:

$$\boldsymbol{\alpha}_t = \arg\min_{\boldsymbol{\alpha}} \left\{ \frac{1}{2} \left\| \mathbf{y}_t - \mathbf{D}\boldsymbol{\alpha} \right\|_2^2 + \lambda \|\boldsymbol{\alpha}\|_1 \right\} \tag{4.3}$$

where λ is the regularization parameter. The first term in this expression quantifies the *reconstruction error*. The second term denotes the ℓ_1-norm of $\boldsymbol{\alpha}$ defined as: $\|\boldsymbol{\alpha}\|_1 = \sum_i |\alpha_i|$ which quantifies the level of sparsity in the co-efficient vector, $\boldsymbol{\alpha}_t$. In order to exploit the temporal information inherent to speech signal, a sequence of $2c+1$ posterior feature vectors are concatenated to form a contextually rich vector for dictionary construction as well as sparse representation as follows,

$$\tilde{\mathbf{y}}_t = [\mathbf{y}_{t-c}^\top \ldots \mathbf{y}_t^\top \ldots \mathbf{y}_{t+c}^\top]^\top \tag{4.4}$$

This mechanism is referred to as *context appending* which is a typical approach to incorporate the dynamics of exemplars (Gemmeke et al., 2011; Dighe et al., 2016a). This context is a system parameter to be optimized using development queries.

The reconstructed vectors using the corresponding sparse representations will be: $\hat{\mathbf{y}}_t = \mathbf{D}\boldsymbol{\alpha}_t$. The subspace which can best represent a test vector $\mathbf{y}_t$ corresponds to the least reconstruction error (Chen et al., 2011; Ram et al., 2015). Hence, we use the Euclidean-norm based recon-

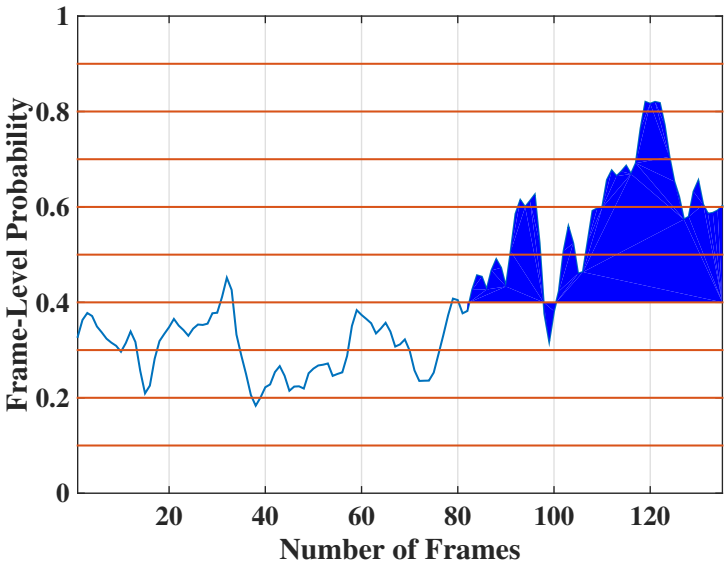

Figure 4.1: Frame-level probability and different thresholds for an utterance containing “SO THAT CONCLUDES MY PRESENTATION". The query speech contains “PRESENTATION"

struction error to perform QbE-STD at a later stage. The reconstruction errors are calculated as follows:

$$e(\mathbf{y}_t) = \frac{1}{2}\left\|\mathbf{y}_t - \hat{\mathbf{y}}_t\right\|_2 = \frac{1}{2}\left\|\mathbf{y}_t - \mathbf{D}\boldsymbol{\alpha}_t\right\|_2 \tag{4.5}$$

We use these reconstruction errors to calculate frame-level empirical probabilities of the query occurring in a test utterance as: $p(\mathbf{y}_t) = 1 - e(\mathbf{y}_t)$. These frame-level probabilities constitute a probability curve indicating the possibility of the query occurring in a test utterance. We perform a non-linear smoothing to compensate for the potential errors made by the sparse coding system. The probability curve for an example utterance is shown in Figure 4.1. In order to identify a hypothesized region of occurrence from this curve, we use Kadane's algorithm (Bentley, 1984), a very simple dynamic programming technique with linear time complexity to obtain a contiguous sub-array within an one-dimensional array of numbers which has the largest sum. In our case, we subtract a threshold value from the probability curve to get an array of numbers. This threshold provides a trade-off between the missed detection and false alarm rate. Subsequently, we apply Kadane's algorithm to obtain a sub-array with the largest sum, which essentially indicates the hypothesized region. The area under this segment of the curve represents the likelihood score. We normalize this score with the length of the hypothesized region. Later, we compare the length of the hypothesized region with half the length of the query and reject the ones having a smaller length in order to reduce false alarms. A comprehensive block diagram for the proposed system is presented in Figure 4.2. The steps to implement the system is summarized in Algorithm 2.

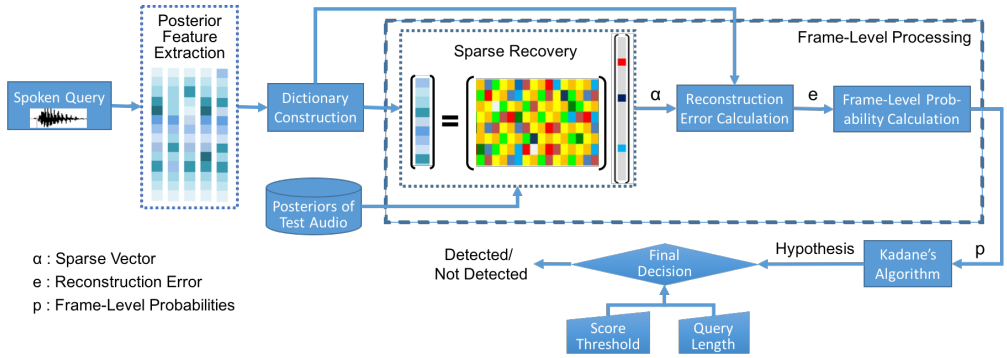

Figure 4.2: Block diagram of the sparse subspace detection system. We first extract posterior features from the query and use it to construct a dictionary. We employ this dictionary to compute the sparse representation and corresponding reconstruction error for each frame of test audio. Exploiting these reconstruction errors, we use Kadane's algorithm (Bentley, 1984) to hypothesize the region of occurrence of the query and calculate the likelihood score. If the length of the hypothesis is smaller than half the query length, it is discarded to reduce false alarm rate. Otherwise, the hypothesis score is compared to a threshold to yield a final decision.

Algorithm 2 Sparse Subspace Detection (SSD) (Fig. 4.2)

Input: Spoken query and posteriors of a test utterance
Output: Decision if the query occurs in the test utterance

1: Extract the posterior features from spoken query
2: Perform context appending according to (4.4) for both query and test posteriors
3: Construct a dictionary by concatenating the posteriors from different examples or learn a dictionary using (4.2)
4: Compute sparse representation of test posteriors using the dictionary according to (4.3)
5: Compute reconstruction error using (4.5)
6: Use Kadane's algorithm to find out a hypothesized region and corresponding score
7: Use query length and score threshold to make a final decision

The QbE-STD system developed in this section does not use DTW to find a query speech in a test utterance. However, the proposed system is not able to capture the temporal information so well as compared to a DTW based system as we will see in Section 4.6. Thus, we propose new approaches to build hybrid systems in the following section which will be able to combine the positive aspects of both systems.

4.4 Sparse-DTW Hybrid Systems

In this section we propose two different ways to incorporate information coming from a DTW system and the sparsity based system discussed above. The first method implements a system-level fusion, whereas the second method performs a re-scoring of the hypotheses from the DTW system using sparsity. We describe these systems in the following.

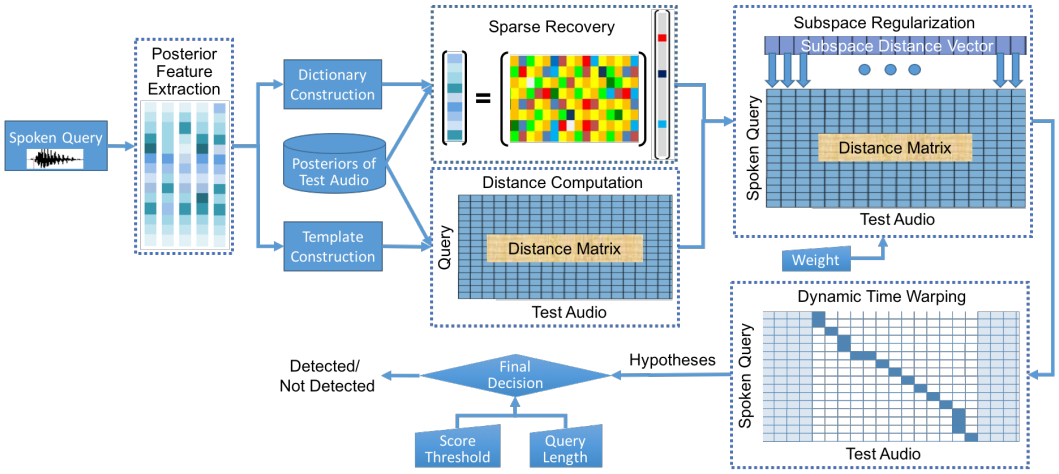

Figure 4.3: Block diagram of the Subspace Regularized DTW system. We first extract posterior features from the query and use it to construct a dictionary and a template. The dictionary is used to calculate reconstruction errors for each frame of test audio to generate the subspace based distance vector. The distance matrix for DTW is computed using the query template and test posteriors. Each column of the distance matrix is then regularized using the errors from sparse recovery. DTW is finally applied to obtain a hypothesis. If the length of the hypothesis is smaller than half the query length, it is discarded to reduce false alarm. Otherwise, the hypothesis score is compared to a threshold to yield a final decision.

4.4.1 Subspace Regularized DTW (SR-DTW)

The system presented here relies on the notion that the reconstruction error for a test frame can be considered as distance between the query subspace and the corresponding test frame (Ram et al., 2017). In this method, we propose to use the subspace based distance to regularize the distance matrix for DTW as shown in Figure 4.3. Let us consider, $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_m]$ represent the posterior feature vectors corresponding to the query speech and $\mathbf{Y} = [\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_n]$ corresponding to a test utterance. Here, m and n represent the number of frames in the query speech and test utterance respectively. The distance matrix (Δ) used for DTW can then be calculated as follows:

$$\Delta(i,j) = d(\mathbf{x}_i, \mathbf{y}_j) \quad \forall\ i = 1,2,\ldots,m \\ \text{and } j = 1,2,\ldots,n \tag{4.6}$$

where $d(.,.)$ is a standard distance measure such as Euclidean, cosine, etc.

On the other hand, the subspace based distance (reconstruction error $e(j)$) for a test frame $\mathbf{y}_j$ can be calculated using (4.3) and (4.5). We observe that each column of this distance matrix corresponds to the frame-level distance between a test frame and all frames of the query whereas we only have one number representing the distance from a test frame to the query subspace as a whole. The DTW distance matrix is then regularized by replacing each of its columns by a weighted average of each element in this column and the subspace based

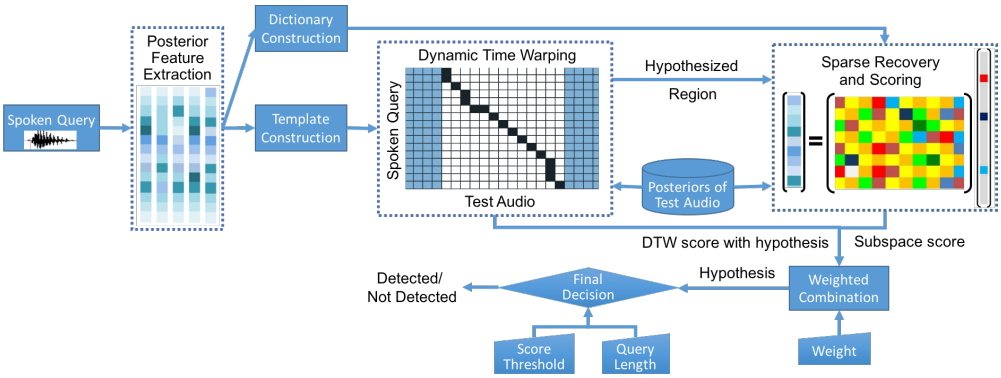

Figure 4.4: Block diagram of the system for subspace based re-scoring of DTW. We first extract posterior features from the query and use it to construct a dictionary and a template. The template is used in the baseline DTW system to hypothesize the region of occurrence of the query and obtain a likelihood score. We obtain sparse representation of the hypothesized region and compute subspace score using corresponding reconstruction errors. The final score is then calculated by taking a convex combination these two scores. If the length of the hypothesis is smaller than half the query length, it is discarded to reduce false alarm. Otherwise, the hypothesis score is compared to a threshold to yield the final decision.

distance obtained using the same test frame:

$$\begin{aligned} \Delta_{reg}(i,j) &= w_d \times \Delta(i,j) + (1-w_d) \times e(j) \\ &\forall i = 1,2,\ldots,m \text{ and } j = 1,2,\ldots,n \end{aligned} \tag{4.7}$$

where Δ_{reg} is the regularized distance matrix and w_d is a fixed regularization weight parameter, which will be optimized on an independent query development set. We then perform dynamic programming on this regularized distance matrix of $\Delta_{reg}(i,j)$ in a similar manner as the baseline system (Rodriguez-Fuentes et al., 2014) to obtain a region of occurrence of the query and calculate the likelihood of its occurrence. The whole procedure to implement this system is presented in Algorithm 3.

The key idea behind the proposed method is that the frame-level DTW exploits local similarities and properly models the temporal information, while the subspace-based distance captures the similarity at the subspace-level, which considers all frames present in the query for each test frame. A combination of these two distances is then shown to provide better decision likelihoods, resulting in performance improvement.

In principle, this approach is applicable to any variant of DTW by regularizing the corresponding distance matrix. However, in this work, and to provide us with strong reference points, we implemented the system presented in (Rodriguez-Fuentes et al., 2014) and perform the proposed regularization over the distance matrix followed by dynamic programming to obtain the detection regions along with their likelihood scores.

Algorithm 3 Subspace Regularized DTW (SR-DTW) (Fig. 4.3)

Input: Spoken query and posteriors of a test utterance
Output: Decision if the query occurs in the test utterance

1: Extract the posterior features from spoken query
2: Perform context appending according to (4.4) for both query and test posteriors
3: Construct a dictionary by concatenating the posteriors from different examples or learn a dictionary using (4.2)
4: Construct a template for DTW as discussed in Section 2.6.3
5: Compute sparse representation of test posteriors using (4.3) and corresponding reconstruction errors using (4.5)
6: Construct a distance matrix by computing a normalized cosine distance between each pair of posteriors from query and test utterance as discussed in Section 2.6.4.
7: Regularize the distance matrix using the reconstruction error according to (4.7)
8: Perform DTW to make a decision as described in Section 2.6.4.

4.4.2 Subspace Based Rescoring of DTW (SRS-DTW)

In this section, we investigate another approach to integrate information from sparsity into DTW based systems. Instead of regularizing the distance matrix, we propose to re-score the hypothesized region obtained from DTW using sparse coding.

Considering the spoken query $\mathbf{X}$ and the test audio $\mathbf{Y}$, we apply the modified DTW algorithm (as explained in Section 2.6.4) to obtain a hypothesized region denoted as $\mathbf{Y}_{hyp} = \left[\mathbf{y}_a, \mathbf{y}_{a+1}, \ldots, \mathbf{y}_{b-1}, \mathbf{y}_b\right]$ and the corresponding normalized similarity score is S_{DTW}. Then we construct a dictionary for the query using one of the methods discussed in Section 4.2.2. We use this dictionary in (4.3) to generate sparse representation for each frame of the query and employ those representations in (4.5) to calculate the reconstruction errors for each frame $\mathbf{y}_i$ of the hypothesized region. The resulting error sequence is represented as $e_{a,b} = [\, e_a, e_{a+1}, \ldots, e_{b-1}, e_b \,]$, which is used to calculate another score for the hypothesized region, $\mathbf{Y}_{hyp}$ as follows,

$$S_{Subspace} = 1 - \frac{1}{b-a+1} \sum_{i=a}^{b} e_i \tag{4.8}$$

We call it subspace based score which represents average similarity between the hypothesized region and the spoken query using subspace structure of speech. Once we have scores from both systems, we take a weighted average as follows:

$$S = w_s \times S_{DTW} + (1 - w_s) \times S_{Subspace} \tag{4.9}$$

where S is the final similarity score between the hypothesized region and the spoken query, and w_s represents the associated weight. A block diagram of this proposed re-scoring mechanism is presented in Figure 4.4. Also, a step-by-step summary for implementing the system is described in Algorithm 4.

Algorithm 4 Subspace Based DTW Rescoring (SRS-DTW) (Fig. 4.4)

Input: Spoken query and posteriors of a test utterance
Output: Decision if the query occurs in the test utterance

1: Extract the posterior features from spoken query
2: Perform context appending according to (4.4) for both query and test posteriors
3: Construct a dictionary by concatenating the posteriors from different examples or learn a dictionary using (4.2)
4: Construct a template for DTW as discussed in Section 2.6.3
5: Perform DTW using the template to obtain a hypothesized region and corresponding score, S_{DTW} as described in Section 2.6.4
6: Compute sparse representation of test posteriors of the hypothesized region using (4.3)
7: Compute corresponding reconstruction errors according to (4.5) and use it to obtain the subspace based score, $S_{Subspace}$ according to (4.8)
8: Compute the final score using S_{DTW} and $S_{Subspace}$ according to (4.9)
9: Use query length and score threshold to make a final decision

4.5 Experimental Set-up

We use two different databases to evaluate and analyze the proposed approaches: the AMI meeting corpus (IHM set) and the SWS 2013 corpus. The details of these datasets are described in Section 2.4. In this section, we present different pre-processing steps and evaluation metrics used for our experiments.

4.5.1 Query Selection

In AMI meeting corpus, there are approximately 12k words in the training, out of which we extracted 200 more frequent words (excluding functional words) for our detection experiments including very short words such as 'ADD' to long words such as 'TECHNOLOGY' (as described in Section 3.5.1). Later, these queries are divided into two groups in a random manner to obtain sets of 100 queries each. One set is used as development queries to optimize the parameters of different systems whereas the other set is used to evaluate the performance of corresponding system. We use the test set of AMI as the search database for QbE-STD evaluation which contains 12612 utterances.

4.5.2 Feature Extraction

We have used the setup presented in Section 2.3.2 to extract phone posterior features for our detection experiments. The setup corresponding to different databases is implemented separately as described below.

- **AMI Phone Recognizer**: The phone posterior features are estimated from a DNN with MFCC based spectral features as input. These MFCC features are extracted form speech

data by following the steps described in Section 2.3.1. To add contextual information, 4 frames of left and right acoustic contexts are appended (total 9 frames) to have a 351 dimensional input vector to the DNN. The DNN consists of 3 hidden layers of 1024 neurons each, to estimate 43 dimensional phone posteriors at the output. There are 39 phones obtained from CMU pronunciation dictionary[1] for lexical modeling. The remaining 4 phones are used to model silence and non-speech sounds. The training labels for the DNN are generated using a GMM-HMM based speech recognizer (Hinton et al., 2012). The recognizer is used to force align the training data to obtain the corresponding phonetic transcription. This whole setup is implemented using the Kaldi toolkit (Povey et al., 2011).

- **BUT Phone Recognizer**: There is no data available for training a phone posterior extractor for the SWS 2013 database. Thus, we use the same phone recognizers as used in (Rodriguez-Fuentes and Penagarikano, 2013) to estimate phone posteriors. The phone recognizers were developed at Brno University of Technology (BUT) for three different languages: Czech, Hungarian and Russian (Schwarz, 2008). These recognizers were trained using SpeechDAT(E) (Pollák et al., 2000) database which contains 12, 10 and 18 hours of speech for the respective languages. There are 43, 59 and 50 phones for the respective languages. In all cases, 3 additional units were considered to model silence and non-speech sounds.

4.5.3 Speech Activity Detection

We perform speech activity detection (SAD) to remove the noisy frames from test utterances as well as queries as discussed in Section 2.6.2. In case of AMI dataset, we have one set of phone posteriors that we use to perform SAD. However, we have 3 sets of phone posteriors (Czech, Hungarian and Russian) in case of SWS 2013 dataset. We average the probability of non-speech units from all 3 phone posteriors to perform SAD.

4.5.4 Score Normalization

We obtain a likelihood score corresponding to each pair of spoken query and test utterance using QbE-STD. These scores are normalized to have zero-mean and unit-variance per query to reduces the variability in scores across different queries (Section 3.5.5).

4.5.5 Evaluation Metric

We use $MTWV$, C_{nxe}^{min} and DET curves (as described in Section 2.5) to evaluate and compare the performance of different systems. The $MTWV$ considers prior probability of occurrence (P_{target}) of a query in the test utterances which is 4×10^{-3} and 8×10^{-4} for AMI and SWS 2013 respectively. For our experiments, we consider cost of false alarm (C_{fa}) to be 1, cost of missed

[1] http://www.speech.cs.cmu.edu/cgi-bin/cmudict

Table 4.1: Performance comparison of the three proposed systems for query detection with baseline system using only 1 example per query.

System	$MTWV \uparrow$	$C_{nxe}^{min} \downarrow$	$SSF \downarrow$
Baseline	0.4758	0.6526	0.1778
SSD	0.3030	0.7874	**0.0367**
SR-DTW	**0.4914****	**0.6376****	0.1889
SRS-DTW	0.4875**	0.6399*	0.1831

* significant at $p < 0.05$; ** significant at $p < 0.001$

detection (C_m) to be 100 resulting in the weight factor (β) of 2.49 and 12.49 for AMI and SWS 2013 respectively. We also perform statistical significance test to measure the improvements in $MTWV$ and C_{nxe}^{min} and the corresponding p-values are indicated with the results.

In order to compare the computational efficiency of different approaches, we also report the Searching Speed Factor (SSF) (Rodriguez-Fuentes and Penagarikano, 2013), which indicates the amount of CPU effort required to search a query in an audio document. Let the duration of a query and a test audio be t_q and t_a units of time, respectively. If an algorithm takes t units of time to search the query, the SSF is defined as: $\frac{t}{t_q \times t_a}$. The total CPU time is reported as if it was computed on a single CPU. If multiple examples are used to search a given query, we use average duration of those examples as the query duration. Lower value of SSF corresponds to a faster system.

4.6 Experimental Analysis

We conducted extensive experiments on AMI meeting corpus to analyze the performance of different systems proposed in this paper. The experiments are performed in two challenging scenarios when very few examples (10 examples) or just one query example is provided for QbE-STD, and the test utterances are conversational spontaneous speech with competitive speakers.

4.6.1 Baseline System

The DTW based QbE-STD system discussed in Section 2.6 is used as a highly competitive baseline system (Anguera et al., 2014). The performance of this system using evaluation queries are shown in Tables 4.1 and 4.2 corresponding to single and multiple examples per query respectively. We observe that the performance with multiple examples per query is significantly better than its one example counterpart. This indicates that template averaging is able to incorporate variations from multiple examples of the same query which is similar to the observations presented in (Rodriguez-Fuentes et al., 2014).

Table 4.2: Performance comparison of the three proposed systems for query detection with baseline system using 10 examples per query.

System	10 Examples (Concatenation)			10 Examples (Learning)		
	$MTWV\uparrow$	$C_{nxe}^{min}\downarrow$	$SSF\downarrow$	$MTWV\uparrow$	$C_{nxe}^{min}\downarrow$	$SSF\downarrow$
Baseline†	0.6028	0.5014	0.2070	0.6028	0.5014	0.2070
SSD	0.4117	0.6613	**0.1109**	0.3992	0.6897	**0.0406**
SR-DTW	0.6332***	0.4797**	0.2415	0.6231***	0.4847**	0.2220
SRS-DTW	**0.6374*****	**0.4610*****	0.2242	**0.6323*****	**0.4674*****	0.2123

†the baseline system uses template averaging in case of 10-examples
* significant at $p < 0.05$; ** significant at $p < 0.001$; *** significant at $p < 0.00001$;

4.6.2 Sparse Subspace Detection (SSD)

In this section, we evaluate the sparse subspace detection system presented in Section 4.3. This is a sparsity based system which completely relies on the subspace structure of speech data. The purpose of developing this system is to quantify the contribution of subspace structure of speech for the task of QbE-STD. We follow the steps presented in Algorithm 2 to implement this system. As discussed in Section 4.3, in case of one example per query context appended posterior features of the example constitute the dictionary. On the other hand, with multiple examples for each query, we construct the dictionary in two ways: (i) concatenation of context appended posteriors and, (ii) learning from the context appended posteriors of different examples of the query. The size of these dictionaries vary depending on the length of query examples, context size and the dictionary construction method being used (concatenation or learning). The number of rows equals the length of context appended posteriors whereas the number of columns (atoms) depend on the number of frames in the query examples and the dictionary construction method. Dictionary construction is followed by the rest of the steps in Algorithm 2 to find the region of occurrence and likelihood score of the query in the test utterance. Context size (c), the level of sparsity (λ) and a single threshold parameter used in Kadane's algorithm are optimized using development queries to have the best detection performance. The parameters are optimized for all development queries, and these are not dependent on individual queries. The results using evaluation queries are presented in Tables 4.1 and 4.2 corresponding to single and multiple examples per query respectively. The performance of this system is not as good as the baseline system in both cases. However, this is a much faster technique compared to the baseline system as indicated by the lower value of SSF. Also, it shows the subspace structure of speech can be used to perform QbE-STD with reasonable accuracy. The performance degradation can be attributed to the absence of a framework to incorporate temporal information.

4.6.3 Subspace Regularized DTW (SR-DTW)

The subspace regularized DTW system is proposed to incorporate the temporal information from the spoken utterance, as discussed in Section 4.4.1. To evaluate this system, we construct a dictionary for each query as discussed in previous section and follow the steps shown in Algorithm 3 to obtain the likelihood score for a query occurring in a test utterance. The parameters (Context, λ and w_d) are optimized using the development queries and the results on evaluation queries are shown in Tables 4.1 and 4.2 respectively. The optimization is done by varying all the parameters in their respective ranges and maximizing the $MTWV$. Clearly, this system gives improvement over the baseline system in both cases of single and multiple examples per query which shows the importance of exploiting the subspace structure of speech while developing a QbE-STD system.

4.6.4 Subspace Based Rescoring of DTW (SRS-DTW)

Another approach to take advantage of both DTW and sparsity based system, is to combine their respective scores as discussed in Section 4.4.2. In this case also, we construct different dictionaries depending on the number of examples provided for each query and perform corresponding detection experiment. We follow the procedure described in Algorithm 4 to obtain the likelihood score for a query occurring in a test utterance. For this system, the Context and λ parameter are optimized by keeping w_s equal to 0. This essentially means that we are trying to obtain the best set of scores using only the sparsity based errors, irrespective of the scores generated by the baseline system. Once the context and λ have been so optimized , we vary w_s in a given range to obtain the best value.

The resulting performances are summarized in Tables 4.1 and 4.2 corresponding to single and multiple examples per query respectively. Similar to the SR-DTW system, this system also gives improvement over the baseline system in both cases, once again indicating the importance of subspace structure of speech for the problem at hand. The performance of these systems are also shown using DET curves in Figures 4.5 and 4.6 corresponding to single and multiple examples case respectively. The curves show that the performance improvement is consistent over all operating points in the DET curve.

4.6.5 Concatenated vs Learned Dictionary

We have performed two sets of experiments for all systems proposed in this work when multiple examples per query are provided. They differ in the way corresponding query dictionaries are constructed. When we compare the performance in these cases for all three systems (as presented in Table 4.2), we can see that the performance is worse when the dictionary is learned from the given examples compared to concatenating them to form the dictionary. This indicates that the dictionary learning algorithm has not been able to capture all the information from the query examples provided. However, the performance difference is small,

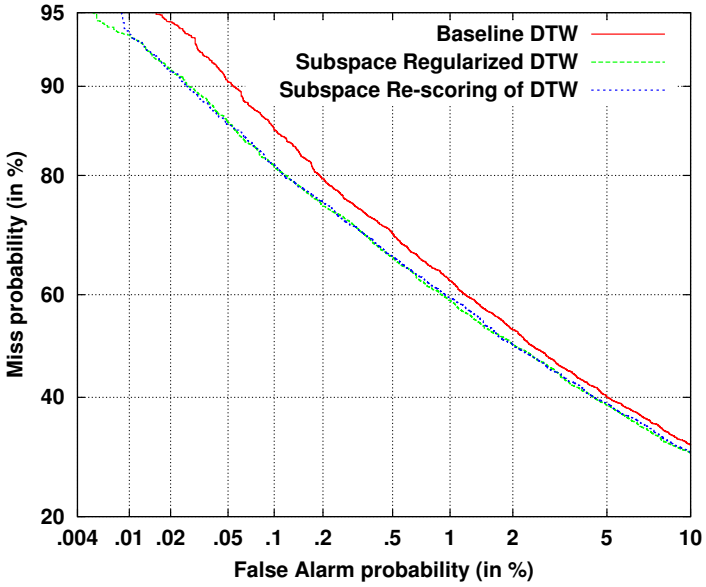

Figure 4.5: DET curves showing the performance of the Sparse-DTW hybrid systems compared to the baseline DTW system using 1 example per query.

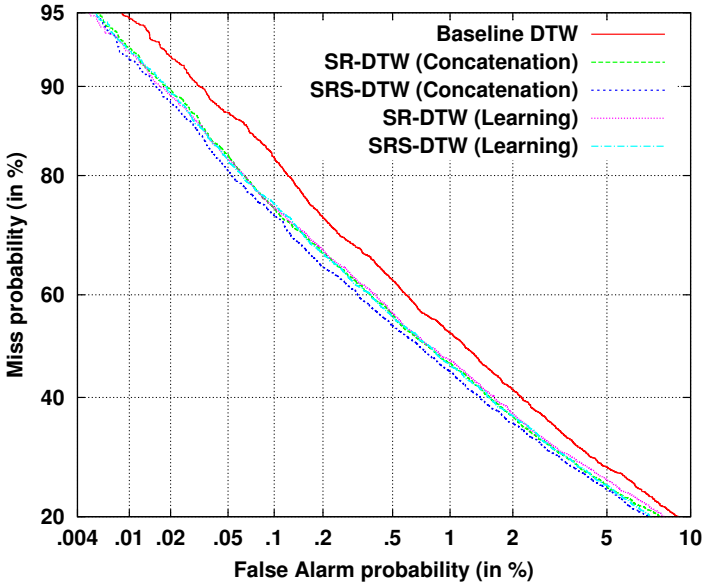

Figure 4.6: DET curves showing the performance of the Sparse-DTW hybrid systems compared to the baseline DTW system using 10 examples per query.

Table 4.3: Optimized Values of Context and λ giving the highest MTWV score on development queries for different systems

Systems	1 Example		10 Examples			
			Concatenation		Learning	
	Context	λ	Context	λ	Context	λ
SSD	4	0.01	9	0.6	8	0.1
SR-DTW	1	0.1	2	0.3	2	0.2
SRS-DTW	4	0.01	7	0.5	8	0.1

indicating the validity of dictionary learning approach when concatenating the examples of a query increases the computational cost significantly.

4.6.6 Effect of Context and λ

In this section, we discuss the effect of acoustic context (as indicated in (4.4)) and λ on different systems. The optimal value of these parameters to obtain best $MTWV$ using development queries is presented in Table 4.3. The context size depends on the average query length to capture longer temporal dependency. As we add more examples to generate query templates, the optimal context size increases. On the other hand, the value of λ indicates the desired level of sparsity. It is dependent on the number of atoms present in the dictionary for sparse representation. Bigger dictionaries require higher λ to achieve good reconstruction of the test frames. Thus we need higher λ for the 10-examples case compared to the 1-example case. However, the number of atoms in case of 10-examples (Learning) is higher than 1-example case, but lower than 10-examples (Concatenation) case. This leads to the optimal value of λ for 10-examples (Learning) case being higher than 1-example case but lower than 10-examples (Concatenation) case. We also observe that optimal context size in case of SR-DTW is smaller compared to other two systems. The reason is, in SR-DTW system, we are not only trying to obtain better reconstruction of test frames, but also want to hypothesize the regions representing queries. Broader context produces better reconstruction for smaller regions, effectively reducing the length of the hypothesized regions. Thus the system makes a trade-off between quality of reconstruction and length of the detected region and the optimal context size is smaller than other systems. As an example of this optimization, we present in Figure 4.7 the variation of $MTWV$ score with respect to context size and different values of λ for 10-examples (Concatenation) case. The scores are generated using SRS-DTW system on development queries while keeping the weight parameter, $w_s = 0$. Clearly, $Context = 7$ and $\lambda = 0.5$ give the best performance, which is later used to optimize the value of w_s.

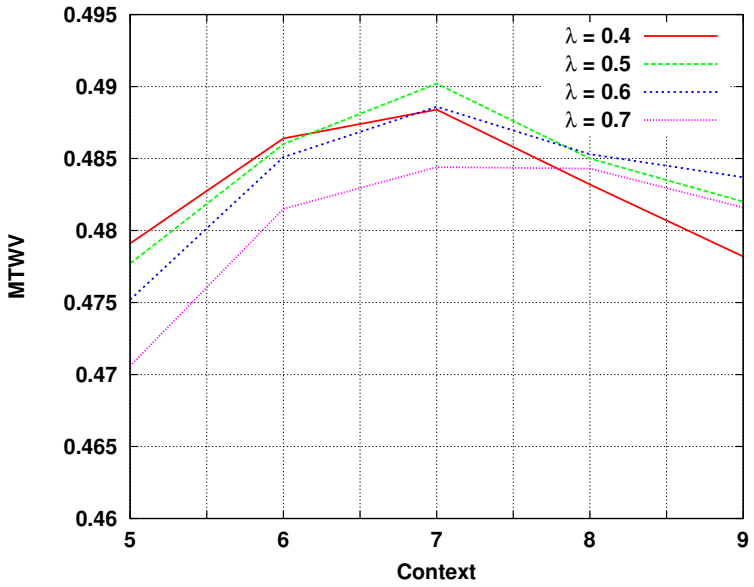

Figure 4.7: Variation of $MTWV$ with changing context for different values of λ. The experiments are performed using SRS-DTW system on development queries with 10 examples (Concatenation) per query, while keeping $w_s = 0$. It corresponds to the scenario when we obtain the scores from sparse reconstruction errors by using boundaries from the baseline system

4.6.7 Effect of Fusion Weight

We have proposed two ways of fusing the baseline DTW and sparsity based system as discussed in Section 4.4. Parameters w_d and w_s indicate the fusion weights for SR-DTW and SRS-DTW system, respectively. In both cases, $(1-w)$ represent the contribution of information obtained by relying on the subspace structure of speech. Thus, higher w corresponds to lower contribution from sparsity. We have optimized these fusion weights on development queries for both systems. As an example, we show the performance variation of SRS-DTW system with corresponding weight, w_s in Table 4.4, while keeping the other parameters ($Context$ and λ) fixed. We also present the corresponding baseline performance for comparison. We observe that, $w_s = 0.7, 0.8, 0.9$ gives very similar results for 1-example case and $w_s = 0.6, 0.7, 0.8$ for 10-examples (Concatenation) case. This indicates a range of values of w_s to obtain similar results which are better than the baseline. The optimal values of w_s (to obtain best $MTWV$) are 0.8 and 0.7 corresponding to the cases of 1-example and 10-examples (Concatenation). So, the effective weights for sparsity based scores are 0.2 and 0.3 respectively. It shows that the sparsity based scores provide better discrimination with more examples for each query. This is in conformity with the idea of subspace modeling where many examples are needed for better modeling of a class (Mairal et al., 2010).

Table 4.4: Variation of $MTWV$ and C_{nxe}^{min} for different values of fusion weight (w_s). The experiments are performed using SRS-DTW system on development queries.

weight (w_s)	1 Example		10 Examples (Concat.)	
	$MTWV \uparrow$	$C_{nxe}^{min} \downarrow$	$MTWV \uparrow$	$C_{nxe}^{min} \downarrow$
0.6	0.4638	0.6394	0.6043	**0.4757**
0.7	0.4761	**0.6345**	**0.6051**	0.4765
0.8	**0.4795**	0.6355	0.6015	0.4828
0.9	0.4767	0.6414	0.5890	0.4965
Baseline	0.4646	0.6558	0.5684	0.5192

4.6.8 Computational Efficiency

The computational efficiency of different systems is shown in Tables 4.1 and 4.2 using SSF metric. It can be observed that sparse subspace detection (SSD) is the most efficient system among all in both cases of using different number of examples. The price for this efficiency is paid by degradation in detection performance. On the other hand, the hybrid approaches need more computation than the baseline system, because in both systems, we perform DTW as in the baseline system while performing additional computation for obtaining the sparse representation to complete the hybridization. Also, SRS-DTW system is computationally more efficient than the SR-DTW system because in the case of SRS-DTW system, we perform sparse coding only for a sub-sequence (hypothesized region from baseline) of the test utterance, whereas for SR-DTW system, we need the sparse representation for the whole utterance to obtain the regularized distance matrix for DTW. We further observe that, dictionary learning approach is faster than the concatenation of examples of a query. This difference in speed is due to the smaller size of dictionary used for sparse coding when we have learned a dictionary from different examples of a query. Thus in all cases, there is a trade-off between performance enhancement and computational efficiency of the systems and we can choose a system to perform QbE-STD depending on our requirements.

4.7 Experiments on SWS 2013

We conducted another set of experiments on SWS 2013 database to show the validity of proposed approach in real life scenarios. As discussed in Section 4.5.2, we use 3 different BUT phone recognizers to extract the posterior features. In Rodriguez-Fuentes et al. (2014), the authors concatenate the feature vectors obtained from different phone recognizers to perform query detection, which was their best individual system. Thus, we implemented it as our baseline system.

Out of the three proposed systems, we use the SRS-DTW system due to its ease of parameter optimization and superior performance compared to other systems. We perform separate

Table 4.5: Performance comparison of the baseline system and subspace based re-scoring of DTW system on SWS 2013. Each system is evaluated for three different cases where different number of examples per query is available.

Examples per query	Baseline System		Proposed System	
	$MTWV \uparrow$	$C_{nxe}^{min} \downarrow$	$MTWV \uparrow$	$C_{nxe}^{min} \downarrow$
1	0.4287	0.6183	0.4362^{*}	0.6071^{**}
3	0.3007	0.6682	0.3204^{***}	0.6571^{**}
10	0.2740	0.6893	0.3020^{***}	0.6703^{***}

* significant at $p < 0.05$; ** significant at $p < 0.001$; *** significant at $p < 0.00001$;

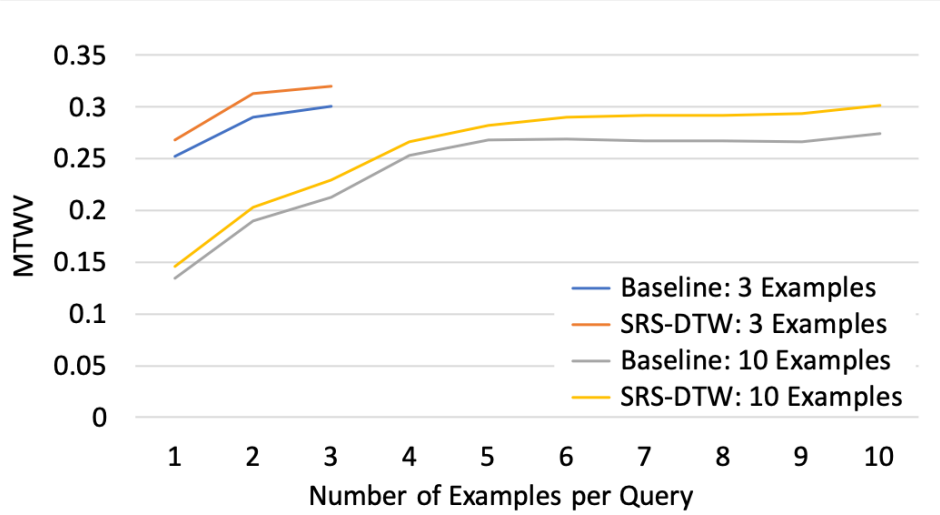

Figure 4.8: Comparison of improvements in $MTWV$ score with additional examples per query for baseline DTW and proposed SRS-DTW system. The performance gain is higher with the proposed system.

experiments for queries with different number of examples available per query. In case of multiple examples per query, we concatenate them to construct the corresponding dictionary as it gives better performance compared to dictionary learning experiments on AMI corpus. The parameters of our system are optimized using development queries and the results using evaluation queries are presented in Table 4.5. The difference in performance in three sets of queries can be attributed to the corresponding quality of recordings. Clearly, our system performs better than the baseline system in all three cases. We observe that the performance gain increases with increasing number of examples per query. This is similar to the results obtained on AMI database. To analyze the effect of additional examples per query (for queries with 3 or 10 examples), we conduct another set of experiments where we add one example at a time to each query and obtain the corresponding detection performance. The resulting $MTWV$ values are presented as a function of the number of examples per query in Figure 4.8. Clearly, the performance improvement is higher with additional examples for SRS-DTW system compared to the baseline. The overall performance gain indicates that the proposed methods are generalizable to real-world scenario and shows the importance of low-dimensional subspace structure of speech for the task of QbE-STD.

4.8 Conclusion

In this chapter, we have proposed three different systems exploiting the low-dimensional subspace structure of speech. The performance of these systems indicate the usefulness of this structure for QbE-STD. The sparse subspace detection system is shown to be faster than the baseline template matching system with reasonable accuracy. On the other hand, the hybrid systems relying on sparse representation as well as template matching approach yield better performance. The improvement is higher in case of multiple examples per query, which indicates the capability of the proposed approaches to exploit the information from multiple examples better than the baseline system. The performance gain in MediaEval challenge database validates our approach in challenging real-world scenarios. It has also been shown that the proposed systems benefit from multiple examples of a query.

5 Neural Network Modeling for QbE-STD

Contents

This chapter is based on the following papers:

Dhananjay Ram, Lesly Miculicich, and Hervé Bourlard. CNN based query by example spoken term detection. In *Nineteenth Annual Conference of the International Speech Communication Association (INTERSPEECH)*, 2018c

Dhananjay Ram, Lesly Miculicich, and Hervé Bourlard. Multilingual bottleneck features for query by example spoken term detection. In *Automatic Speech Recognition & Understanding, ASRU. IEEE Workshop on*, 2019a

Dhananjay Ram, Lesly Miculicich, and Hervé Bourlard. Neural network based end-to-end query by example spoken term detection. *arXiv preprint arXiv:1911.08332*, 2019b

5.1 Introduction

In the last two chapters, we investigated several sparsity based approaches to (1) generate better phone or phonological posterior features, and (2) improve the DTW based matching algorithm. In spite of the significant improvements over the baseline, those systems still suffer from the language mismatch problem during DNN based feature extraction. To deal with this problem, we train multilingual networks aimed at obtaining language independent representation.

We also observe that DTW based template matching is not well suited in case of local mismatches between the test utterances and its associated query, resulting in low matching score. Thus,we propose a novel approach using Convolutional Neural Network (CNN) architecture to match a query and a test utterance instead of DTW based matching. Finally, we integrate the representation learning and CNN-based matching to jointly train and further improve the QbE-STD performance. Different components of this system are implemented separately to analyze their performance before building the end-to-end system as discussed in the following.

(i) *Representation Learning (Section 5.2)*: In contrast to using several language dependent posterior features (as discussed in Section 4.5.2) for QbE-STD, here we propose to train multilingual bottleneck networks to estimate language independent representation of the query and test utterances. This is achieved by using multitask learning principle (Caruana, 1997) to jointly classify phones from multiple languages and the shared network is able to learn language independent representation. These representations are used to perform both DTW based template matching (Section 5.3) and CNN based Matching.

(ii) *CNN based Matching (Section 5.4)*: The DTW based template matching is applied on a frame-level similarity matrix computed from the feature vectors of the query and the test utterance to compute the likelihood score of occurrence. Unlike DTW, we view the similarity matrix as an image and propose to approach the QbE-STD problem as an image classification task. We observe that the similarity matrix contains a quasi-diagonal pattern if the query occurs in the test utterance. Otherwise, no such pattern is observed. Thus for each spoken query, a test utterance can be categorized as an example of positive or negative class depending on whether the query occurs in it or not.

(iii) *End to End QbE-STD System (Section 5.5)*: The proposed neural network based end-to-end system takes spectral features (MFCC) corresponding to a query and a test utterance as input, and the output indicates whether the query occurs in the test utterance. It has three components: (i) Feature extraction, (ii) Similarity matrix computation and (iii) CNN based matching, combined into one architecture for end-to-end training. The feature extractor aims at obtaining language independent representation to produce better score for similarity matrix which in turn improves the CNN based matching.

The proposed end-to-end QbE-STD system has the following advantages over the baseline DTW based approach: (i) the CNN based matching provides a learning framework to the problem (ii) the CNN considers the whole similarity matrix at once to find a pattern, whereas the DTW algorithm takes localized decisions on the similarity matrix to find a warping path, (iii) the CNN based matching introduces a discrimination capability in the system and (iv) the end-to-end training enables joint optimization of the representation learning and the matching network.

The proposed methods are evaluated on SWS 2013 database and their generaliaztion ability is analyzed on QUESST 2014 database as described in Section 5.6. The significant improvements obtained using these approaches show the importance of a learning framework for QbE-STD. Finally, we present the conclusions in Section 5.7.

5.2 Representation Learning

DNNs have been traditionally used to obtain bottleneck feature based representation for speech related tasks (Yu and Seltzer, 2011; Veselỳ et al., 2012; Szoke et al., 2015) as discussed in Section 2.3.3. In this section, we present different DNN architectures to obtain monolingual as well as multilingual bottleneck features.

5.2.1 Monolingual Neural Network

We train DNNs for phone classification using five languages to estimate five distinct monolingual bottleneck features. Our monolingual DNN architecture, consists of 3 fully connected layers of 1024 neurons each, followed by a linear bottleneck layer of 32 neurons, and a fully connected layer of 1024 neurons. The final layer feeds to the output layer of size c_i corresponding to number of classes (e.g. phones) of the i-th language. The architecture is presented in Figure 5.1.

The monolingual bottleneck features have previously been shown to provide good performance for this task (Szoke et al., 2015). Here, we analyze their performance and propose to train multilingual networks to estimate better features for QbE-STD.

5.2.2 Multilingual Neural Network

Multilingual neural network have been studied in the context of ASR in order to obtain language independent representation of speech signal (Veselỳ et al., 2012). Those networks are trained using multitask learning (Caruana, 1997) which aims at exploiting similarities across tasks resulting in an improved learning efficiency when compared to training each task separately. Generally, the network architecture consists of a shared part and several task-dependent parts. In order to obtain multilingual bottleneck features we model phone classification for each language as different tasks, thus we have a language independent part and a language

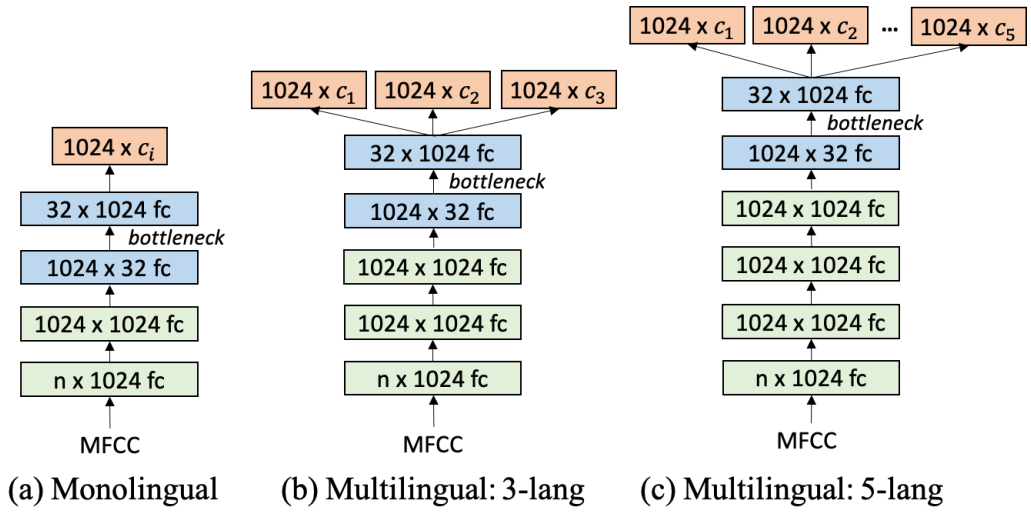

Figure 5.1: Monolingual and multilingual DNN architectures for extracting bottleneck features using multiple languages. c_i is the number of classes for the i-th language and n is the size of input vector.

dependent part. The language independent part is composed of the first layers of the network which are shared by all languages forcing the network to learn common characteristics. The language dependent part is modeled by the output layers (marked in red in Figure 5.1), and enables the network to learn particular characteristics of each language.

In this work, we train two different multilingual networks using 3 languages and 5 languages respectively in order to analyze the effect of training with additional languages. The architecture of these networks are presented in Figure 5.1 and described in the following.

- **Multilingual (3 languages)**: this architecture consists of 4 fully connected layers having 1024 neurons each, followed by a linear bottleneck layer of 32 neurons. Then, a fully connected layer of 1024 neurons feeds to 3 output layers corresponding to the different training languages. The 3 output layers are language dependent while the rest of the layers are shared among the languages.

- **Multilingual (5 languages)**: this architecture is similar to the previous one except it uses an additional fully connected layer of 1024 neurons, and two extra output layers corresponding to the 2 new languages. The increased number of layers is intended at modeling the extra training data gained by adding languages.

All neural networks discussed in this section have rectifier linear unit (ReLU) as non-linearity used after each linear transform except in the bottleneck layer and the output layer. The output layer has multiple softmax layers corresponding to each language.

5.3 DTW based Template Matching

We use different types of bottleneck features (monolingual and multilingual) discussed in previous section to perform DTW based template matching for QbE-STD. We follow the system described in Section 2.6 for this purpose. To construct the distance matrix for DTW with bottleneck features, cosine distance has been shown to yield better performance (Szoke et al., 2015) than the logarithm of cosine distance used for posterior features. Our preliminary experiments show similar trend, thus we use cosine distance for our experiments. The query matching can also be performed using CNN as discussed in the following section.

5.4 CNN based Matching

In this section, we present a novel CNN based query matching technique to perform QbE-STD. We cast the problem as a binary classification of images. We discuss how to construct these images and use a CNN for classification in the following.

5.4.1 Image Construction

We describe the procedure to construct a similarity matrix from a spoken query and a test utterance which is used as an image for binary classification. We extract bottleneck features (as discussed in Section 5.2) from both spoken queries and test utterances using MFCC features as input. Let us consider, $\mathbf{Q} = [\,\mathbf{q}_1, \mathbf{q}_2, \ldots, \mathbf{q}_m\,]$ representing the bottleneck features corresponding to a spoken query and $\mathbf{T} = [\,\mathbf{t}_1, \mathbf{t}_2, \ldots, \mathbf{t}_n\,]$ corresponding to a test utterance. Here, m and n represent the number of frames in the query and test utterance respectively. Given any two bottleneck feature vectors $\mathbf{q}_i$ and $\mathbf{t}_j$, we compute cosine similarity (Szoke et al., 2015) as follows:

$$s(\mathbf{q}_i, \mathbf{t}_j) = \frac{\mathbf{q}_i \cdot \mathbf{t}_j}{\|\mathbf{q}_i\| \cdot \|\mathbf{t}_i\|} \tag{5.1}$$

Higher values of s indicate higher similarity between the vectors. We further apply a range normalization such that all values in the similarity matrix will be between -1 to 1. This helps in dealing with variations in similarity scores for different pairs of query and search utterances.

$$s_{norm}(\mathbf{q}_i, \mathbf{t}_j) = -1 + 2.\frac{(s(\mathbf{q}_i, \mathbf{t}_j) - s_{min})}{(s_{max} - s_{min})} \tag{5.2}$$

where $s_{min} = \min_{i,j}(s(\mathbf{q}_i, \mathbf{t}_j))$ and $s_{max} = \max_{i,j}(s(\mathbf{q}_i, \mathbf{t}_j))$.

The similarity matrix is categorized in two class of images: (i) if a query occurs in a test utterance (positive class) and (ii) if a query does not occur in a test utterance (negative class). We present one example for each of this type of images in Figures 5.2 and 5.3 respectively. The vertical and horizontal axes represent the frames of query and test utterance respectively. The colors indicate strength of values in the matrix, higher values correspond to red and lower

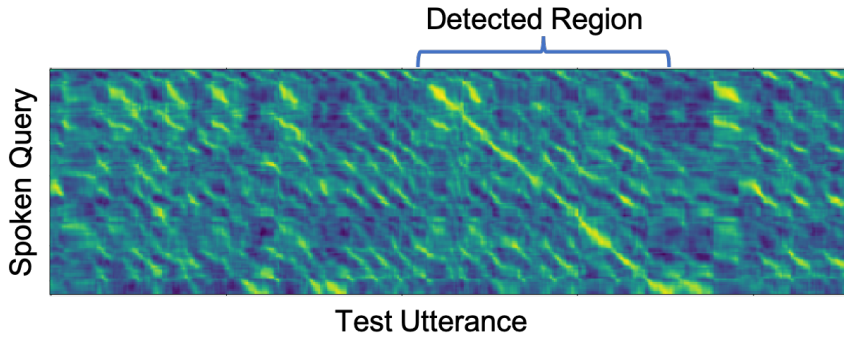

Figure 5.2: Positive case: the query occurs in the test utterance

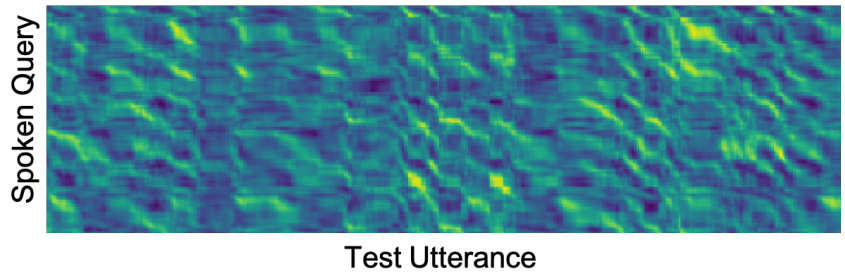

Figure 5.3: Negative case: the query does not occur in the test utterance

values to blue. The quasi-diagonal pattern observed in the positive class helps to discriminate between the two classes. We present our methodology in the following section to achieve this goal.

5.4.2 Methodology

In this section, we present a CNN based classifier for QbE-STD. Our CNN architecture is similar to the VGG network (Simonyan and Zisserman, 2014) which has been shown to perform well in image recognition task. It consists of a series of convolution and max-pooling layers with a fixed setting of hyper-parameters for all layers, which simplifies the selection of hyper-parameters.

Contrary to the standard image classification task, the input of our CNN is a similarity matrix. Therefore, we use only one channel instead of three corresponding to the RGB color model for images. The architecture consists of four sets of two convolution layers and one max-pooling layer; followed by two fully-connected layers with a soft-max on top. The details are described in Table 5.1. All convolution layers use ReLU (Krizhevsky et al., 2012) as activation function. The number of channels and dropout were optimized to 30, and 0.2 respectively with a development set. Our architecture has eight convolution layers in total. We expected that a simpler network will be able to perform reasonably well given the simplicity of the task. However, preliminary experiments with less layers were not able to outperform the baseline system. It should be noted that, our system is a language independent system which can be

Table 5.1: CNN Architecture

Layer	Description
Input	100×800×1
Maxpool	Channel: in=1, out=1, Filter: 2x2, Stride: 2
Conv	Channel: in=1, out=30, Filter: 3x3, Stride: 1
Conv	Channel: in=30, out=30, Filter: 3x3, Stride: 1
Maxpool	Channel: in=30, out=30, Filter: 2x2, Stride: 2
Conv	Channel: in=30, out=30, Filter: 3x3, Stride: 1
Conv	Channel: in=30, out=30, Filter: 3x3, Stride: 1
Maxpool	Channel: in=30, out=30, Filter: 2x2, Stride: 2
Conv	Channel: in=30, out=30, Filter: 3x3, Stride: 1
Conv	Channel: in=30, out=30, Filter: 3x3, Stride: 1
Maxpool	Channel: in=30, out=30, Filter: 2x2, Stride: 2
Conv	Channel: in=30, out=30, Filter: 3x3, Stride: 1
Conv	Channel: in=30, out=15, Filter: 3x3, Stride: 1
Maxpool	Channel: in=15, out=15, Filter: 2x2, Stride: 2
FC	Input:1×23×15, Output=60
FC	Input:60, Output=2
SM	Input:2, Output=2

Conv: Convolution; FC: Fully connected; SM: Softmax

trained using query and test utterance pairs from any language with minimal supervision (without corresponding transcriptions) because it only requires the information whether the query occurs in the test utterance.

We faced two main challenges to train the CNN for our task which are described as follows:

- **Variable size input:** The similarity matrices have variable widths and lengths corresponding to the number of frames of spoken queries and test utterances respectively. We deal with this issue by fixing the size for all input matrices to an average width and length of the training samples (in our training set, it is 100×800). In case the similarity matrix has length or width larger than the defined input, we down-sample it by deleting its rows and/or columns in regular intervals. On the other hand, if the length or width is smaller, we simply fill the gap with the lowest similarity value from the corresponding distance matrix. Down sampling does not affect the quasi-diagonal pattern severely as the rows and columns being deleted are spread throughout the distance matrix. Also, we did not apply segmentation of test utterances in fixed size intervals because it will require the region of occurrence of the query in a test utterance which is not available for QbE-STD.
- **Unbalanced data:** Typically, the frequency of occurrence of a particular query in the search space is very small. As a consequence, the number of positive and negative samples is highly unbalanced (in our training data is 0.1% to 99.9% respectively). To deal with this problem, we balance the training set with equal number of positive and

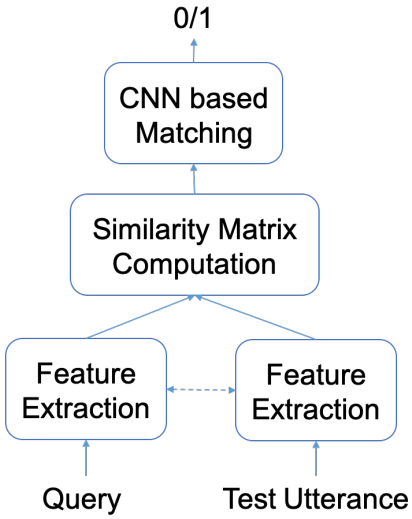

Figure 5.4: Neural network based end-to-end architecture for QbE-STD. The two feature extraction blocks share the same set of parameters.

negative examples. The negative examples were randomly sampled from the corresponding set at each iteration. Preliminary experiments showed that this strategy has better performance than using weighted loss function for training.

5.5 End to End QbE-STD System

In this section, we propose a novel neural network based end-to-end architecture to perform QbE-STD. We combine the representation learning network with the CNN based matching network in one architecture such that the input to the network are MFCC features corresponding to a query and a test utterance, and the output indicates whether the query occurs in the test utterance. We discuss the architecture and the training procedure in the following sections.

5.5.1 Architecture

The end-to-end architecture for QbE-STD has 3 components as shown in Figure 5.4: (i) Feature extraction, (ii) Similarity matrix computation and (iii) CNN based matching. The feature extraction block is used to obtain a frame-level representation using MFCC features as input for both the query and test utterance. The goal of this block is to obtain a language independent representation which produces better frame-level similarity score to construct the similarity matrix. This block can be implemented using DNN, CNN or LSTM network. In this work we use DNN for this purpose.

We can use any of the 3 architectures presented in Section 5.2 as our feature extraction block. However, we will see in Section 5.6 that the multi-lingual network trained using 5 languages

generates the best bottleneck features for QbE-STD. Thus we use this network as feature extraction block for our end-to-end architecture. We use the the language independent part (first 5 layers of the network, until bottleneck layer) of the network to extract features from both the query and test utterance which feeds to the 2nd block of our architecture.

The 2nd block of our architecture computes a frame-level similarity matrix between the query and the test utterance using the corresponding feature vectors as discussed in Section 5.4.1. This similarity matrix is then fed to the CNN to produce a matching score as discussed in Section 5.4.2. This whole network is jointly optimized by training it in an end-to-end manner as discussed in the following section.

5.5.2 Training Challenges

The end-to-end network faces same challenges as the CNN based matching network due to the nature of the problem: (i) variable size input, (ii) unbalanced data. We fix the size of input similarity matrix by either down-sampling or zero-padding whereas we randomly sample from the negative example set to balance the data from positive and negative classes as discussed in Section 5.4.2.

In addition, we do not have sufficient data to train this network from scratch. Thus, we use the principle of transfer learning (Pratt et al., 1991) to initialize different blocks of this network using previously trained network instead of random initialization. The CNN based matching block is initialized with the trained network from Section 5.4 and the feature extraction block is initialized with the first 5 layers of the 5 language neural network presented in Section 5.2.2. The weight matrices corresponding to CNN based matching block can be frozen during training to enable the system to only train the feature extraction block. In this setting, the CNN based matching block can be viewed as a loss function to extract better features. These feature vectors should be able to produce more discriminative quasi-diagonal patterns (as discussed in Section 5.4.1) required to classify the positive examples from the negative ones.

5.6 Experimental Analysis

In this section, we describe the databases used to train and evaluate the three systems proposed in this chapter: (i) *DTW based Matching* of bottleneck features, (ii) *CNN based Matching* of bottleneck features and (iii) *End-to-End* neural network model. Then, we present the training details corresponding to each system and analyze their QbE-STD performance.

5.6.1 Databases and Evaluation Metrics

We use the GlobalPhone database (see Section 2.4.4) to train the monolingual as well as multilingual models presented in Section 5.2. The QbE-STD experiments are performed on both SWS 2013 (see Section 2.4.2) and QUESST 2014 (see Section 2.4.3). In addition, we use

the SWS 2013 dataset to train the *CNN based Matching* network as well as the *End-to-End* network and evaluate the corresponding models. We use the QUESST 2014 dataset to show the generalization ability of those models.

Similar to previous chapters, we use $C_{nxe}^{\min}$ and $MTWV$ scores as well as DET curves (as described in Section 2.5) in order to evaluate and compare the three systems presented in this chapter. $C_{nxe}^{\min}$ is used as the primary metric to choose models for evaluation. For both SWS 2013 and QUESST 2014 databases, We consider the cost of false alarm (C_{fa}) to be 1 and cost of missed detection (C_m) to be 100 for computing $MTWV$. We also performed statistical significance test to measure the improvements in $C_{nxe}^{\min}$ and $MTWV$ scores.

5.6.2 DTW based Template Matching

We perform DTW based template matching using the bottleneck features extracted from several mono-lingual as well as multilingual networks. In the following, we discuss the training details for those networks to extract the corresponding bottleneck features and present their QbE-STD performance on SWS 2013 and QUESST 2014 databases.

Bottleneck Feature Extraction

We use Kaldi toolkit (Povey et al., 2011) to extract MFCC features and generate the target labels for training different neural networks presented in Section 5.2. MFCC features with a context of 6 frames (both left and right) constitutes the input vector of size 507. The context value is optimized using the development queries on SWS 2013. The outputs are monophone based tied states (also known as pdfs in Kaldi) corresponding to each language as presented in Table 2.4. The training labels for these networks are generated using a GMM-HMM based speech recognizer (Hinton et al., 2012). The number of classes corresponding to French, German, Portuguese, Spanish and Russian are 124, 133, 145, 130, 151 respectively. Note that, we also trained these networks using senone classes, however they perform worse than the mono phone based training.

The architectures presented in Section 5.2 are implemented using Pytorch (Paszke et al., 2017). We apply layer normalization (Ba et al., 2016) before the linear transforms and use rectifier linear unit (ReLU) as non-linearity after each linear transform except in the bottleneck layer for both mono and multilingual networks. We train those networks with batch size of 255 samples and dropout of 0.1. In case of multilingual training, we use equal number of samples from each language under consideration. Adam optimization algorithm (Kingma and Ba, 2014) is used with an initial learning rate of 10^{-3} to train all networks by optimizing cross entropy loss. The learning rate is halved every time the development loss increases compared to the previous epoch until a value of 10^{-4}. All the networks were trained for 50 epochs. We extract bottleneck features from these trained networks and apply speech activity detection (SAD) before using them for DTW as well as CNN based matching.

Table 5.2: Performance of the DTW based template matching approach in SWS 2013 using monolingual bottleneck features for single and multiple examples per query using all evaluation queries.

Training Language	Single Example		Multiple Examples	
	$C_{nxe}^{\min} \downarrow$	$MTWV \uparrow$	$C_{nxe}^{\min} \downarrow$	$MTWV \uparrow$
Portuguese (PT)	**0.6771**	**0.3786**	**0.6478**	0.3963
Spanish (ES)	0.6776	0.3754	0.6501	**0.3967**
Russian (RU)	0.7035	0.3184	0.6767	0.3383
French (FR)	0.7021	0.333	0.6757	0.3511
German (GE)	0.7503	0.2643	0.7257	0.2919

Table 5.3: Performance of the DTW based template matching approach in SWS 2013 using multilingual bottleneck features for single and multiple examples per query using all evaluation queries.

Multilingual Network	Single Example		Multiple Examples	
	$C_{nxe}^{\min} \downarrow$	$MTWV \uparrow$	$C_{nxe}^{\min} \downarrow$	$MTWV \uparrow$
PT-ES-RU	0.6330	0.4305	0.6023	0.4478
PT-ES-RU-FR-GE	**0.6204**	**0.4358**	**0.5866**	**0.4580**

We used the setup presented in Section 4.5.3 to perform SAD utilizing the phone posteriors computed from Czech, Hungarian and Russian phone recognizers trained on SpeechDAT(E) (Pollák et al., 2000) database. The 3 sets of phone posteriors are estimated for both SWS 2013 and QUESST 2014 databases and the corresponding non-speech probabilities are used to eliminate the noisy frames before matching.

Performance on SWS 2013

We consider two cases depending on the number of examples per query to evaluate different bottleneck features for QbE-STD. In case of a single example per query, the corresponding bottleneck features constitute the template. On the other hand, with multiple examples per query we compute an average template using traditional DTW (Sakoe and Chiba, 1978) before performing the detection experiment as discussed in Section 2.6.3.

The $C_{nxe}^{\min}$ and $MTWV$ scores for query detection experiments using monolingual bottleneck features is shown in Table 5.2. We can see that the bottleneck features from Portuguese (PT) performs the best in terms of $C_{nxe}^{\min}$ score with very close performance from Spanish (ES) bottleneck features.

We implemented two multilingual networks using 3 languages (PT, ES, RU) and 5 languages

Table 5.4: Performance of the DTW based template matching approach in QUESST 2014 using monolingual bottleneck features for different types of queries in evaluation set.

Training Language	T1 Queries		T2 Queries		T3 Queries	
	$C_{nxe}^{\min} \downarrow$	$MTWV \uparrow$	$C_{nxe}^{\min} \downarrow$	$MTWV \uparrow$	$C_{nxe}^{\min} \downarrow$	$MTWV \uparrow$
Portuguese (PT)	**0.5582**	**0.4671**	**0.6814**	**0.3048**	**0.8062**	**0.1915**
Spanish (ES)	0.5788	0.4648	0.7074	0.2695	0.8361	0.1612
Russian (RU)	0.6119	0.4148	0.7285	0.2434	0.8499	0.1385
French (FR)	0.6266	0.4242	0.7462	0.2086	0.8522	0.1249
German (GE)	0.6655	0.3481	0.7786	0.1902	0.8533	0.1038

Table 5.5: Performance of the DTW based template matching approach in QUESST 2014 using multilingual bottleneck features for different types of queries in evaluation set.

Multilingual Network	T1 Queries		T2 Queries		T3 Queries	
	$C_{nxe}^{\min} \downarrow$	$MTWV \uparrow$	$C_{nxe}^{\min} \downarrow$	$MTWV \uparrow$	$C_{nxe}^{\min} \downarrow$	$MTWV \uparrow$
PT-ES-RU	0.4828	0.5459	0.6218	0.3626	0.7849	0.2057
PT-ES-RU-FR-GE	**0.4606**	**0.5663**	**0.6013**	**0.3605**	**0.7601**	**0.2138**

(PT, ES, RU, FR, GE) as discussed in Section 5.2.2. The 3 language network uses the best performing monolingual training languages. Performances of the features extracted from these networks are shown in Table 5.3. We observe that PT-ES-RU-FR-GE features significantly outperform PT-ES-RU features indicating that additional languages for training provide better language independent features.

Performance on QUESST 2014

We have only one example per query in case of QUESST 2014 dataset, thus the corresponding bottleneck features constitute the template. It has three different types of queries as discussed in Section 2.4.3. Similar to (Rodríguez-Fuentes et al., 2014), we did not employ any specific strategies to deal with those different types of queries. The $C_{nxe}^{\min}$ and $MTWV$ scores corresponding to different types of queries using monolingual bottleneck features is shown in Table 5.4. We can see that the bottleneck features from Portuguese (PT) performs the best in terms of $C_{nxe}^{\min}$ score for all three types of queries. The corresponding results using mulit-lingual bottleneck features is shown in Table 5.5. We have a similar observation as in SWS 2013 that PT-ES-RU-FR-GE network performs better than PT-ES-RU network indicating that more language for training helps in obtaining better bottleneck features for DTW.

Table 5.6: Performance of the CNN based matching approach in SWS 2013 using PT-ES-RU-FR-GE bottleneck features for single and multiple examples per query using all evaluation queries.

System	Single Example		Multiple Examples	
	$C_{nxe}^{\min} \downarrow$	$MTWV \uparrow$	$C_{nxe}^{\min} \downarrow$	$MTWV \uparrow$
DTW based Matching	0.6204	**0.4358**	0.5866	**0.4580**
CNN based Matching	**0.6078**	0.3986	**0.5767**	0.4115

5.6.3 CNN based Matching

We use the bottleneck features extracted from PT-ES-RU-FR-GE network to train a CNN for matching queries and test utterances, as it performs the best for both SWS 2013 and QUESST 2014 databases. We describe the CNN training process and compare its performance with the DTW based matching in the following.

CNN Training

The development and evaluation queries in SWS 2013 database share the same search space for QbE-STD. The labels provided for development queries indicate whether a query occurs in a test utterance or not. Thus we only have these queries to train our CNN. We use 495 out of 505 queries for training and rest of the 10 queries are used for tuning which were chosen in a random manner. Effectively, we have 1551 queries when we consider different examples of the same query. We have designed our experiment in this manner to follow the setup of SWS 2013 task and make a fair comparison.

We use the bottleneck features from all the queries and test utterances, and filter them using a SAD to obtain 1488×10750 training example pairs. Out of these examples 24118 are positive examples and rest are negative examples. We balance the classes following the strategy discussed in Section 5.4.2. We combine the examples from both classes and prepare batches of 20 samples of query and search utterance pairs. We use the Adam optimization algorithm (Kingma and Ba, 2014) with a learning rate of 10^{-4} to train the CNN by optimizing cross entropy loss. The whole setup is implemented using Pytorch (Paszke et al., 2017).

Performance on SWS 2013

We present the performance of CNN based matching system for QbE-STD in Table 5.6 and compare it with the best DTW based matching system from previous section for different number of examples per query. Similar to DTW based system, we use template averaging to obtain the template for queries with multiple examples. This process was only performed during test time, however the training samples were formed using only single example per

Table 5.7: Performance of the CNN based matching approach in SWS 2013 using PT-ES-RU-FR-GE bottleneck features for different types of queries in evaluation set.

System	T1 Queries		T2 Queries		T3 Queries	
	$C_{nxe}^{\min} \downarrow$	$MTWV \uparrow$	$C_{nxe}^{\min} \downarrow$	$MTWV \uparrow$	$C_{nxe}^{\min} \downarrow$	$MTWV \uparrow$
DTW based Matching	0.4606	0.5663	0.6013	0.3605	0.7601	0.2138
CNN based Matching	**0.4121**	**0.6103**	**0.5235**	**0.4375**	**0.6569**	**0.3603**

query. We observe from Table 5.6 that the CNN based matching performs significantly better in terms of $C_{nxe}^{\min}$ score for both single and multiple examples per query case. This indicates that the CNN produces more informative scores about the ground-truth than the DTW.

Performance on QUESST 2014

We use the model trained on SWS 2013 for testing on QUESST 2014 evaluation set to analyze the generalizability of CNN based matching system. We compare the performance of DTW and CNN based matching and present the results in Table 5.7. As discussed earlier, it has three types of queries and we do not apply any specific strategies to deal with them. We can clearly see that CNN performs significantly better than the DTW system for all 3 types of queries. The performance gets increasingly worse from Type 1 to Type 2 and from Type 2 to Type 3. This can be attributed to the training of our system using only queries from SWS 2013 which are similar to Type 1 queries from QUESST 2014. However the consistency in performance improvement for all kinds of queries shows that CNN based matching system is generalizable to newer datasets.

5.6.4 End to End QbE-STD System

In this section, we utilize the bottleneck feature extractor and CNN based matching network to construct the end to end QbE-STD system. We present the end to end training procedure and its performance on both SWS 2013 and QUESST 2014 databases. We also discuss that the CNN based matching network can be used as a loss function to obtain better features for DTW based template matching.

End to End Training

We implement the architecture presented in Section 5.5.1 using Pytorch (Paszke et al., 2017). The training and development sets consists of the same pairs of query and test utterances as used for training the CNN in previous section. The difference is, we train the CNN using bottleneck features, whereas the end-to-end network uses the corresponding MFCC features. We attempt to train the network by randomly initializing the weight matrices of the whole

Table 5.8: Performance of the End-to-End neural network based approach in SWS 2013 for single and multiple examples per query using all evaluation queries. Different number of layers in the feature extractor block were frozen to train with limited data.

# of layers frozen	Single Example		Multiple Examples	
	$C_{nxe}^{\min} \downarrow$	$MTWV \uparrow$	$C_{nxe}^{\min} \downarrow$	$MTWV \uparrow$
3	0.5555	0.4328	0.5396	0.4552
2	0.5637	0.4417	0.5541	0.4557
1	0.5522	0.**4461**	0.5395	**0.4682**
0	**0.5339**	0.4412	**0.5207**	0.4654

network. However those trained models yield very poor QbE-STD performance. This can be attributed to the limited training data as well as the complexity of the problem. Thus, we start the training by initializing different blocks of the model with corresponding pre-trained network as discussed in Section 5.5.2. In order to limit the trainable parameters, we progressively freeze the first few layers of the feature extraction block and train separate networks. In this case of end-to-end training, the frame-level speech activity detection (SAD) (as discussed in Section 5.6.2) is performed on the output of feature extraction network before using them to compute the similarity matrix. It is not applied on the MFCC features in order to avoid discontinuities in the contextual input vectors.

Performance on SWS 2013

The performance of the end-to-end QbE-STD system is presented in Table 5.8. We freeze the first few layers of the feature extractor while keeping the rest of network trainable and show the corresponding results. Similar to previously presented systems, we use template averaging to obtain the template for queries with multiple examples. However, the template averaging is performed after the query examples are forward passed through the feature extractor. We can see from Table 5.8 that the best performance is obtained by training all layers of the feature extractor. It shows that the problem of limited training data can be alleviated by pre-training different parts of the network.

Performance on QUESST 2014

The generalization ability of the models trained on SWS 2013 is evaluated using QUESST 2014 database. The Qbe-STD performance of those models on QUESST 2014 is presented in Table 5.9. We observe that T1 queries perform best with the model trained using 2 frozen layers, whereas T2 and T3 queries perform best with the model trained using 1 frozen layer. It can be attributed to the training of the models using SWS 2013, which enables the network to optimize for that database when fine-tuning all layers of the feature extractor.

Table 5.9: Performance of the End-to-End neural network based approach in QUESST 2014 for different types of queries in evaluation set. Different number of layers in the feature extractor block were frozen to train with limited data.

# of layers frozen	T1 Queries		T2 Queries		T3 Queries	
	$C_{nxe}^{\min} \downarrow$	$MTWV \uparrow$	$C_{nxe}^{\min} \downarrow$	$MTWV \uparrow$	$C_{nxe}^{\min} \downarrow$	$MTWV \uparrow$
3	0.3881	0.6395	0.5238	0.4362	0.6254	0.3669
2	**0.3796**	**0.6499**	0.5158	0.4433	0.6278	0.3617
1	0.3888	0.6309	**0.5124**	**0.4513**	**0.6148**	**0.3793**
0	0.4268	0.6190	0.5338	0.4338	0.6591	0.3646

Table 5.10: Performance of the DTW based template matching approach using multilingual bottleneck features which are fine tuned using CNN based loss function. The experiments were performed using evaluation queries in SWS 2013 for single and multiple examples per query.

# of layers frozen	Single Example		Multiple Examples	
	$C_{nxe}^{\min} \downarrow$	$MTWV \uparrow$	$C_{nxe}^{\min} \downarrow$	$MTWV \uparrow$
3	0.5788	0.4633	0.5521	0.4888
2	0.5705	0.4708	0.5539	**0.4914**
1	**0.5607**	**0.4719**	**0.5429**	0.4894
0	0.5718	0.4597	0.5593	0.4738
Bottleneck	0.6204	0.4358	0.5866	0.4580

CNN based Matching as Loss Function

In the end-to-end model for QbE-STD, we can freeze the parameters of the CNN based matching network and consider the CNN as a loss function for fine tuning the feature extraction network. This loss function enables the feature extractor to learn and generate features which produce more discriminative similarity matrices to be classified by the CNN. It can be observed through the performance of the system. We use the features obtained after fine-tuning the network to perform DTW based matching and compare it with the best performance obtained using bottleneck features as shown in Section 5.6.2. Similar to previous section, we progressively freeze different number of layers of the feature extractor. The results are presented in Table 5.10. We observe that the feature extractor retrained with 1 frozen layer gives the best results which is significantly better than the bottleneck features indicating the importance of CNN based loss function.

Table 5.11: Performance comparison of *DTW based Matching*, *CNN based Matching* and *End-to-End* neural network model for QbE-STD in SWS 2013 using single and multiple examples of all evaluation queries.

System	Single Example		Multiple Examples	
	$C_{nxe}^{\min} \downarrow$	$MTWV \uparrow$	$C_{nxe}^{\min} \downarrow$	$MTWV \uparrow$
DTW based Matching	0.6204	0.4358	0.5866	0.4580
CNN based Matching	0.6078	0.3986	0.5767	0.4115
End-to-End	**0.5339**	**0.4412**	**0.5207**	**0.4654**

Table 5.12: Performance comparison of *DTW based Matching*, *CNN based Matching* and *End-to-End* neural network model for QbE-STD in QUESST 2014 using different types of queries in evaluation set.

System	T1 Queries		T2 Queries		T3 Queries	
	$C_{nxe}^{\min} \downarrow$	$MTWV \uparrow$	$C_{nxe}^{\min} \downarrow$	$MTWV \uparrow$	$C_{nxe}^{\min} \downarrow$	$MTWV \uparrow$
DTW based Matching	0.4606	0.5663	0.6013	0.3605	0.7601	0.2138
CNN based Matching	0.4121	0.6103	0.5235	0.4375	0.6569	0.3603
End-to-End	**0.3796**	**0.6499**	**0.5158**	**0.4433**	**0.6278**	**0.3617**

5.6.5 System Comparisons

In this section, we compare the three systems for QbE-STD presented in this chapter: (i) *DTW based Matching* of bottleneck features, (ii) *CNN based Matching* of bottleneck features and (iii) *End-to-End* neural network model using different metrics as discussed in the following.

$C_{nxe}^{\min}$ and $MTWV$ scores

The performance comparisons using $C_{nxe}^{\min}$ and $MTWV$ scores corresponding to SWS 2013 and QUESST 2014 databases are presented Tables 5.11 and 5.12 respectively. We observe that the *CNN based Matching* performs significantly better than the *DTW based Matching* and the *End-to-End* system performs significantly better than the *DTW based Matching* in both databases in terms of $C_{nxe}^{\min}$ score.

DET curves

We also present the same system comparison using DET curves for both databases in Figures 5.5 and 5.6 respectively. In case of SWS 2013 database, we compare the performance using single example per query, and for QUESST 2014 database T1 query performances are compared. In both databases the *CNN based Matching* and *End-to-End* system performs

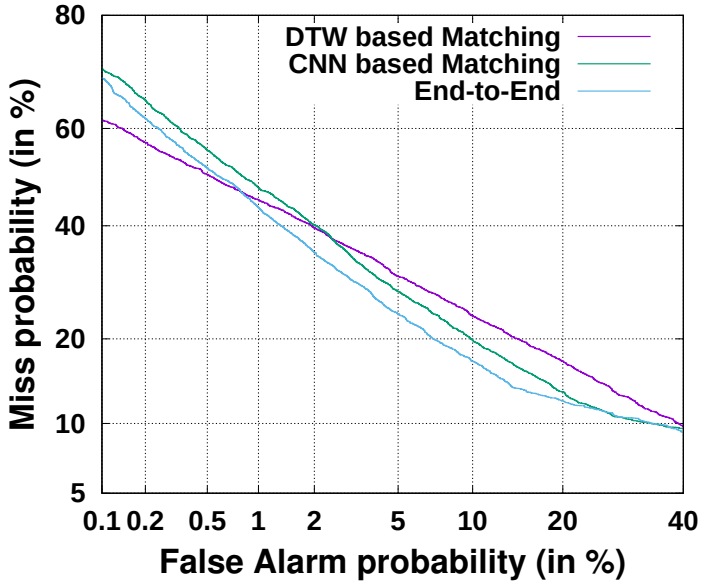

Figure 5.5: DET curves comparing the performance of *DTW based Matching, CNN based Matching* and *End-to-End* system on SWS 2013 database using evaluation queries with single example.

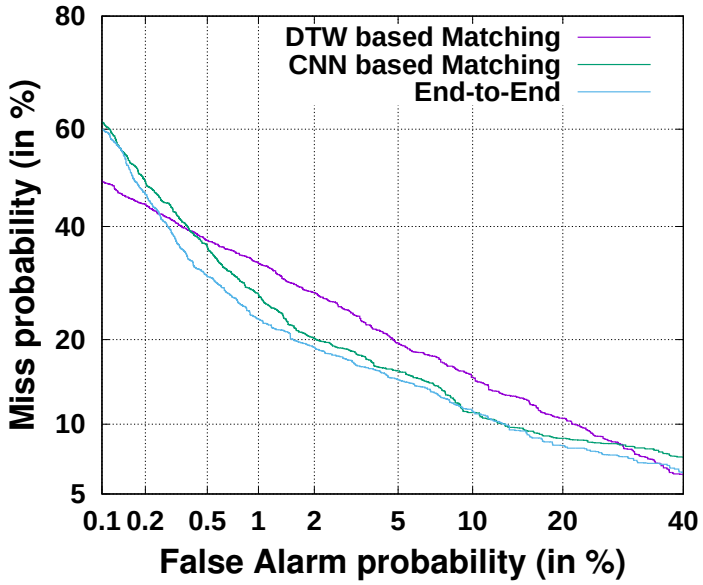

Figure 5.6: DET curves comparing the performance of *DTW based Matching, CNN based Matching* and *End-to-End* system using T1 evaluation queries of QUESST 2014 database.

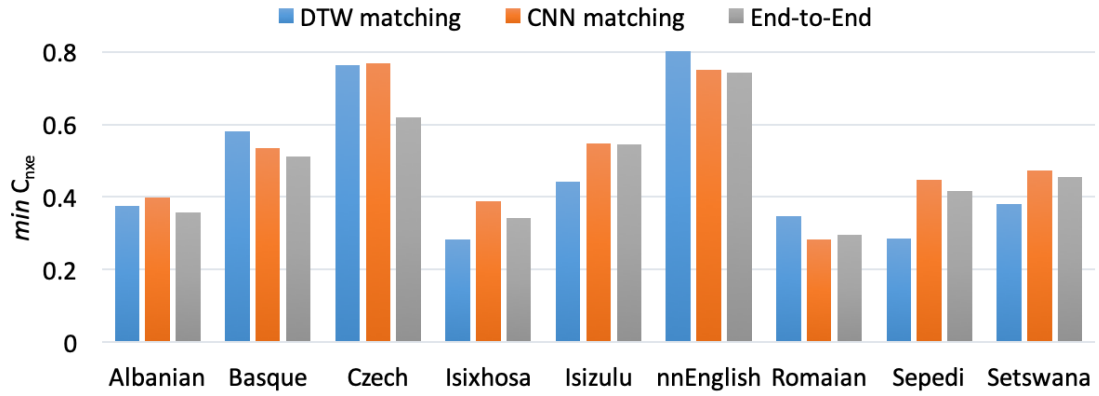

Figure 5.7: Comparison of QbE-STD performance of language specific evaluation queries (single example per query) of SWS 2013 using $C_{nxe}^{\min}$ values (lower is better)

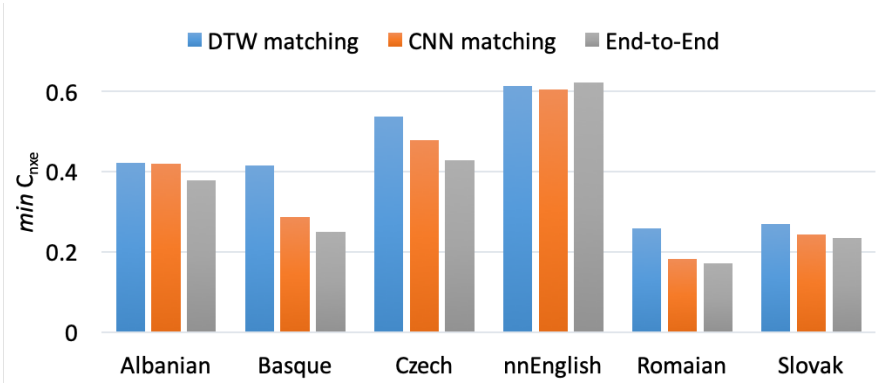

Figure 5.8: Comparison of QbE-STD performance of language specific evaluation queries (T1 query) of QUESST 2014 using $C_{nxe}^{\min}$ values (lower is better)

better than the *DTW based Matching* except for very low false alarm rates.

Language Specific Performance

Finally, we compare the language specific query performance of different systems for both databases using $C_{nxe}^{\min}$ values in Figures 5.7 and 5.8 respectively. In SWS 2013 database, the experiments are performed using single examples per query. The comparison shows that the performance of *CNN based Matching* and *End-to-End* system are noticeably worse than the *DTW based Matching* for 'Isixhosa', 'Isizulu', 'Sepedi' and 'Setswana' indicating that the performance gains are not uniform throughout different languages. This is due to the considerably less amount of training data corresponding to those languages which can be seen in Table 2.2 in Chapter 2.

In QUESST 2014 database, we compare the T1 query performances. Similar to SWS 2013 database, non-uniform performance improvement is observed for queries of different lan-

guages. The performance is marginally worse only for 'non-native English' queries in *End-to-End* system.

5.7 Conclusions

In this chapter, we implemented several monolingual as well as multilingual neural network networks to extract bottleneck features for QbE-STD and show that more training languages give better performance. Then, we proposed a novel CNN based matching approach for QbE-STD using those bottleneck features. It enables discriminative learning between positive and negative classes, which is not featured in DTW based matching systems. It gives significant improvement over the best DTW system with bottleneck features. We also propose to integrate the bottleneck feature extractor with the CNN based matching network to provide an end-to-end learning framework for QbE-STD. It gives further improvement over the CNN based matching approach. Both the CNN based matching and end-to-end system are generalizable to other database, giving significant improvement over the DTW based matching. We also show that the CNN matching block in the end-to-end system can be used as a loss function to obtain better language independent features.

6 Conclusions and Future Work

In this chapter, we summarize the conclusions of this thesis in Section 6.1 and present some possible future research directions in Section 6.2.

6.1 Conclusions

In this thesis, we addressed the problem of language independent spoken term detection in zero resource scenario. We investigated sparse modeling and neural network based approaches to improve the performance over state-of-the-art.

We exploited the low-dimensional subspace structures of speech signal to obtain better posterior features for QbE-STD. We used dictionary learning to model the phonetic subspaces in an unsupervised manner and use them to enhance the phone posteriors. We also employed DNNs to model the sub-phonetic attributes using corresponding labels, resulting in phonological posteriors. Phone and phonological posteriors were shown to capture complementary information. Hence, the use of a distance fusion technique to combine those two information sources clearly results in significant performance improvement.

It was then shown that query matching can be casted as a subspace detection problem. In this case, the query subspaces are modeled using dictionaries for sparse representation and frame-level reconstruction errors are used to detect a query in a test utterance. This approach is shown to be much faster than the DTW algorithm, however the detection performance was worse. Thus, the reconstruction errors are used to improve the DTW based matching score in two ways: (i) regularizing the distance matrix for DTW and (ii) re-scoring the DTW based score. Both approaches yield significant improvement over the state-of-the-art.

We employ several monolingual as well as multilingual neural network to extract features for query detection using DTW and show that more training languages give better performance. The DTW-based matching was replaced by CNN-based matching to introduce a learning framework for QbE-STD, yielding further improvements. In this case, the DTW distance matrix is viewed as an image which contains a quasi-diagonal pattern if the query occurs in a test

utterance. Finally, we integrate the CNN matching network with the feature extractor and train the whole architecture in an end-to-end manner to gain further improvement. Both CNN based matching and end-to-end system generalizes well to a very challenging database (QUESST 2014). We also show that the CNN matching network can be used as a loss function for this whole network and better language independent features can be obtained.

6.2 Directions for Future Research

We proposed a neural network based framework to replace the DTW. This new approach has the potential to be used in other problems where DTW based systems are applicable (e.g. time series analysis). The resulting system, although performing very well compared to previous state-of-the-art, could still be improved in several ways, including:

- The CNN-based matching network has to deal with variable size images. We have proposed simple down-sampling and zero-padding approaches to deal with this problem. In the future, the system can benefit from better up-sampling and down-sampling approaches.
- We have used the information whether a query occurs in a test utterance or not as ground-truth label for that pair. Even though it is not as extensive as the corresponding transcriptions, it could still be difficult to obtain. Thus, a DTW-based matching score can be used as ground-truth for training the CNN.
- The query matching with CNN can be performed using a triplet loss function (Schroff et al., 2015) to make it more discriminative. In this case, the system processes three inputs instead of two. Each input sample consists of a test utterance with two queries, one that occurs in the test utterance and other one that does not.

We have shown that language independent features obtained from multilingual networks yield significantly better performance than the language dependent features obtained from monolingual networks. We also demonstrated that the CNN-based matching can be used as loss function to obtain more language independent features. Thus, true language independent representations are the key to improving the query detection score. Future research can be focused on achieving that goal with novel neural network architectures and loss functions.

Bibliography

Xavier Anguera. Information retrieval-based dynamic time warping. In *Seventeenth Annual Conference of the International Speech Communication Association (INTERSPEECH)*, pages 1–5, 2013.

Xavier Anguera and Miquel Ferrarons. Memory efficient subsequence DTW for query-by-example spoken term detection. In *2013 IEEE International Conference on Multimedia and Expo (ICME)*. IEEE, 2013.

Xavier Anguera, Florian Metze, Andi Buzo, Igor Szoke, and Luis J Rodriguez-Fuentes. The Spoken Web Search task. In *the MediaEval 2013 Workshop*, 2013.

Xavier Anguera, Luis J Rodriguez-Fuentes, Igor Szoke, Andi Buzo, Florian Metze, and Mikel Penagarikano. Query-by-example spoken term detection evaluation on low-resource languages. In *The 4th International Workshop on Spoken Language Technologies for Under-resourced Languages (SLTU'14)*, 2014.

Afsaneh Asaei, Milos Cernak, Hervé Bourlard, and Dhananjay Ram. Sparse pronunciation codes for perceptual phonetic information assessment. In *Workshop on Signal Processing with Adaptive Sparse Structured Representations (SPARS)*, 2017.

Afsaneh Asaei, Dhananjay Ram, and Hervé Bourlard. Phonological posterior hashing for query by example spoken term detection. *Proc. Interspeech 2018*, pages 2067–2071, 2018.

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014.

Jon Bentley. Programming pearls: algorithm design techniques. *Communications of the ACM*, 27(9):865–873, 1984.

Christopher M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag, Berlin, Heidelberg, 2006. ISBN 0387310738.

Hervé Bourlard and Nelson Morgan. Connectionist speech recognition: A hybrid approach. 1994.

Niko Brümmer and Edward De Villiers. The BOSARIS toolkit: Theory, algorithms and code for surviving the new DCF. *arXiv preprint arXiv:1304.2865*, 2013.

Rich Caruana. Multitask learning. *Machine learning*, 28(1):41–75, 1997.

Milos Cernak, Afsaneh Asaei, and Hervé Bourlard. On structured sparsity of phonological posteriors for linguistic parsing. *Speech Communication*, 84:36–45, 2016.

Milos Cernak, Stefan Benus, and Alexandros Lazaridis. Speech vocoding for laboratory phonology. *Computer Speech & Language*, 42:100–121, 2017.

Guoguo Chen, Carolina Parada, and Tara N Sainath. Query-by-example keyword spotting using long short-term memory networks. *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2015.

H. Chen, C. C. Leung, L. Xie, B. Ma, and H. Li. Multitask feature learning for low-resource query-by-example spoken term detection. *IEEE Journal of Selected Topics in Signal Processing*, 11 (8):1329–1339, Dec 2017. ISSN 1932-4553. doi: 10.1109/JSTSP.2017.2764270.

Yi Chen, Nasser M Nasrabadi, and Trac D Tran. Sparse representation for target detection in hyperspectral imagery. *Selected Topics in Signal Processing, IEEE Journal of*, 5(3):629–640, 2011.

Tee Kiah Chia, Khe Chai Sim, Haizhou Li, and Hwee Tou Ng. A lattice-based approach to query-by-example spoken document retrieval. In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 363–370. ACM, 2008.

N. Chomsky and M. Halle. *The Sound Pattern of English*. Harper & Row, 1968.

George N Clements. The geometry of phonological features. *Phonology*, 2(01):225–252, 1985.

Steven Davis and Paul Mermelstein. Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences. *IEEE transactions on acoustics, speech, and signal processing*, 28(4):357–366, 1980.

Li Deng. Switching dynamic system models for speech articulation and acoustics. In *Mathematical Foundations of Speech and Language Processing*, pages 115–133. Springer New York, 2004.

Pranay Dighe, Afsaneh Asaei, and Hervé Bourlard. Sparse modeling of neural network posterior probabilities for exemplar-based speech recognition. *Speech Communication*, 76:230–244, 2016a.

Pranay Dighe, Gil Luyet, Afsaneh Asaei, and Hervé Bourlard. Exploiting low-dimensional structures to enhance DNN based acoustic modeling in speech recognition. In *Acoustics, Speech and Signal Processing (ICASSP), IEEE International Conference on*, 2016b.

Pranay Dighe, Gil Luyet, Afsaneh Asaei, and Herve Bourlard. Exploiting low-dimensional structures to enhance DNN based acoustic modeling in speech recognition. In *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5690–5694. IEEE, 2016c.

Ehsan Elhamifar and Rene Vidal. Sparse subspace clustering: Algorithm, theory, and applications. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 35(11):2765–2781, 2013.

Jort F Gemmeke, Tuomas Virtanen, and Antti Hurmalainen. Exemplar-based sparse representations for noise robust automatic speech recognition. *Audio, Speech, and Language Processing, IEEE Transactions on*, 19(7):2067–2080, 2011.

Alex Graves, Abdel-rahman Mohamed, and Geoffrey Hinton. Speech recognition with deep recurrent neural networks. In *2013 IEEE international conference on acoustics, speech and signal processing*, pages 6645–6649. IEEE, 2013.

Timothy J Hazen, Wade Shen, and Christopher White. Query-by-example spoken term detection using phonetic posteriorgram templates. In *Automatic Speech Recognition & Understanding, 2009. ASRU 2009. IEEE Workshop on*, pages 421–426. IEEE, 2009.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

Geoffrey Hinton, Li Deng, Dong Yu, George E Dahl, Abdel-rahman Mohamed, Navdeep Jaitly, Andrew Senior, Vincent Vanhoucke, Patrick Nguyen, Tara N Sainath, et al. Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *Signal Processing Magazine, IEEE*, 29(6):82–97, 2012.

Geoffrey E Hinton and Ruslan R Salakhutdinov. Reducing the dimensionality of data with neural networks. *science*, 313(5786):504–507, 2006.

Aren Jansen and Partha Niyogi. Intrinsic spectral analysis. *IEEE Transactions on Signal Processing*, 61(7):1698–1710, 2013.

Damianos Karakos, Richard Schwartz, Stavros Tsakalidis, Le Zhang, Shivesh Ranjan, Tim Tim Ng, Roger Hsiao, Guruprasad Saikumar, Ivan Bulyko, Long Nguyen, et al. Score normalization and system combination for improved keyword spotting. In *Automatic Speech Recognition and Understanding (ASRU), 2013 IEEE Workshop on*, pages 210–215. IEEE, 2013.

Simon King, Joe Frankel, Karen Livescu, Erik McDermott, Korin Richmond, and Mirjam Wester. Speech production knowledge in automatic speech recognition. *The Journal of the Acoustical Society of America*, 121(2):723–742, 2007.

Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

Chin-Hui Lee and Sabato Marco Siniscalchi. An information-extraction approach to speech processing: Analysis, detection, verification, and recognition. *Proceedings of the IEEE*, 101 (5):1089–1115, 2013.

Lin-shan Lee and Yi-cheng Pan. Voice-based information retrieval-how far are we from the text-based information retrieval? In *Automatic Speech Recognition & Understanding, 2009. ASRU 2009. IEEE Workshop on*, pages 26–43. IEEE, 2009.

Gil Luyet, Pranay Dighe, Afsaneh Asaei, and Hervé Bourlard. Low-rank representation of nearest neighbor phone posterior probabilities to enhance DNN acoustic modeling. In *Seventeenth Annual Conference of the International Speech Communication Association (INTERSPEECH)*, 2016.

Julien Mairal, Francis Bach, Jean Ponce, and Guillermo Sapiro. Online learning for matrix factorization and sparse coding. *Journal of Machine Learning Research (JMLR)*, 11:19–60, 2010.

Anupam Mandal, KR Prasanna Kumar, and Pabitra Mitra. Recent developments in spoken term detection: a survey. *International Journal of Speech Technology*, pages 1–16, 2013.

Alvin Martin, George Doddington, Terri Kamm, Mark Ordowski, and Mark Przybocki. The DET curve in assessment of detection task performance. Technical report, National Inst of Standards and Technology Gaithersburg MD, 1997.

John J McCarthy. Feature geometry and dependency: A review. *Phonetica*, 45(2-4):84–108, 1988.

Iain McCowan, Jean Carletta, W Kraaij, S Ashby, S Bourban, M Flynn, M Guillemot, T Hain, J Kadlec, V Karaiskos, et al. The AMI meeting corpus. In *Proceedings of the 5th International Conference on Methods and Techniques in Behavioral Research*, volume 88, 2005.

Lesly Miculicich, Dhananjay Ram, Nikolaos Pappas, and James Henderson. Document-level neural machine translation with hierarchical attention networks. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2947–2954, Brussels, Belgium, October-November 2018. Association for Computational Linguistics.

Meinard Müller. *Information retrieval for music and motion*. Springer, 2007.

Carolina Parada, Abhinav Sethy, and Bhuvana Ramabhadran. Query-by-example spoken term detection for OOV terms. In *IEEE Workshop on Automatic Speech Recognition & Understanding (ASRU)*, pages 404–409, 2009.

Alex S Park and James R Glass. Unsupervised pattern discovery in speech. *IEEE Transactions on Audio, Speech, and Language Processing*, 16(1):186–197, 2008.

Adam Paszke, Sam Gross, and Soumith Chintala. Pytorch, 2017. [online] http://pytorch.org/.

Douglas B Paul and Janet M Baker. The design for the wall street journal-based csr corpus. In *Proceedings of the workshop on Speech and Natural Language*, pages 357–362. Association for Computational Linguistics, 1992.

Petr Pollák, Jerome Boudy, Khalid Choukri, Henk Van Den Heuvel, Klara Vicsi, Attila Virag, Rainer Siemund, Wojciech Majewski, Piotr Staroniewicz, Herbert Tropf, et al. SpeechDat(E) - Eastern European telephone speech databases. In *the Proc. of XLDB 2000, Workshop on Very Large Telephone Speech Databases*. Citeseer, 2000.

Daniel Povey, Arnab Ghoshal, Gilles Boulianne, Lukas Burget, Ondrej Glembek, Nagendra Goel, Mirko Hannemann, Petr Motlicek, Yanmin Qian, Petr Schwarz, et al. The Kaldi speech recognition toolkit. In *IEEE 2011 workshop on automatic speech recognition and understanding*. IEEE Signal Processing Society, 2011.

Lorien Y Pratt, Jack Mostow, Candace A Kamm, and Ace A Kamm. Direct transfer of learned information among neural networks. In *AAAI*, volume 91, pages 584–589, 1991.

Lawrence R Rabiner, Aaron E Rosenberg, and Stephen E Levinson. Considerations in dynamic time warping algorithms for discrete word recognition. *The Journal of the Acoustical Society of America*, 63(S1):S79–S79, 1978.

Dhananjay Ram, Afsaneh Asaei, Pranay Dighe, and Hervé Bourlard. Sparse modeling of posterior exemplars for keyword detection. In *Sixteenth Annual Conference of the International Speech Communication Association (INTERSPEECH)*, 2015.

Dhananjay Ram, Afsaneh Asaei, and Hervé Bourlard. Subspace detection of DNN posterior probabilities via sparse representation for query by example spoken term detection. In *Seventeenth Annual Conference of the International Speech Communication Association (INTERSPEECH)*, 2016.

Dhananjay Ram, Afsaneh Asaei, and Hervé Bourlard. Subspace regularized dynamic time warping for spoken query detection. In *Workshop on Signal Processing with Adaptive Sparse Structured Representations (SPARS)*, 2017.

Dhananjay Ram, Afsaneh Asaei, and Hervé Bourlard. Phonetic subspace features for improved query by example spoken term detection. *Speech Communication*, 103:27–36, 2018a.

Dhananjay Ram, Afsaneh Asaei, and Hervé Bourlard. Sparse subspace modeling for query by example spoken term detection. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 26(6):1126–1139, June 2018b. ISSN 2329-9290. doi: 10.1109/TASLP.2018.2815780.

Dhananjay Ram, Lesly Miculicich, and Hervé Bourlard. CNN based query by example spoken term detection. In *Nineteenth Annual Conference of the International Speech Communication Association (INTERSPEECH)*, 2018c.

Dhananjay Ram, Lesly Miculicich, and Hervé Bourlard. Multilingual bottleneck features for query by example spoken term detection. In *Automatic Speech Recognition & Understanding, ASRU. IEEE Workshop on*, 2019a.

Dhananjay Ram, Lesly Miculicich, and Hervé Bourlard. Neural network based end-to-end query by example spoken term detection. *arXiv preprint arXiv:1911.08332*, 2019b.

Irina Rish and Genady Grabarnik. *Sparse modeling: theory, algorithms, and applications*. CRC press, 2014.

Luis J Rodriguez-Fuentes and Mikel Penagarikano. Mediaeval 2013 spoken web search task: system performance measures. *n. TR-2013-1, Department of Electricity and Electronics, University of the Basque Country*, 2013.

Luis Javier Rodríguez-Fuentes, Niko Brümmer, Mikel Penagarikano, Amparo Varona, Mireia Diez, and Germán Bordel. The albayzin 2012 language recognition evaluation plan (albayzin 2012 lre). *URL: http://iberspeech2012. ii. uam. es/images/PDFs/albayzin_lre12_evalplan_v1. 3_springer. pdf*, 2012.

Luis Javier Rodriguez-Fuentes, Amparo Varona, Mike Penagarikano, Germán Bordel, and Mireia Diez. High-performance query-by-example spoken term detection on the SWS 2013 evaluation. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7819–7823, 2014.

Luis Javier Rodríguez-Fuentes, Amparo Varona, Mikel Penagarikano, Germán Bordel, and Mireia Diez. GTTS-EHU systems for QUESST at mediaeval 2014. In *MediaEval*, 2014.

Reza Sahraeian, Dirk Van Compernolle, and Febe de Wet. Under-resourced speech recognition based on the speech manifold. In *INTERSPEECH*, pages 1255–1259, 2015.

Tara N Sainath, Bhuvana Ramabhadran, Michael Picheny, David Nahamoo, and Dimitri Kanevsky. Exemplar-based sparse representation features: From TIMIT to LVCSR. *Audio, Speech, and Language Processing, IEEE Transactions on*, 19(8):2598–2613, 2011.

Hiroaki Sakoe and Seibi Chiba. Dynamic programming algorithm optimization for spoken word recognition. *IEEE transactions on acoustics, speech, and signal processing*, 26(1):43–49, 1978.

Florian Schroff, Dmitry Kalenichenko, and James Philbin. Facenet: A unified embedding for face recognition and clustering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 815–823, 2015.

Tanja Schultz, Ngoc Thang Vu, and Tim Schlippe. Globalphone: A multilingual text & speech database in 20 languages. In *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 8126–8130. IEEE, 2013.

Petr Schwarz. *Phoneme recognition based on long temporal context*. PhD thesis, Faculty of Information Technology BUT, 2008.

Wade Shen, Christopher M White, and Timothy J Hazen. A comparison of query-by-example methods for spoken term detection. Technical report, DTIC Document, 2009.

Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

Pablo Sprechmann, Ignacio Ramirez, Guillermo Sapiro, and Yonina C Eldar. C-HiLasso: A collaborative hierarchical sparse modeling framework. *Signal Processing, IEEE Transactions on*, 59(9):4183–4198, 2011.

Igor Szoke, Miroslav Skacel, and Lukas Burget. BUT QUESST 2014 system description. In *MediaEval*, 2014.

Igor Szoke, Miroslav Skacel, Lukas Burget, and Jan Cernocky. Coping with channel mismatch in query-by-example-but quesst 2014. In *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5838–5842. IEEE, 2015.

Robert Tibshirani. Regression shrinkage and selection via the Lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 267–288, 1996.

Karel Veselỳ, Martin Karafiát, Frantivsek Grézl, Milovs Janda, and Ekaterina Egorova. The language-independent bottleneck features. In *2012 IEEE Spoken Language Technology Workshop (SLT)*, pages 336–341. IEEE, 2012.

H. Wang, T. Lee, C. C. Leung, B. Ma, and H. Li. Using parallel tokenizers with DTW matrix combination for low-resource spoken term detection. In *IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 8545–8549, May 2013. doi: 10.1109/ICASSP.2013.6639333.

Yun Wang and Florian Metze. An in-depth comparison of keyword specific thresholding and sum-to-one score normalization. In *Fifteenth Annual Conference of the International Speech Communication Association (INTERSPEECH)*, 2014.

John Wright, Allen Y Yang, Arvind Ganesh, Shankar S Sastry, and Yi Ma. Robust face recognition via sparse representation. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 31(2):210–227, 2009.

Dong Yu and Michael L Seltzer. Improved bottleneck features using pretrained deep neural networks. In *Twelfth annual conference of the international speech communication association*, 2011.

Yaodong Zhang and James R Glass. Unsupervised spoken keyword spotting via segmental DTW on gaussian posteriorgrams. In *Automatic Speech Recognition & Understanding, 2009. ASRU 2009. IEEE Workshop on*, pages 398–403. IEEE, 2009.

Yaodong Zhang, Ruslan Salakhutdinov, Hung-An Chang, and James Glass. Resource configurable spoken query detection using deep boltzmann machines. In *IEEE International*

Conference on Acoustics, Speech and Signal Processing (ICASSP), pages 5161–5164. IEEE, 2012.

DHANANJAY RAM

dhananjay.iem@gmail.com

EDUCATION

École Polytechnique Fédérale de Lausanne (EPFL), Switzerland *2015 - 2019 (expected)*
Ph.D (Pursuing), Electrical Engineering

Indian Institute of Technology Kanpur (IIT Kanpur), India *2011 - 2013*
M.Tech, Electrical Engineering
Specialization in Signal Processing, Communication & Networks

Institute of Engineering & Management (IEM), Kolkata, India *2007 - 2011*
West Bengal University of Technology
B.Tech, Electronics & Communication Engineering

ACADEMIC RESEARCH EXPERIENCE

Idiap Research Institute, Switzerland *Jan 2015 - Present*
Research Assistant

· Supervisor : Prof. Hervé Bourlard
Title : Language Independent Query by Example Spoken Term Detection (QbE-STD)
Objective : The project aims at developing a QbE-STD system in zero-resource scenario. We investigated sparse representation and deep learning based systems to improve over the state-of-the-art dynamic time warping (DTW) approach.

Idiap Research Institute, Switzerland *Jul 2014 - Dec 2014*
Research Intern

· Supervisor : Dr. Petr Motlicek
Title : Integration of Real-Time Speech Processing Technologies for Online Gaming
Objective : We developed "Who wants to be a millionaire" like game using **speech recognition** and text-to-speech technologies as a proof of concept for an interactive application for multilingual collaboration between users.

Indian Institute of Technology Kanpur, India *Sep 2013 - May 2014*
Project Engineer

· Supervisor : Dr. Ketan Rajawat, IIT Kanpur
Title : Multi-Point Spectrum Sensing using Factor Graph
Objective : We developed a cooperative spectrum sensing framework for multiple primary user cognitive radio network using factor graph.

M.Tech Thesis *Jun 2012 - Jul 2013*

· Supervisors : Dr. Rajesh M. Hegde and Dr. Debasis Kundu, IIT Kanpur
Title : A Bayesian Approach To Estimation of **Speaker Normalization** Parameters
Objective : Normalization of formant frequencies of the subject speaker by describing the relation between formant frequency vectors of subject and a reference speaker giving us a $\sim$ **6% relative improvement** over vocal tract length normalization (VTLN).

INDUSTRIAL RESEARCH EXPERIENCE

Amazon.com Inc, Boston, USA *May 2018 - Aug 2018*
Applied Scientist Intern

· Objective : Wake-word Detection

Microsoft Research, Redmond, USA *Sep 2018 - Nov 2018*
Research Intern

· Objective : Speech Source Localization, Voice Activity Detection for mixed source signal

TECHNICAL STRENGTHS

Programming Languages	Python
Speech Recognition Tool	Kaldi
Deep Learning Tool	Pytorch, Tensorflow
Numerical Computing Tool	MATLAB
Operating systems	Linux, Windows

SCHOLASTIC ACHIEVEMENTS

1. Stood among **top 4%** students in the Graduate Aptitude Test in Engineering organized by IITs *2011*
2. Received **full scholarship** from Govt. of India for studying M.Tech in IIT Kanpur *2011*
3. Finalist in the event 'Softronix' in Festronix'09, the technical festival of IEM Kolkata *2009*
4. Stood among **top 1%** students in the West Bengal Joint Entrance Examination *2007*
5. Won the sports quiz competition in Burdwan Town School *2004*

PUBLICATIONS

1. D. Ram, L. Miculicich, H. Bourlard, "Multilingual Bottleneck Features for Query by Example Spoken Term Detection", Submitted to IEEE Automatic Speech Recognition and Understanding (ASRU) Workshop, 2019
2. D. Ram, L. Miculicich, H. Bourlard, "CNN based Query by Example Spoken Term Detection", Nineteenth Annual Conference of the International Speech Communication Association (INTERSPEECH), 2018
3. A. Asaei, D. Ram, and H. Bourlard, "Phonological Posterior Hashing for Query by Example Spoken Term Detection", Nineteenth Annual Conference of the International Speech Communication Association (INTERSPEECH), 2018.
4. D. Ram, A. Asaei, and H. Bourlard, "Phonetic Subspace Representation for Spoken Query Retrieval", Speech Communication, 2018
5. D. Ram, A. Asaei, and H. Bourlard, "Sparse Subspace Modeling for Query by Example Spoken Term Detection", in IEEE/ACM Transactions on Audio, Speech, and Language Processing, 2018.
6. L. Miculicich, D. Ram, N. Pappas, and J. Henderson, "Document-Level Neural Machine Translation with Hierarchical Attention Networks", in Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP), 2018
7. L. Miculicich, N. Pappas, D. Ram, A. Popescu-Belis, "Self-Attentive Residual Decoder for Neural Machine Translation", Annual Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, 2018
8. D. Ram, A. Asaei, and H. Bourlard, "Subspace Regularized Dynamic Time Warping for Spoken Query Detection", in Workshop on Signal Processing with Adaptive Sparse Structured Representations (SPARS), 2017
9. A. Asaei, M. Cernak, H. Bourlard and D.Ram, "Sparse Pronunciation Codes for Perceptual Phonetic Information Assessment", in Workshop on Signal Processing with Adaptive Sparse Structured Representations (SPARS), 2017
10. D. Ram, A. Asaei, and H. Bourlard, "Subspace Detection of DNN Posterior Probabilities via Sparse Representation for Query by Example Spoken Term Detection", Seventeenth Annual Conference of the International Speech Communication Association (INTERSPEECH), 2016.
11. D. Ram, D. Kundu, and R. M. Hegde, "A Bayesian Approach to Estimation of Speaker Normalization Parameters", arXiv preprint arXiv:1610.05948 (2016).
12. D. Ram, A. Asaei, P. Dighe, and H. Bourlard, "Sparse modeling of posterior exemplars for keyword detection", in Sixteenth Annual Conference of the International Speech Communication Association (INTERSPEECH), 2015.
13. D. Ram, D. Kundu, and R. M. Hegde, "A Bayesian approach to speaker normalization using vowel formant frequency", Communications (NCC), 2014 Twentieth National Conference on , vol., no., pp.1,6, Feb. 28 2014-March 2 2014.