Acoustic Model Selection for Recognition of Regional Accented Speech

Maryam Najafian

Department of Electrical, Electronic and Computer Engineering

University of Birmingham

This dissertation is submitted for the degree of

Doctor of Philosophy

December 2015

I would like to dedicate this thesis to my loving parents . . .

Declaration

I hereby declare that except where specific reference is made to the work of others, the contents of this dissertation are original and have not been submitted in whole or in part for consideration for any other degree or qualification in this, or any other university. This dissertation is my own work and contains nothing which is the outcome of work done in collaboration with others, except as specified in the text and Acknowledgements. This dissertation contains fewer than 65,000 words including appendices, bibliography, footnotes, tables and equations and has fewer than 150 figures.

Maryam Najafian
December 2015

Acknowledgements

I would like to express my sincere gratitude towards my supervisor Professor Martin Russell for his excellent support, invaluable guidance, and kind encouragement throughout my undergraduate and postgraduate studies. This research would never have been possible without his great assistance and support.

Many thanks also goes to members of the Apple Siri speech group, in particular Melvyn Hunt, John Bridle, Matthew Seigel, and Shoichiro Yamanishi for their kind support and guidance during my internship.

I am also very grateful to be surrounded by great friends who have provided moral support during this time, so another big thanks goes to my friends and colleagues from the University of Birmingham Jingjing, Roozbeh, Emilie, Linxue, Saeid, Chloe, Eva, Xizi, Marius, Ahmed, William, James, Edward, Mohammad abbas, and Dimitris for a great time we spent together.

I would like to thank Philip Weber, Michael Carey, Stephen Houghton, Soud Hanani, Costas Constantinou, Peter Jancovic, Phil Atkins from the University of Birmingham, Simon King from the University of Edinburgh, Angel Mario Castro Martinez from the University of Oldenburg, Peter Bell from the University of Edinburgh, Andrea DeMarco and Stephen Cox from University of East Anglia, Michael Tjalve from Microsoft research lab, Daniel Povey from Johns Hopkins University, Karel Vesely from the Brno University of Technology, Bagher BabaAli from the University of Tehran, Erfan Loweimi from the University of Sheffield, Shahab Jalalvand from University of Trento, Amir Hossein Rezaie from Amirkabir Univerisity for their helpful support, encouragement and discussions.

Thanks also to the researchers which I have met in various conferences and tutorials during my research. A special thank goes to Li Deng from Microsoft research lab for discussing ideas, his great tutorials and publications. I would also like to thank Michiel Bacchiani, Mike Schuster, Heiga Zen, Andrew Senior, and Navdeep Jaitly from Google speech research labs, Frank Seide and Dong Yu from Microsoft speech research labs, Michael Stoss from Voice2choice, and Steve Renals from the University of Edinburgh.

Last but not least, a special thanks to my parents, my brother Mohammad, my uncles and aunts for their emotional support and constant encouragement. This thesis is especially dedicated to my wonderful family.

Abstract

Accent is cited as an issue for speech recognition systems [1]. Research has shown that accent mismatch between the training and the test data will result in significant accuracy reduction in Automatic Speech Recognition (ASR) systems. Using HMM based ASR trained on a standard English accent, our study shows that the error rates can be up to seven times higher for accented speech, than for standard English. Hence the development of accent-robust ASR systems is of significant importance.

This research investigates different acoustic modelling techniques for compensating for the effects of regional accents on the performance of ASR systems. The study includes conventional Gaussian Mixture Model-Hidden Markov Model (GMM-HMM) and more contemporary Deep Neural Network (DNN)-HMM systems. In both cases we consider both supervised and unsupervised techniques. This work uses the WSJCAM0 corpus as a set of 'accent neutral' data and accented data from the Accents of the British Isles (ABI) corpora.

Initially, we investigated a model selection approach, based on automatic accent identification (AID). Three AID systems were developed and evaluated in this work, namely i-vector, phonotactic, and ACCDIST-SVM. Each focuses on a different property of speech to achieve AID. We use two-dimensional projections based on Expectation Maximization-Principal Component Analysis (EM-PCA) and Linear Discriminative Analysis (LDA) to visualise the different accent spaces and use these visualisations to analyse the AID and ASR results.

In GMM-HMM based ASR systems, we show that using a small amount of data from a test speaker to select an accented acoustic model using AID, results in superior performance compared to that obtained with unsupervised or supervised speaker adaptation. A possible objection to AID-based model selection is that in each accent there exist speakers who have varying degrees of accent, or whose accent exhibits properties of other accents. This motived us to investigated whether using an acoustic model created based on neighbouring speakers in the accent space can result in better performance. In conclusion, the maximum reduction in error rate achieved over all GMM-HMM based adaptation approaches is obtained by using AID to select an accent-specifc model followed by speaker adaptation. It is also shown that the accuracy of an AID system does not have a high impact on the gain obtained by accent

adaptation. Hence, in real time applications one can use a simple AID system for accented acoustic model selection.

Recently, HMM systems based on Deep Neural Networks (DNNs) have achieved superior performance compared to more traditional GMM-HMM systems, due to their discriminative learning ability. Our research confirms this by showing that a DNN-HMM system outperforms a GMM-HMM system, even after the latter has benefited from two stages of accent followed by speaker adaptation. We investigate the effect of adding different types of accented data to the baseline training set. The addition of data is either supervised or unsupervised, depending on whether the added data corresponds to the accent of the test speaker. Our results show that overall accuracy of the DNN-HMM system on accented data is maximized when either the accent diversity of the supplementary training data is highest, or data from the most 'difficult' accent groups is included in the training set.

Finally, the performance of the baseline DNN-HMM system on accented data prompts an investigation of the accent characteristics of the WSJCAM0 corpus, which suggests that instead of being 'neutral' it contains speech that exhibits characteristics of many of the accents in the ABI corpus.

Table of contents

List of figures

List of tables

Chapter 1

Introduction

1.1 Introduction

Speech is one of the primary means of communication among humans. In recent years, improvements in speech technology has resulted in a considerable enhancement in human-human and human-machine communication and has become one of the most reliable and favourable tools for interaction. There are many examples of Automatic Speech Recognition (ASR) applied to improve human-human communications and human-machine interaction. For instance, speech to speech translation systems are able to enhance communication between humans speaking in different languages (e.g. s2s translation system by Microsoft).

Other examples of ASR technology improving human communication are found in transcribing callers' messages into text (e.g. Google's voice help), indexing and transcribing lectures and meetings (e.g. YouTube's automatic captioning). Examples of popular applications of ASR for human-machine communication can be found in voice search in mobile phones and cloud-based voice services (e.g. Amazon echo, Android phone, iPhone, Windows phone), wearable devices (e.g. Apple's iWatch, Google's glass), gaming (e.g. Microsoft Xbox games), and smart home devices (e.g. Google's Weave to control devices), where communicating through speech is a more preferable modality than a keyboard [5]. There has been a great improvement in ASR performance during the past few years [7]. Despite all the recent advances in speech technology, often these systems struggle with issues caused by speech variabilities. Such variabilities in speech can occur due to speaker-dependent characteristics (for example shape of the vocal tract, age, gender, or health), environmental noise, channel distortion (for example recording conditions and microphone), speaking rate (for example changes in timing and realisation of phonemes) and speaking style (read speech versus spontaneous speech), accent variabilities (regional accents, or non-native accents) [5, 8].

One of the most common issues in ASR systems is due to accent variability of the users which results in dramatic reduction in recognition accuracy [1, 9]. In fact, accent variabilities in speech are often associated with a shift within the feature space [10]. Statistical analysis techniques have shown that the second most important principal component of variation corresponds to accent properties of the speaker (foreign accented or native regionally accented speech) [11]. The development of accent-robust ASR is of significant practical importance.

Various techniques have been proposed in the literature for addressing the problems caused by regional accents in the context of Hidden Markov Model (HMM) based ASR systems. These techniques can be categorised in four broad groups, namely front-end techniques, acoustic modelling techniques, pronunciation modelling techniques, or a combination of different techniques.

The focus of this work is on addressing the acoustic variabilities caused by regional accents of the British Isles using an acoustic modelling approach. In this thesis we investigate different strategies to improve the Gaussian Mixture Model (GMM)-HMM and Deep Neural Network (DNN)-HMM based acoustic models using limited data.

1.2 Scope of this thesis

We evaluate the effect of accented speech from 14 regions of the British Isles (from the ABI-1 corpus [12]) on the performance of a baseline HMM based speech recogniser trained on the WSJCAM0 corpus [13].

The main goal of this research is to address the problems caused by regional accents faced by ASR systems at the acoustic model level. Although the focus of this research is only at the acoustic model level, a complete solution to the accent issues require a combination of different approaches, such as adding accent related features at the feature level, or selecting a pronunciation dictionary based on the test speaker's accent.

To gain a better understanding of the effect of different regional accents on the ASR performance, we proposed an accent space visualisation technique which represents distribution of different regional accents in a two-dimensional accent feature space. This research targets British English accents. However, it is very likely that the techniques described are applicable to accented speech in other languages.

1.3 Thesis outline

This thesis consists of the following chapters:

- **Background chapters**: Chapters 2 to 5 are the background and theory chapters.

  - Chapter 2 describes the existing approaches for addressing accent issues in ASR systems. In particular, four categories of approaches are outlined, namely front-end techniques, acoustic modelling techniques, pronunciation modelling techniques, and a combination of these techniques. Then, it describes three different Accent Identification (AID) systems, i-vector [14, 15], phonotactic, and ACCDIST-SVM which will be later used for supervised and unsupervised accented acoustic model selection. After that, it describes our proposed approach for visualisation of the accent feature space, which will be used for analysis of the AID and ASR results in our experiments in Chapters 6 to 9.
  - Chapter 3 describes four main stages of the speech recognition process, namely feature extraction, language modelling, acoustic modelling and speech decoding. It provide full description of the GMM-HMM based and hybrid DNN-HMM based acoustic modelling techniques in ASR systems. Finally, it describes two popular acoustic model adaptation techniques in GMM-HMM based systems, namely Maximum Likelihood Linear Regression (MLLR) and Maximum A Posteriori (MAP). Both MAP and MLLR are used for speaker and accent adaptation in Chapter 6.
  - Chapter 4 describes the DNN background, and the error back-propagation with Stochastic Gradient Descent (SGD) for training the DNN model parameters. Finally, it presents a popular generative approach to the network pre-training and model parameter initialisation referred to as the Restricted Boltzmann Machine (RBM). Both pre-training and fine-tuning algorithms are used in the DNN-HMM based acoustic modelling.
  - Chapter 5 contains the full description of the speech corpora used in our AID and ASR experiments, namely the ABI speech corpus and the WSJCAM0 speech corpus.

- **Experimental chapters**: Chapters 6 to 9 describe our experiment results and contribution, and Chapter 10 provides a summary of our major contributions and suggests different ways this research can be improved in future.

- Chapter 6 reports the performance of three popular AID systems, namely i-vector, phonotactic and ACCDIST-SVM and compares their complexity and accuracy against that of methods in the literature. Our AID systems provide competitive AID accuracy and use a less complex architecture. Their simplicity makes them attractive for real-time applications such as accent-specific acoustic model or pronunciation dictionary selection. In addition to that, this chapter analyses the AID results using a two-dimensional accent space visualisation map which shows the relative distribution of different regional accents.
- Chapter 7 provides the results for different text-dependent (supervised) approaches to compensate for the effects of regional accents of British English on the GMM-HMM based ASR. We report the relative gain in performance achieved by using MAP and MLLR for the supervised speaker and accent adaptation applications. In this chapter we either use the true accent label of the speaker or the ACCDIST-SVM AID system for selecting an accented acoustic model.
- Chapter 8 presents the results for different text-independent (unsupervised) approaches to compensate for the effects of accents on the GMM-HMM based ASR. Three unsupervised AID systems are applied for the accented acoustic model selection, and their results are reported. This chapter also investigates how much data is required from the test speaker in speaker adaptation in order to achieve similar results as the accent adaptation.
- Chapter 9 provides the experiment results on a baseline DNN-HMM based ASR system, and compares the results achieved from a baseline DNN-HMM system and that of an accent followed by speaker-adapted GMM-HMM based system. Throughout this chapter we investigate the effect of adding supplementary training data with different types of accents, and various accent diversities and sizes. This supplementary data can is added in an accent dependent (based on the test speaker's accent) or in an accent-independent (regardless of the test speaker's accent).
- Chapter 10 summarises the major contributions and conclusions of the thesis and suggests future directions for improving this research.

1.4 Key contributions of this thesis

The research described in this thesis provides original contributions to the field of accented speech recognition. The major contributions can be listed as following.

- Proposed a multi-accent phonotactic AID system comprising 14 accent-specific and one accent-neutral parallel Phone Recognition followed by Language Modelling (PRLM) systems. For recognizing the accents of ABI-1 speakers, we showed that using a multi-lingual phonotactic system comprising four language-specific PRLMs (proposed by Hanani et al. [16]) rather than our multi-accent English phonotactic system reduces the AID error rate by 7.5%. Also we showed that fusing a single i-vector based and 15 phonotactic based AID systems rather than fusing 630 i-vector AID systems (proposed by Demarco et al. [17]) results in 4.5% relative AID error reduction (Chapter 6).

- Proposed an accent space visualisation approach which illustrates the relative distributions of different regional accents in the AID feature space (Chapter 2). Presented the AID accent space visualisation results for i-vector, phonotactic, and ACCDIST-SVM AID supervectors (Chapter 6). Developed an understanding of AID and ASR recognition results on multi-accented data using three accent space visualisation maps (Chapters 7 to 9).

- Demonstrated a clear effect of 14 different British regional accents on the performance of HMM based speech recognition systems (both DNN-HMMs and GMM-HMMs), and showed that for some difficult accents, such as Glaswegian, the ASR error rate can be up to seven times higher than that of a standard southern British English accent (Chapters 7 to 9).

- Given a small amount of data from the test speaker, we proposed three approaches for accent-dependent modelling in a GMM-HMM system, namely using the true accent model, choosing a model using an AID system, and building a model using data from neighbouring speakers in AID space. Showed that all three approaches provide considerable gain in ASR performance and outperform speaker adaptation significantly (Chapters 8 and 9).

- Evaluated the gain achieved using four different AID systems, namely two different i-vector systems, a phonotactic system, and an ACCDIST-SVM system for acoustic model selection in GMM-HMM systems. The results showed that the accuracy of an AID system does not have a high impact on the gain obtained by accent adaptation (Chapters 7 and 8).

- Showed that a DNN-HMM baseline outperforms an accent followed by speaker-adapted GMM-HMM system. Demonstrated that this result can further improve either by adding very small amount of data based on the speaker's accent, or by adding more accent diverse data regardless of the speaker's accent, or adding the data from only the most difficult accents (such as Glaswegian) to the training set (Chapter 9).

1.5 Publications resulting from this thesis

Some of the ideas and results in this thesis have been published in reviewed conference papers as follows:

- Maryam Najafian, Andrea DeMarco, Stephen J. Cox and Martin J. Russell, "Unsupervised Model Selection for Recognition of Regional Accented Speech." Fifteenth Annual Conference of the International Speech Communication Association (INTERSPEECH), 2014.
- Maryam Najafian, Saeid Safavi, Abualsoud Hanani and Martin J. Russell, "Acoustic model selection using limited data for accent robust speech recognition." Signal Processing Conference (EUSIPCO), 2014 Proceedings of the 22nd European, 2014.
- Maryam Najafian, Saeid Safavi, Philip Weber and Martin J. Russell, "Effect of Accent Variations in Training of HMM-DNN Acoustic Models." Submitted to ICASSP, IEEE 2016.
- Maryam Najafian, Saeid Safavi and Martin J. Russell, "Computationally efficient approach for identification of the regionally accented British English speech." Submitted to ICASSP, IEEE 2016.

Chapter 2

Accent issues in speech recognition and regional accent identification

2.1 Introduction

The speech signal contains information beyond its linguistic content, including clues to the speaker's regional accent, social background, or level of education. In the first volume of the 'Accents of English' book [18], 'accent of English' is defined as "a pattern of pronunciation used by a speaker for whom English is the native language or, more generally, by the community or social grouping to which he or she belongs". This is different from dialect which also includes the use of words or phrases that are characteristic of those regions. Dialect includes varieties of English spoken as a first language in different countries (for example, US versus Australian English), geographical variations within a country, and patterns of pronunciation associated with particular social or ethnic groups.

Our results showed that accent mismatch between training and test data can increase the recognition error rate by up to seven times for accented speech, than for standard English. If they are to be widely deployed, ASR systems must address the problems caused by regional accents to deliver consistently high performance across user populations. Hence, dealing with accent variations in speech is of high importance for speech technology.

Just as the term 'accent' covers a range of phenomena, 'accent adaptation' refers to a number of different problems in ASR. In this chapter we start by describing different approaches to address accent issues in ASR. Then we describe different AID systems, namely ACCDIST-SVM based, phonotactic based, and i-vector based AID at the acoustic level by selecting an appropriate accent-dependent acoustic model that matches the test speaker's accent.

The proposed AID systems are successfully applied to the British English regional accent recognition task (results are reported in Chapter 6), and will be used for the supervised and unsupervised acoustic model selection in Chapters 7 and 8 respectively. In Chapter 9, we used an AID system to investigate the effect of accent diversity in the training set on the DNN-HMM recognition performance.

Finally this chapter presents a new method for visualisation of the accent feature space, which will be used to understand and analyse the results of our AID and multi-accent speech recognition systems in Chapters 6 to 9.

2.2 Addressing accent issues in ASR systems

Various approaches have been proposed in the literature for accent adaptation. These techniques can be categorised in different broad groups, namely front-end techniques, acoustic modelling techniques, pronunciation modelling techniques, or a combination of these techniques.

2.2.1 Front-end techniques

The purpose of accent adaptation using front-end techniques is to add supplementary information at the feature level to improve the acoustic model ability to discriminate between different phonetic events without having to make any modification to the training process.

- **Addition of accent-discriminative acoustic features**: In addition to accent-specific acoustic model adaptation based on the test speaker's accent, Zheng et al. [19] suggested to append accent discriminative acoustic features such as the first three formants, and their amplitudes to the 39 dimensional MFCC feature vectors. Given a GMM-HMM baseline system trained on 30 hours of speech data from the Mandarin Broadcast news corpus, this approach resulted in 1.4% character error rate reduction over the baseline system trained on Wu-accented Chinese speech.

- **Adding i-vector features**: Recently, Demarco et al. [17] and Bahari et al. [20] reported that i-vectors [14, 15] which are speaker-specific features, can be successfully used to identify native accents as they contain useful accent information.

  In an initial study, Saon et el. [21] incorporated i-vectors at the feature level to adapt deep neural networks to the individual speaker. By providing the speaker's i-vectors in parallel with the regular acoustic features as the DNN's input, DNN acoustic models are adapted to the speaker's characteristics.

In another set of studies, providing i-vector features as additional input features has proved to be advantageous for the accented speech recognition. For example, by feeding 39 dimensional MFCCs concatenated with each speaker's i-vector as input to a DNN-HMM system, Guta et al. [22] achieved 1.53% Word Error Rate (WER) reduction. This system is trained on 478 hours of French TV broadcast data and tested on 8.6 hours of data from French radio and TV programmes.

Similarly, by incorporating i-vector features at the feature level, Saon et al. [21] and Miao et al. [23] achieved 10% and 7.5% relative improvement compared to the baseline DNN system using the 300-hour and 110-hour set up of the Switchboard corpus respectively.

2.2.2 Pronunciation modelling techniques

Accent specific acoustic variabilities affect acoustic realisation of utterances at the phoneme level. Using a polyphone decision tree to model accent-specific acoustic variants or an accent dependent pronunciation dictionary are beneficial as they can represent underlying accent variations at the phoneme level.

- **Using an accent dependent pronunciation dictionary**: In initial work suggested by Kat et al. [24], accent-specific pronunciation rules from the non-native Cantonese speakers of English are incorporated into the Native English dictionary with standard pronunciation rules. This approach results in 19% relative WER reduction. Similar gains in accented speech recognition performance have been achieved through accent-specific pronunciation adaptation in literature [25–30].

- **Polyphone decision tree adaptation**: The Polyphone Decision Tree (PDT) models contextual acoustic variants which are the main source of accent differences by clustering context dependent states in an ASR. A study by Nallasamy et al. [31] an existing ASR system trained on 39 hours of multi-accented Arabic data is adapted using a PDT to match the test speaker's accent which might not have been seen in the training set. In this work the Arabic training data comes from the Broadcast Conversations (BC) part of LDC GALE corpus. Using the adaptation data from two Arabic accents (3 hours from each accent), PDT adaptation obtained 7% relative WER reduction compared to the accent adapted results using MAP adaptation.

  In another study, Nallasamy et al. [32] showed that using PDT adaptation rather than accent-specific MAP adaptation achieved 13.9% relative improvement for English accent adaptation. In this study the baseline model is trained on 66 hours of WSJ1 corpus with US accent, and adaptation data consists of 3 hours of UK accented data from the WSJCAM0 corpus.

A similar study has been performed on South African English, and Kamper et al. [33] showed that by integrating accent into the HMM decision-tree triphone clustering process (allowing questions related to accent) a considerable gain can be achieved.

2.2.3 Acoustic modelling techniques

Benzeghiba et al. [8] suggested that acoustic model adaptation approaches such as Maximum A Posteriori (MAP) and Maximum Likelihood Linear Regression (MLLR) can reduce this mismatch between the training and test set variabilities.

Recently, DNN-HMM based ASR systems have emerged as a successful alternative to GMM-HMMs [7, 34, 35]. This has been suggested to be due to their discriminative learning ability, which enhances their robustness against different sources of variation in speech, for example accent [36].

The overall performance of DNN based systems on an accented ASR task is better than GMM based systems. However, Huang et al. [37] showed that even classifiers as powerful as DNNs cannot completely address the issues caused by accent variability.

- **Acoustic model adaptation in GMM-HMMs**: Oh et al. [38] suggested that in ASR systems with GMM-based acoustic models, model adaptation techniques such as MLLR (including pronunciation variation regression classes) and MAP can provide adaptation towards a target accent. Given a GMM-HMM based recogniser trained on WSJ0 English corpus (consists of 130,507 words), a study has been carried out to recognise the test data of the Korean-Spoken English (K-SEC) corpus (686 utterances,7,154 words). In this experiment a subset of the K-SEC corpus (105,152 words) was used to adapt the acoustic models. Using MLLR followed by MAP adaptation techniques the WER was reduced by 78.01% compared to the baseline.

  In another study, Kirchhoff et el. [39] reported that Egyptian Conversational Arabic (ECA) utterances are recognised successfully after a baseline ASR system trained on 40 hours of modern standard Arabic was MAP adapted to 20 hours of ECA data.

- **Use of SGMMs for acoustic modelling**: In a Subspace Gaussian Mixture Model (SGMM)-HMM based [40] system, all phonetic states share a common GMM structure, and the means and mixture weights vary in a subspace of the total parameter space. For computing the state-dependent GMMs, this compact representation will lead to a more accurate model estimation using small amount of training data compared to a conventional GMM modelling approach.

In a recent study, Motlicek et al. [41] applied SGMMs for acoustic model adaptation towards UK and US accented English speech (UK and US versions of WSJ corpora). Using SGMMs led to 8% relative improvement compared to speaker-adapted acoustic models in a GMM-HMM system.

- **Using DNN based acoustic model**: Chen et al. [42] compared the accuracy of ASR systems using a DNN-based and a GMM-based acoustic model. The results showed 27% and 47% relative gain compared to the GMM-HMM baseline for non-native and native speech recognition tasks respectively. This study is carried out using 412 hours of Chinese data with native and non-native accents (English, Japanese, Korean, Russian, Arabic, Spanish, Thai, and other accents) from the Spoken Chinese Test (SCT) which is designed for second language learners of Chinese language.

- **Model adaptation in DNN systems**: Huang et al. [9] proposed a study on British and Indian accented speech using a multi-accent DNN acoustic model with an accent-specific top layer and shared bottom hidden layers. On a mobile short message dictation task, with 1K, 10K, and 100K British and 1K, 10K, and 100K Indian accented utterances, this approach achieves 18.1%, 26.0%, and 28.5% WER reduction on British accented utterances or 16.1%, 25.4%, and 30.6% WER reduction on Indian accented utterances compared to the baseline trained on 400 hours of speech.

2.2.4 Combination of multiple techniques

A complete solution to the accent variability issues may consist of a combination of different approaches. Examples of applying complementary methods to address accent issues are listed bellow.

- **Using accent-specific acoustic models and pronunciation dictionaries**: Huang et al. [1] showed that by applying both accent-specific pronunciation dictionary adaptation and accent-dependent acoustic model adaptation using MLLR will lead to approximately 36.02% relative WER reduction compared to the baseline trained on 660 hours of Multi-accented Mandarin speech. In this study GMM-HMM based speech recognition system uses the data from three different Mandarin accents, namely Beijing (330 hours), Shanghai (220 hours) and Guangdong (110 hours).

- **Combination of model based adaptation and discriminative features**: Chen et al. [43] suggested to concatenate speaker-specific i-vectors with acoustic features as the input of a DNN-based ASR system. DNN performance is measured using 400 hours of multi-accented Mandarin speech from the Intel accented Mandarin speech recognition Corpus. In this work a DNN-HMM system shares the bottom hidden layers across all accents while the top softmax layers are accent-dependent. The shared bottom layers are used for extracting high-level cross-accent features and provide maximal knowledge sharing among different accents. The accent-specific top layers are used for recognition of speech from different accents. Their experiment results resulted in 11.8% relative improvement in %WER compared to the baseline DNN system.

2.3 Using AID systems to address accent issues in ASR

The focus of our work is on addressing the acoustic variabilities caused by British regional accents. It is possible to address accent issues in the context of Hidden Markov Model (HMM) based ASR systems, by using an Accent Identification (AID) system to select an appropriate accent-dependent acoustic model or pronunciation dictionary.

2.4 Different AID systems

The problem of automatically extracting regional accent information from a short utterances, is addressed through different AID approaches. The AID systems used in this work can be divided into two categories, namely text-dependent (supervised) and text-independent (unsupervised). A word-level transcription of the test utterance is required for the text-dependent accent identification, while this is not the case for the text-independent systems.

In this section, two of the most popular approaches to text-independent AID, namely phonotactic and i-vector, are described. Then, the ACCDIST-SVM which is one of the most successful text-dependent AID systems is introduced.

2.4.1 Phonotactic AID

The phonotactic approach to AID relies on the phone sequence distribution to recognise the accent of the speaker. The phonotactic method was initially applied to Language Identification (LID). This text-independent (unsupervised) approach is called Phone Recognition followed by Language Modelling (PRLM) [44–48].

PPRLM for accent identification

Early work on the use of a phonotactic PRLM system for accent recognition was carried out by Zissman [49]. This approach recognises the accent of the speaker from a small amount of his or her speech, based on the frequency of use of certain phonetic sequences. Later by applying Parallel PRLMs (PPRLMs) and discriminative classification techniques such as Support Vector Machine (SVM) [50, 51] and Logistic Linear Regression (LLR) [52, 53] the accuracy of phonotactic AID system has further improved [54–56]. Full description of SVM and LLR classification approaches can be found in Appendix B.1.1 and Appendix B.1.2.

In this section, we introduce the phonotactic approach. In Section 6.2.1, we report the performance of this technique in recognising the accent of ABI-1 speakers.

The PPRLM process is carried out in four stages (Figure 2.1), namely phone recognition, vectorisation, SVM classification, and LLR score fusion. The proposed PPRLM approach uses a multi-class SVM classifier and it is referred to as PPRLM-SVM.

- **Phone recognition**: Each utterance is passed through L accent specific phone recognisers (L is the number of phone recognition systems) to generate phone level transcriptions of the utterances from C accent groups (C is the number of classes). Each phone recogniser is trained on one specific accent, and produces a phone sequence per utterance.

- **Vectorisation:** Phone N-grams comprise a sequence of N consecutive phones. For each utterance the relative frequency of each pre-defined set of phone N-grams is calculated. In this way the sequence of phones corresponding to an utterance is represented as a D dimensional vector, whose i-th entry is the relative frequency of the i-th phone N-gram in the set and denoted by p_i (Equation 2.1). Here $Count(C_i)$ is the number of times the N-gram C_i occurs in the utterance. The outcome of this stage is a D-dimensional vector that represents N-gram frequencies per utterance. This high dimensional vector is referred to as the phonotactic system's supervector.

$$p_i = \frac{Count(C_i)}{\sum_{j=1}^{D} Count(C_j)} \tag{2.1}$$

- **Multi-class SVM:** Given a set of labelled training utterances from C accent groups, there are in total L accent specific PRLM systems trying to classify the test utterances into C classes. Thus, in each PRLM system the multi-class SVM produces C scores per test utterance [57]. These scores determines to what extent each supervector belongs to each accent group (using the one against all approach).

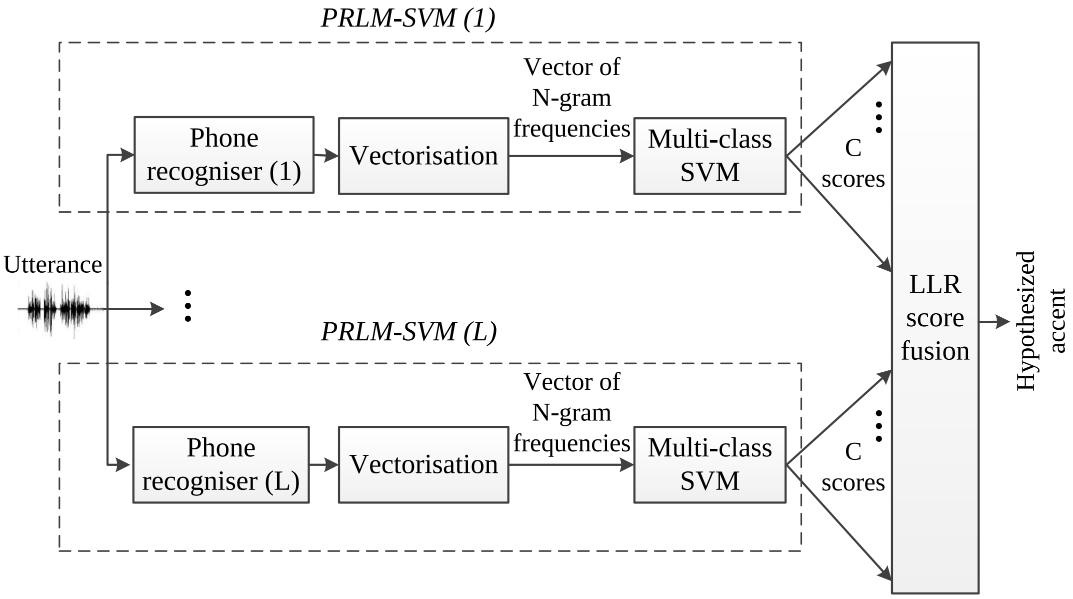

Fig. 2.1 *The SVM scores produced by L parallel PRLM-SVM systems are fused using LLR*

- **Score fusion using LLR:** To determine the test speaker's accent, the SVM scores generated by *L* different accent-specific PRLM systems are fused using the LLR approach (described in Appendix B.1.2)[58]. Figure 2.1 shows the process in which the scores from *L* accent specific PRLM-SVM systems are fused to recognise the test speakers accent.

2.4.2 I-vector based AID

I-vectors provide a low-dimensional representation of feature vectors that can be successfully used for classification and recognition tasks. I-vectors were initially introduced for speaker recognition [59, 60], and after their success in this area, they were applied to language [61, 62] and accent recognition [17, 20, 63]. The idea behind this text-independent (unsupervised) approach comes from the Joint Factor Analysis (JFA) [64–66] technique proposed for speaker verification. In JFA, speaker and channel variabilities are represented in two separate subspaces, whereas in the i-vector approach only one single space is defined for all types of variability (including both speaker and session variabilities). This is called the 'total variability' space.

The motivation for the use of a single 'total variability' space is that in JFA, traces of speaker dependent information can be found in the channel factors, and therefore separating speaker and channel variabilities will lead to losing some speaker dependent information [60]. The architecture of an i-vector system with a multi-class SVM classifier is illustrated in Figure 2.2.

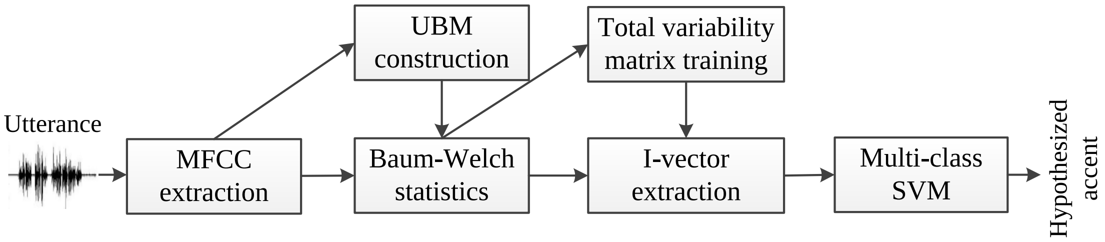

Fig. 2.2 *A summary of different stages of i-vector extraction process*

The process of building an i-vector system consists of the following stages.

- **Universal Background Model (UBM) construction**: Speech from a large population of speakers is used to estimate the parameters of a K component GMM (typically $K \geq 512$). Since this GMM models a whole population of speakers it is called the Universal Background Model (UBM). Using the EM algorithm, the UBM can represent the Speaker Independent (SI) distribution of features. During the EM iterative process the UBM model parameters $\lambda = \{c_k, \mu_k, \Sigma_k\}$ are updated for each Gaussian component $k = \{1, ..., K\}$ such that the probability of the observed data $O = \{o_1, ..., o_T\}$ given the model λ is locally maximized itteratively as shown in Equation 2.2 [67, 68]. The value of λ after the n-th iteration is denoted by $\lambda(n)$.

$$p(O|\lambda(n)) \geq p(O|\lambda(n-1)) \tag{2.2}$$

In each iteration of the EM algorithm, UBM parameters are updated. The UBM model parameters are the weight vectors, the mean vectors, and the covariance matrices which are updated using Equations 2.3, 2.4, and 2.5 respectively. The diagonal covariance matrix Σ_k models the speaker independent variability, and later we use Σ as a supercovariance matrix whose diagonal elements are the concatenation of these covariance matrices.

$$c_k = \frac{1}{T} \sum_{t=1}^{T} p(k|o_t, \lambda) \tag{2.3}$$

$$\mu_k = \frac{\sum_{t=1}^{T} p(k|o_t,\lambda)o_t}{\sum_{t=1}^{T} p(k|o_t,\lambda)} \tag{2.4}$$

$$\Sigma_k = \frac{\sum_{t=1}^{T} p(k|o_t,\lambda)(o_t-\mu_k)(o_t-\mu_k)'}{\sum_{t=1}^{T} p(k|o_t,\lambda)} \tag{2.5}$$

The posterior probability of the k-th Gaussian component is denoted by Equation 2.6.

$$\begin{aligned} p(k|o_t,\lambda) &= \frac{c_k\mathcal{N}(o_t|\mu_k,\Sigma_k)}{\sum_{k=1}^{K} c_k\mathcal{N}(o_t|\mu_k,\Sigma_k)} \\ \mathcal{N}(o_t|\mu_k,\Sigma_k) &= \frac{1}{(2\pi)^{D/2}|\Sigma_k|^{1/2}} exp[-\frac{1}{2}(o_t-\mu_k)'\Sigma_k^{-1}(o_t-\mu_k)] \end{aligned} \tag{2.6}$$

- **Baum-Welch statistics**: Let's assume an utterance u has L feature frames, where each feature o_t has F dimensions. The UBM trained in the previous stage can now be used for extracting the Baum-Welch statistics.

  The zero-order Baum-Welch statistic for k-th component of the UBM is denoted by $N_k(u)$ and calculated as shown in Equation 2.7. The value of the first-order, $\tilde{F}_k(u)$, and the second-order, $\tilde{S}_k(u)$, Baum-Welch statistics centralised over the UBM mean, μ_k, are computed through Equations 2.8 and 2.9 respectively.

$$N_k(u) = \sum_{t=1}^{L} p(k|o_t,\lambda) \tag{2.7}$$

$$\tilde{F}_k(u) = \sum_{t=1}^{L} p(k|o_t,\lambda)(o_t-\mu_k) \tag{2.8}$$

$$\tilde{S}_k(u) = diag(\sum_{t=1}^{L} p(k|o_t,\lambda)(o_t-\mu_k)(o_t-\mu_k)') \tag{2.9}$$

For an utterance u, the zero-order, $N(u)$, and first-order, $\tilde{F}(u)$, Baum-Welch statistics of a UBM can be summarised by concatenating corresponding values of $N_k(u)$ and

$\tilde{F}_k(u)$ for all K components of the UBM respectively. Similarly, the diagonal elements of a diagonal matrix $\tilde{S}(u)$ consists of $\tilde{S}_k(u)$ blocks where $k = \{1, ..., K\}$ [69].

- **Total variability modeling**: In the i-vector approach, each utterance is described in terms of a speaker and channel dependent GMM mean supervector M. Suppose that K is the number of Gaussian components in the UBM and F is the dimension of the acoustic feature vectors. The speaker and channel independent supervector, m, of dimension $KF \times 1$ is constructed by concatenating means for each Gaussian component of the UBM.

  The aim of the total variability modelling technique is to find a low rank rectangular 'total variability matrix', T, of dimension $KF \times H$ with $H << KF$, and low dimensional 'identity vector', w, of dimension $H \times 1$ such that the probability of the training utterances given the model defined by the supervector M in 2.10 is maximised (Equation 2.10). The dimension of the 'identity vector' (i-vector) is chosen empirically. For simplicity, from now on the 'total variability matrix' is referred as the T-matrix.

  $$M = m + Tw \tag{2.10}$$

  For the utterance dependent mean offset, Tw, the components of the i-vector best describe the coordinates of the speaker in the reduced total variability space. Presenting the utterances in the low-dimensional total variability space, ensures that for representing a new speaker only a small number of parameters need to be estimated. To achieve this the total variability space needs to encapsulate as much as possible of the supervectors in its restricted number of dimensions.

  The value of T-matrix and i-vector are estimated iteratively using the EM algorithm to maximize the likelihood of the training data. In the Expectation step, T is assumed to be known, and we update w. In the Maximization step, w is assumed to be known and we update the T [14, 60, 69, 70].

- **Estimating the total variability matrix**: In the Maximization step, the value of the T-matrix, is updated based on the Baum-Welch statistics and the current values of mean, $E[w(u)]$, and covariance, $E[w(u)w(u)']$, for the posterior distribution of the i-vector $w(u)$. The T-matrix is estimated to capture as much information about the observed data as possible, and as a consequence it includes the information needed to separate out the different classes.

In order to calculate the T-matrix, additional statistics are accumulated across the utterances based on the following Equations.

$$N_k = \sum_u N_k(u) \tag{2.11}$$

$$A_k = \sum_u N_k(u) E[w(u)w(u)'] \tag{2.12}$$

$$\mathbb{C} = \sum_u \tilde{F}(u) E[w(u)'] \tag{2.13}$$

$$N = \sum_u N(u) \tag{2.14}$$

The T-matrix estimate is updated by solving Equation 2.15, where T_i is the i-th row of the T-matrix, and $\mathbb{C}_i$ is the i-th row of $\mathbb{C}$, where i is set to $i = (k-1)F + f$ for each Gaussian component, $k = \{1,...,K\}$, and each observation frame $f = \{1,...,F\}$ [69] (the proof can be found in [71]).

$$T_i A_k = \mathbb{C}_i \qquad i = \{1,...,KF\} \tag{2.15}$$

- **Extracting the i-vectors**: Given the utterance, u, in the Expectation step, the posterior distribution of the i-vector $w(u)$ is updated. This posterior distribution is estimated given the current value of the T-matrix (obtained in the previous step), and the Baum-Welch statistics extracted from the UBM (for utterance u). The value of $w(u)$ is represented in terms of a Gaussian normal distribution (Equation 2.16). Mean and covariance values for the posterior distribution of the i-vector are represented by expectations, $E[w(u)]$, and covariance matrices, $E[w(u)w(u)']$, according to Equations 2.17 and 2.18) respectively (the proof can be found in [71]).

$$w(u) = \mathcal{N}(E(w(u)), E[w(u)w(u)']) \tag{2.16}$$

$$E[w(u)] = \ell^{-1}(u).T'\Sigma^{-1}\tilde{F}(u) \tag{2.17}$$

$$E[w(u)w(u)'] = \ell^{-1}(u) \tag{2.18}$$

$$\ell(u) = I + T'\Sigma^{-1}N(u)T \tag{2.19}$$

The exact value of the i-vector w is equal to the mean of the posterior distribution (Equation 2.17), and the term Σ^{-1} represents the inverse UBM covariance matrix that models the residual variability not captured by the T-matrix [14, 69]. The i-vectors generated in this stage are referred to as the i-vector system's supervector.

- **Multi-class SVM:** For classification, we applied the linear kernel SVM classifier (described in Appendix B.1.1). Given a set of labelled training utterances from C accent groups, a multi-class SVM is trained using the corresponding accent-specific i-vectors. Then, the test speaker's i-vector is scored against each accent specific SVM (using a 'one against all' approach). The accent which gives the maximum score determines the accent of the test utterance.

2.4.3 ACCDIST-SVM based AID

The approach for Accent Characterization by Comparison of Distances in the Inter-segment Similarity Table is called ACCDIST. This text-dependent (supervised) AID approach was first introduced by Huckvale [72]. The ACCDIST measure depends on absolute spectral properties. In this section, first the ACCDIST system introduced by Huckvale [72], and then a more generalised approach suggested by Hanani [56] called ACCDIST-SVM will be described. In this work we use Hanani's ACCDIST-SVM system and report the AID accuracy using this AID system in Section 6.2.4.

Huckvale's ACCDIST based AID

Using the ACCDIST method, the accent of a speaker is characterised using the similarities between realisation of vowels in certain words. It compares a speaker's pronunciation system with average pronunciation systems for known accent groups to recognise his or her accent.

For instance, the realisation of the vowel in the words 'c*a*t', '*a*fter', and 'f*a*ther' is a clue to distinguish between southern and northern British English regional accents. In this example the distance table for the mean cepstral envelopes of the vowel show that for northern English speakers the realisation of vowel in words 'c*a*t' and '*a*fter' is more similar, while for southern English speakers the vowel in '*a*fter', and 'f*a*ther' are more similar.

Huckvale's ACCDIST accent identification approach is carried out in five stages, namely forced-alignment, vowel feature vector generation, vowel distance table measurement, vectorisation and correlation distance measurement [72] as shown in Figure 2.3.

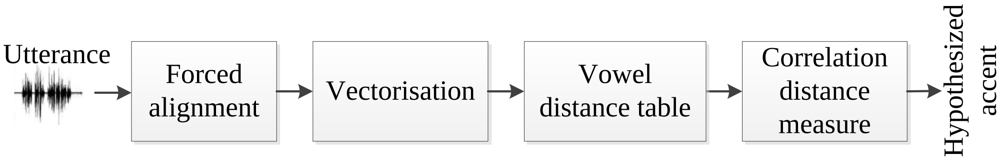

Fig. 2.3 *ACCDIST AID system based on correlation distance measure*

- **Forced-alignment:** In the ACCDIST text-dependent system, during the forced-alignment stage the corresponding phone level transcription and time segmentation is generated for each utterance using a pronunciation dictionary. Next, for each speaker only the vowel segments of the utterances are analysed.

- **Vowel feature vector generation:** Given a phone-level transcription for each utterance, the start and end time index of vowels are determined. Each vowel is divided into two halves by time. For each half the average of 19 MFCC coefficients is calculated. The mean cepstral envelopes in each half are concatenated. Each vowel is then represented by a 40-dimensional vector. If there are multiple instances of a vowel then the 40 dimensional vectors are averaged over all of these instances.

- **Vowel distance measurement:** For each speaker a vowel distance table is generated by computing the distances between the 40 dimensional vectors corresponding to each pair of monophone vowels.

- **Vectorisation:** The distance tables generated in the previous stage are concatenated to form a supervector.

- **Correlation distance measurement:** For each utterance, in order to determine the closest accent group to the test speaker, the correlation between the test speaker's supervector and the accent group mean supervector for each accent is computed. In this process, using distances between pairs of vowels produced by the same speaker, makes the comparison insensitive to various speaker-dependent properties other than the speaker's accent. The correlation distance d between two mean and variance normalized vectors v_1 and v_2 is computed as shown by Equation 2.20. Here '.' represents the dot product between the two vectors, and J is the length of the vectors. The correlation distance for two independent vectors is zero and for two identical vectors is one.

$$d(v_1, v_2) = \sum_{j=1}^{J} v_1.v_2 \tag{2.20}$$

Hanani's ACCDIST-SVM based AID

Huckvale's AID system requires every utterance to correspond to exactly the same known phrase or set of phrases.

Later, Hanani [56] generalised this technique by comparing the realisation of vowels in the triphones rather than the words. In addition, for determining the closest accent group to the test speaker, Hanani's classifier is based on Support Vector Machines (SVMs) with correlation distance kernel. We refer to the Hanani's system as the ACCDIST-SVM system[56].

Given utterances from different speakers, the task is to create and compare the speaker distance tables of the mean cepstral envelope of the most common vowel triphones to identify the speakers accent. The Hanani's ACCDIST accent identification approach is carried out in five stages, namely forced-alignment, vowel feature vector generation, vowel distance measurement, vectorisation and SVM classification as shown in Figure 2.4.

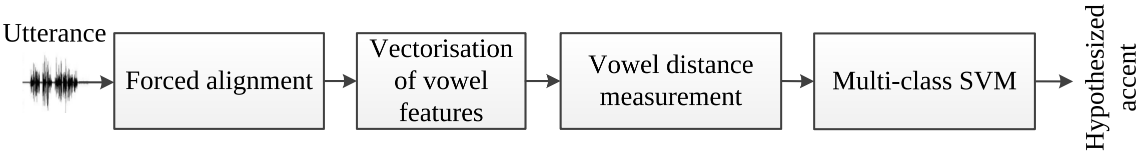

Fig. 2.4 *ACCDIST-SVM AID system*

- **Forced-alignment:** During the forced-alignment stage, a tied-state triphone based phone recognizer is used to generate a triphone level transcription and time segmentation for each utterance. Next, for each speaker only the vowel-triphone segments are analysed.

- **Vowel feature vector generation:** For each utterance, the vowel triphones and their start and end time index are identified from the phone-level transcription. Then, the most common vowel triphones are selected, to ensure that distance tables from different utterances are comparable.

In the same way as Huckvale's method a 40-dimensional vector is constructed. This vector is then concatenated to the vowel duration. For repeated triphones the average of these 41-dimensional vectors is used.

Each realisation of a common vowel-triphone is represented in form of (p_i, v_i) where p_i denotes the i-th vowel triphone, and vector v_i is its corresponding 41-dimensional feature vector.

- **Vowel distance measurement:** A set of cepstral feature vectors were computed in the previous stage for the vowel-triphones. In this stage compute a vowel distance table for each utterance by finding the Euclidean distance between every vowel-triphone pair.
- **Vectorisation:** The distance tables computed in the previous stage are then vectorised and stored in a supervector. The supervector generated in this stage is referred to as the ACCDIST-SVM system's supervector.
- **SVM classification:** In our experiment, ACCDIST-SVM supervectors from C accent groups are used to train a multi-class SVM classifiers with the correlation distance kernel. To address this classification problem, a multi-class SVM with the 'one against all' approach is chosen.

2.5 Proposed approach for visualisation of the AID feature space

Three AID systems were described in this chapter, namely ACCDIST-SVM, phonotactic, and i-vector. The described procedure for AID involved generating a high-dimensional supervector (with accent-specific information) for each utterance, followed by the application of a multi-class SVM accent classification. In this section we propose an approach for visualisation of the AID accent feature space.

Visualisation of the accent feature space provides an insight regarding the relationships between the acoustic realisations of the different accents in the data set and into the performance of multi-accent speech recognition systems.

Each AID method maps an utterance to a relatively high dimensional vector space. This high dimensional feature space is referred to as the 'accent supervector space' or 'accent space', and the high dimensional AID features are referred to as 'accent supervectors'. We are interested in visualising the high dimensional supervector space produced by each AID system. In order to achieve the most insight from the visualisation space, we need to find an approach to gain the maximum separation between the projected classes while reducing the dimensionality of the features space to two. Here, ‘class’ refers to the different regional accents (for example Birmingham accent, Glaswegian accent).

This suggests Linear Discriminant Analysis (LDA) [73], but the rank of the covariance matrix cannot be greater than the number of supervectors that is used to estimate it, which is much less than the dimension of the supervector space. Hence, due to the small sample size N and high dimensionality D of supervector space, it is not possible to invert the within-class covariance matrix. Here, the ‘sample size’ refers to the number of speakers in the data set. The ‘high dimensionality’ refers to the dimension of the AID supervectors.

A solution is to use Principal Components Analysis (PCA) [74] to reduce the dimensionality of the accent supervectors to a new value n, chosen empirically such that $C \leq n \leq N - C$, where C is the number of classes. Then we apply LDA to reduce the dimensionality to two dimensions [75, 76]. In this work we use the EM algorithm to calculate principal components in PCA (EM-PCA) [77]. This enables us to extract the eigenvectors from the high dimensional supervectors with lower computational cost [77].

Full description of the PCA, EM-PCA, and LDA dimension reduction approaches can be found in Appendices A.1.1, A.1.2, and A.1.3 respectively.

For the visualization purpose, after projecting the accent supervectors into a 2-dimensional space, the supervectors that belong to the same ‘accent region’ are represented by a cluster. For each accent region, a v standard-deviation $(0 < v \leq 1)$ contour around the mean value m represents the distribution of supervectors corresponding to speakers with that accent in the supervector space.

We expect to see correlations between the geographical and social similarities and differences between different accent regions and the relative positions of their accent clusters in the AID space. The visualisation for the i-vector, phonotacttic, and ACCDIST-SVM AID feature spaces can be found in Section 6.4.

2.6 Summary

In this chapter we presented four different approaches to address accent issues faced in ASR systems. Then, three popular AID systems were described, namely ACCDIST, phonotactic, and i-vector based systems. The ACCDIST-SVM system is a supervised approach which requires exact transcription of the utterances to recognise the test speaker’s accent. The phonotactic, and i-vector systems are unsupervised and do not rely on pre-transcribed material.

Performance of these three AID systems on British English accented speech is reported in Section 6.2.

The supervectors generated by different AID systems are used for two-dimensional visualisation of the accent space. The visualisation results for different AID accent feature spaces can be found in Section 6.4.

In Chapters 7 and 8 these AID systems will be used for selecting an accented acoustic model that matches the test speaker's accent.

Chapter 3

HMM based speech recognition using GMMs and DNNs

3.1 Introduction

ASR is the task of automatically converting speech data into a written transcription. ASR is not an easy task and visually similar waveforms do not necessarily indicate similar sounds. Therefore, simple pattern recognition is not powerful enough for the speech recognition task [78].

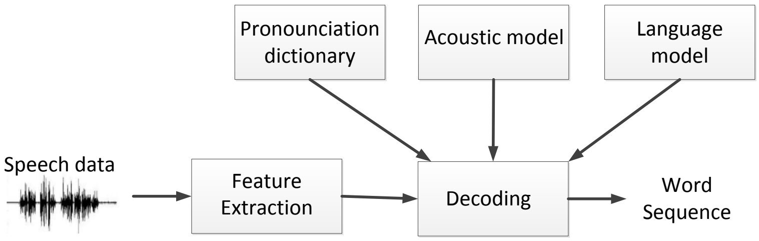

Fig. 3.1 *Diagram of a simple speech recogniser*

The speech recognition task consists of four main stages, namely feature extraction, acoustic modelling, language modelling and decoding [79]. A diagram of a simple speech recognition system is shown in Figure 3.1.

The aim of the feature extraction is to convert the speech waveform into a sequence of acoustic feature vectors which are suitable for applications such as speech recognition and accent recognition. Over the last four decades a low-dimensional representation of speech with capability to preserve the necessary information for ASR systems has been used. An ideal acoustic feature vector should convey the information for making phonetic distinctions, while not being sensitive to speaker specific characteristics such as shape and size of the vocal tract. Many different types of acoustic feature vectors have been proposed in the literature, of which the most popular are Mel Frequency Cepstral Coefficients (MFCCs) [80], Mel Log filterbanks (FBANKs) [80, 81], and Perceptual Linear Prediction coefficients (PLPs) [82]. These low-dimensional features are inspired by physiological evidence of human perception of auditory signals and have been successfully applied to many applications. Recent publications suggested that it is possible to use the raw waveform to train the DNN based acoustic models and the accuracy of these systems matches the result obtained from using the FBANK features [83–85]. The complete description of the acoustic signal processing used in this work can be found in Section 3.3.

In ASR systems, a pronunciation dictionaries is used. Pronunciation dictionaries comprise one or more phone level transcription of the words that occur in the training set. They provide a link between the words and their phone level transcription and can be used in creating the context dependent acoustic modelling (Section 3.4).

The language modelling stage is concerned with the development of structures to model the word sequences using a large quantity of text data to estimate the probability of the word sequences. Details of the language modelling process can be found in Section 3.5.

Acoustic modelling involves a Markov process, to model the underlying sequential structure of a word, plus a mechanism to relate the acoustic feature vectors to the Markov model states which can be achieved with a GMMs or a DNNs. Hidden Markov Models (HMMs) are one of the major approaches for acoustic modelling. They can characterize the observed time-varying speech data samples. In HMMs an observation probability distribution is associated with each state in the HMM, to be able to describe the time-varying speech feature sequences. The complete description of the GMM and DNN based acoustic modelling can be found in Sections 3.7 and 3.10.

The aim of the acoustic decoding stage is to find the most likely word sequence for an observed acoustic data. Sections 3.11 fully describes the speech decoding process.

In sections 3.7 and 3.10, different stages of GMM and DNN based ASR systems are described. The two popular approaches to the GMM-HMM based acoustic model adaptation are introduced. Finally, we present the ASR evaluation formula, to be able to present and compare the recognition performance for different systems.

3.2 The speech recognition problem

During the feature extraction stage the speech waveform is converted into a sequence of acoustic feature vectors $O = \{o_1, ..., o_T\}$. The aim of the speech recognition systems, is to find the most likely sequence of words $w = \{w_1, ..., w_K\}$ that is most likely to have generated O. Since, direct modelling of the $p(w|O)$ is difficult, Bayes rule is applied to transform the problem in Equation 3.1 into an alternative problem in Equation 3.2. Since probability of observation sequence $p(O)$ is not influenced by the word sequence it can be discarded from the Equation 3.2.

$$\hat{w} = argmax_w p(w|O) \tag{3.1}$$

$$\begin{aligned} \hat{w} &= argmax_w p(O|w)p(w)/p(O) \\ &= argmax_w p(O|w)p(w) \end{aligned} \tag{3.2}$$

The probability $p(w)$ is calculated using the language model. The probability $p(O|w)$ of observing feature sequence O given the word sequence w is computed by the acoustic model stage. The acoustic model for any given word sequence w, consists of the corresponding phone level HMMs to make words. The phone level transcription of each word is derived from a pronunciation dictionary.

3.3 Feature extraction

During the feature extraction stage the continuous speech signal is converted into a sequence of acoustic feature vectors. In this work we used MFCC features augmented with delta and acceleration coefficients for the GMM based ASR (in Chapters 7 and 8), and FBANK features for the DNN based ASR systems (in Chapter 6).

3.3.1 Mel Frequency Cepstral Coefficient (MFCC) features

One of the most frequently used acoustic features are MFCCs. The MFCCs are approximately uncorrelated and they provide a good representation of the important speech properties [86]. The main stages of the MFCC feature extraction are illustrated in Figure 3.2.

- **Pre-emphasise:** The high-frequency part of the speech signal are attenuated due to combination of glottal source spectrum with radiation effect of the lips occurred during

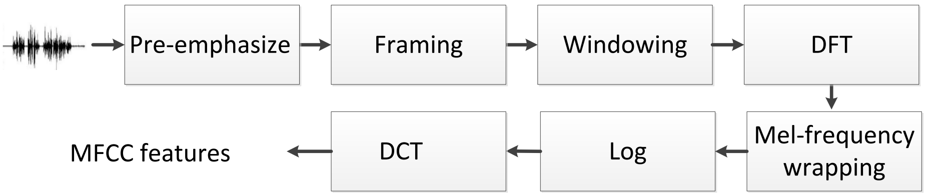

Fig. 3.2 *MFCC feature extraction procedure*

the human sound production process. To compensate the high-frequency of speech and flatten the speech spectrum, a pre-emphasize filter is applied to improve the overall signal-to-noise ratio by increasing the magnitude of the high frequency components. This high pass filter, $HF(z)$, is described in Equation 3.3. In our experiments $\alpha = 0.97$.

$$HF(z) = 1 - \alpha z^{-1} \tag{3.3}$$

- **Framing and windowing:** The pre-emphasized signal is converted into a time series samples that form short segments called frames. Frame lengths are short, to achieve a trade-off between temporal and spectral resolution. The frame duration is usually around 20ms to 30ms, during which the spectrum is relatively constant and the changes in vocal tract shape with time are not significant. The 10ms frame rate captures the changes of the speech signal in time.

  A Hamming window is then applied to each frame to reduce discontinuities at the window edges. Where h represents a sample, and H is the total number of sampling points, Equation 3.4 describes the mathematical expression of the Hamming window $w(h)$.

$$w(h) = 0.54 - 0.46cos(\frac{2\pi h}{H-1}), \quad 0 \leq h \leq H-1 \tag{3.4}$$

- **Discrete Fourier Transform (DFT):** Each frame of H samples in the time domain is transformed into the frequency domain using the Discrete Fourier Transform (DFT) [87]. DFT estimates the short term power spectral density of the speech.

- **Mel-frequency warping:** In this stage a Mel-scale bandpass filterbank is applied to the magnitude of spectrum values to produce filterbank energies. The Mel-frequency scale is a perceptually motivated scale [88] which is designed based on the non-linear sensitivity of the human auditory system with respect to frequencies. The spectrum in frequency domain f is converted into Mel-frequency domain using Equation 3.5, where the Mel-frequency is denoted by $Mel(f)$.

$$Mel(f) = 2595log_{10}(1 + f/700) \tag{3.5}$$

A Mel-scale bandpass filterbank consists of a number of triangular filters which are uniformly spaced on the Mel scale between lower and upper frequency limits [80]. Let $H_m(k)$ be the transfer function of the m-th Mel-scale bandpass filter, and X_k be the power spectrum of each frame. The log spectral energy vector X_m is calculated as shown in Equation 3.6 and contains the energies at the centre frequency of each filter, where M is the number of filters. For example, for a 4 kHz bandwidth approximately 20 filters are used between 0 Hz and 4 kHz.

$$X_m = \sum_{k=1}^{K} ln[|X_k|^2 H_m(k)] \;\; m = 1, ..., M \tag{3.6}$$

- **Discrete Cosine Transform (DCT):** Applying the Cosine transform reduces the correlation between the different components of an acoustic vector. After decorrelating the log-compressed filterbank energies by taking the Discrete Cosine Transform (DCT) of mel-scaled log filterbank energies, MFCCs are generated. In this stage, the DCT is applied to the m-th filter output X_m to produce the i-th MFCC coefficient $c(i)$ as described in Equation 3.7, where M is the total number of filterbank channels.

$$c(i) = \sqrt{\frac{2}{M}} \sum_{m=1}^{M} X_m cos(\frac{\pi i}{M}(m - 0.5)) \tag{3.7}$$

- **Feature normalisation:** Feature normalization such as feature mean and variance normalisation are very important in acoustic modelling specially in neural network training [89]. The cepstral mean value normalisation is applied to reduce the sensitivity to channel effects by simply subtracting the cepstral mean, calculated across the entire speech speech file, from each individual frame. The cepstral variance normalisation is applied to reduce the sensitivity to additive noise by scaling feature coefficients individually to have a unit variance [90, 91]

- **Incorporating delta and acceleration features:** In order to capture the time evolution of the spectral content of the signal, dynamic features have to be added to the MFCC feature vectors obtained in the previous stage. Dynamic features are incorporated in most recognition systems as they improve the performance through capturing the time varying information of the static MFCC feature vectors [92].

Given a set of static feature vectors $c_t = (c_t(1), ..., c_t(N))$, the delta coefficient Δc_t at time t, is obtained from the Equation 3.8. The acceleration coefficients is obtained from the delta coefficients using the Equation 3.9 [93]. For a typical value of $I = 12$, the final feature vector $MFCC(t)$ at time t is defined as shown in Equation 3.10.

$$\Delta c_t(i) = c_{t-B}(i) - c_{t+B}(i) \tag{3.8}$$

$$\Delta^2 c_t(i) = \Delta c_{t-B}(i) - \Delta c_{t+B}(i) \tag{3.9}$$

$$MFCC(t) = (c_t(0), ..., c_t(I), \Delta c_t(0), ..., \Delta c_t(I), \Delta^2 c_t(0), ..., \Delta^2 c_t(I)) \tag{3.10}$$

3.3.2 Filterbank features

During the MFCC feature extraction, the cosine transform remove the correlation between feature components. This is of high importance for GMMs with diagonal covariance matrices. Since modelling the correlated data is not an issue for the DNN systems, there is no need to remove such correlation when DNNs are used. For DNN based modelling, the high-dimensional log Mel filterbank channel outputs (filterbank features) can be used [81].

Better performance was achieved by applying these higher dimensional and richer filterbank features to train the DNN based ASR systems over alternative low-dimensional features (for example MFCCs) [7, 94, 95].

3.4 Pronunciation dictionary

The pronunciation dictionaries are needed to provide the phone level transcription of the words that occur during training and to be able to train the corresponding acoustic models for different phones. During the recognition stage the pronunciation dictionary is used to identify the possible sequence of phones and the words they might form.

3.5 Language modelling

For a large vocabulary speech recognition task the prior probability of a word sequence $w = w_1, ..., w_K$ is approximated using an N-gram word-level Language Model (LM). In an N-gram language model the probability of each word w_i is conditioned on its $N-1$ preceding words, where N is usually in the range of 2 to 4 (Equation 3.11).

$$p(w) \simeq \prod_{k=1}^{K} p(w_k | w_{k-1}, ..., w_{k-(N-1)}) \ \ where \ \ N < k+1 \tag{3.11}$$

The N-gram language models with $N = 1$, $N = 2$ and $N = 3$ are called uni-gram, bigram, and trigram language models respectively. The N-gram probabilities are determined by forming Maximum Likelihood (ML) parameter estimates through counting N-gram occurrences. For three words w_{k-2}, w_{k-1}, w_k, let $C(w_{k-2}, w_{k-1})$ and $C(w_{k-2}, w_{k-1}, w_k)$ denote the number of occurrences of the word pair $\{w_{k-2}, w_{k-1}\}$ and triple $\{w_{k-2}, w_{k-1}, w_k\}$ respectively. The probability of occurrence of w_k given w_{k-1} and w_{k-2} can be estimated by Equation 3.12.

$$\hat{p}(w_k | w_{k-1}, w_{k-2}) = \frac{C(w_{k-2}, w_{k-1}, w_k)}{C(w_{k-2}, w_{k-1})} \tag{3.12}$$

However, many valid word sequences might not have been seen in the training set, and therefore using a simple ML estimation is not accurate. For unseen word sequences, and word sequences that do not occur sufficiently often to obtain an accurate trigram probability estimate using the approach described above, an N-gram probability has to be defined using a different approach. This problem can be moderated by applying discounting and backing-off parameters [96, 97]. The trigram probability can be estimated for any sparse N-gram in terms of a sequence of $N-1$-grams, discount coefficient d and back-off weight α as denoted by Equation 3.13 [2, 79]

$$p(w_n | w_{n-1}, w_{n-2}) = \begin{cases} \frac{C(w_{k-2}, w_{k-1}, w_k)}{C(w_{k-2}, w_{k-1})} & C(w_{k-2}, w_{k-1}, w_k) > C' \\ d\frac{C(w_{k-2}, w_{k-1}, w_k)}{C(w_{k-2}, w_{k-1})} & 0 < C(w_{k-2}, w_{k-1}, w_k) \leq C' \\ \alpha(w_{k-2}, w_{k-1}) p(w_k | w_{k-1}) & otherwise \end{cases} \tag{3.13}$$

Given a count threshold $C^{'}$, when the N-gram counts is bellow this threshold, the discounted ML estimate is applied. The discounting coefficient is determined by $d = (r+1)n_{r+1}/rn_r$, where n_r is the number of N-grams that occur r times in training set [98]. Given a trigram case where the word sequence $\{w_{k-2}, w_{k-1}, w_k\}$ is not observed frequently enough in the training data, then we use a probability based on the occurrence count of a shorter context $\{w_{k-2}, w_{k-1}\}$ which is referred to as 'backing off' and denoted by back-off weight $\alpha(w_{k-2}, w_{k-1})$.

3.6 History of HMM based ASR

In the 1960s, Baum and his colleagues developed the theory of HMMs [99], based on the concept of Markov processes, developed by Andrei Markov [100]. Up to the current time, all modern speech recognition systems use HMMs as they proved to be a natural framework to model the time varying structure of the speech signal [101]. Continuous density HMM-based ASR technology was introduced at the IDA Laboratory [102] and popularised by Bell Laboratory in the early 1980s which was in continuation of the previous work carried out in the 1970s on discrete density HMMs by IBM and Carnegie Mellon University [3].

Speech recognition has been dominated by the GMM–HMMs for almost four decades. In these systems, HMMs model the time-varying speech feature sequences, and the GMMs determine how well each state of each HMM fits the acoustic feature by modelling the state output probability distributions using GMMs [79].

Despite the great success of GMMs one problem is that a large number of components is required to model the actual distribution of acoustic feature vectors associated with a state, and this means a large number of parameters. It is believed that speech has a simple underlying structure, because speech is produced using a small number of parameters of a dynamical system [5].

Maybe other techniques can model the speech features more efficiently by exploiting its simple underlying structure. In fact, neural networks are capable of doing this due their discriminative learning nature to discriminate between different tied tri-phone states. Around two decades ago, neural networks with one non-linear hidden unit managed to successfully predict HMM states from windows of acoustic features [103]. But, it wasn't till year 2010 that Deep learning and the DNNs started making their impact in speech recognition. Before then, the learning algorithms and processing power were not adequate to train a neural networks on large amounts of training data using a network with multiple layers of non-linear hidden units. Current DNN-HMM systems outperform the conventional GMM-HMM systems at acoustic modelling, after using large amount of data, to train neural networks with many hidden layers and non-linear hidden units. In DNN-HMM systems, the sequential property of the speech is modelled by HMMs, and the observation likelihood of the context-dependent states is modelled by DNNs with more than one layer of hidden units between the inputs and outputs [7]. After training the DNNs, each output neuron estimates the posterior probability of HMMs given the input observations [5].

3.7 Acoustic modelling using GMM-HMMs

In most ASR systems, a three state context-dependent HMM is used for the acoustic modelling. The theory behind the GMM-HMM based acoustic modelling is described in this section.

3.7.1 An overview of HMM based ASR

Unlike observable Markov chains that restrict the output at any given state, in HMMs there is no one-to-one correspondence between states and outputs (refer to Appendix C.1 for details on Markov chains). Instead, an observation probability distribution is associated with each state in the HMM, to be able to describe the time-varying speech feature sequences. An HMM produces an observation sequence $O = \{o_1, o_2, ..., o_T\}$, via a hidden state sequence $Q = \{q_1, q_2, ..., q_T\}$ at each given time t. Here each observation vector o_t is of dimension D. It is assumed that the probability of a particular observation at time t does not depend on previous observations and only depends on the current state.

A simple 5-state HMM with its transition probabilities a_{ij} and output probabilities $b_i(o_t)$ is illustrated in Figure 3.3. The entry and the exit states are non-emitting states. The middle three states are emitting states and have output probability distribution associated with them.

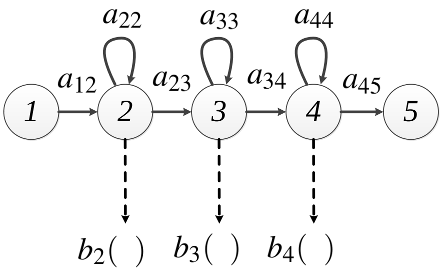

Fig. 3.3 *A left to right, 5-state HMM [2]*

The HMM parameters for a given model λ are namely, an initial state probability vector π, a state transition probability matrix A, and a set of observation Probability Density Functions (PDFs) for each state in the HMM [3, 104–106]:

- **Initial state distribution:** $\pi = \{\pi_i\}$, is an initial state occupation probability vector. As shown in Equation 3.14 it corresponds to the probability that state i is the first state in a set of states $S = s_1, ..., s_N$, where N is the number of states.

$$\pi_i = p(q_0 = s_i) \;\; 1 \leq i \leq N \tag{3.14}$$

- **Transition probability:** $A = \{a_{ij}\}$, is a transition probability matrix, where a_{ij} is the probability of a transition from state i to state j as shown in Equation 3.15 and $i, j = 1, ..., N$.

$$a_{ij} = p(q_t = s_j | q_{t-1} = s_i) \tag{3.15}$$

- **Output probability:** $b_i(o_t)$, is a set of output PDFs, where $b_i(o_t)$ is the observation PDF given HMM state i for a D-dimensional observation vector o_t. A multivariate Gaussian mixture distribution is used to describe $b_i(o_t)$ and as shown in Equation 3.16. Here, the transpose operation is denoted by symbol ' $'$ '.

$$\begin{aligned} b_i(o_t) &= \sum_{m=1}^{M} c_{i,m} \mathcal{N}(o_t | \mu_{i,m}, \Sigma_{i,m}) \\ \mathcal{N}(o_t | \mu_{i,m}, \Sigma_{i,m}) &= \frac{1}{(2\pi)^{D/2} |\Sigma_{i,m}|^{1/2}} exp[-\frac{1}{2}(o_t - \mu_{i,m})' \Sigma_{i,m}^{-1} (o_t - \mu_{i,m})] \end{aligned} \tag{3.16}$$

GMMs are one of the most important probability density functions applied to continuous measurements such as speech feature vectors. Here the output PDFs are expressed as a weighted sum of M Gaussian component densities. For each state i, the weight, mean vector, and covariance matrix for each mixture component, $m = 1, ..., M$, is denoted by $c_{i,m}$, $\mu_{i,m}$, and $\Sigma_{i,m}$ respectively. The mixture weights are positive and satisfy the constraint $\sum_{m=1}^{M} c_{i,m} = 1$. Using the Baum-Welch algorithm which will be described in Section 3.7.3 the parameters $c_{i,m}$, $\mu_{i,m}$, and $\Sigma_{i,m}$ are estimated.

Three problems should be addressed in the context of HMMs before they can be applied to practical applications [107, 108].

- **Likelihood evaluation of an HMM**: Given the model and a sequence of observations, finds the probability that the observed sequence $O = \{o_1, ..., o_T\}$ was produced by model λ and it is denoted by $p(O|\lambda)$. The forward algorithm is used for evaluation of the likelihood of an HMM and it allows us to choose a model which best matches the observations. Full details of the forward-backward algorithm can be found in Section 3.7.2.

- **Estimating the HMM parameters**: Given a model λ, and a set of observations, adjust the model $\hat{\lambda}$ to maximize the probability $p(O|\lambda)$. The Baum-Welch algorithm is applied for learning the HMM models in order to best describe the observed training data and create models that fits real phenomena. The Baum-Welch algorithm is presented in Section 3.7.3.

- **Decoding HMM state sequences**: Given a model λ, and an observation sequence, $O = \{o_1, o_2, ..., o_T\}$, finds the corresponding single best state sequence $Q = q_1, q_2, ..., q_T$ that best explains the observations. The Viterbi algorithm is used for decoding HMM state sequences. The Viterbi algorithm is fully described in Section 3.11.1.

3.7.2 Likelihood evaluation of an HMM (Forward-backward algorithm)

In this section we describe the forward-backward algorithm which computes the $p(O|\lambda)$ [109]. Here, the forward probability, $\alpha_t(i)$, and the backward probability, $\beta_t(i)$ defined by Equations 3.17 and 3.18 respectively. The total likelihood of model can be estimated using the forward algorithm, and the backward algorithm is required for estimating the state occupancies. The α and β values are determined in a recursive manner.

$$\alpha_t(i) = p(o_1, o_2, ..., o_t, q_t = s_i|\lambda) \tag{3.17}$$

$$\beta_t(i) = p(o_{t+1}, o_{t+2}, ..., o_T|q_t = s_i, \lambda) \tag{3.18}$$

The forward algorithm is applied to estimate the probability that an HMM generates an observation sequence O, given the model λ. This is done through evaluating state by state the probability of being at that state given the partial observation sequence. The forward probability, $\alpha_t(i)$, is the probability of observing the partial sequence $O = \{o_1, ..., o_t\}$ and being in state i at time t as shown in Equation 3.17. The forward probability $\alpha_t(i)$ is computed for $1 \leq t \leq T$ and $1 \leq j \leq N$ through the following steps [3].

- **Initialization**: Here, the forward probability is initialised at $t = 1$ for all the states in $1 \leq i \leq N$.

$$\alpha_1(i) = \pi_i b_i(o_1) \tag{3.19}$$

- **Induction**: The induction stage is computed based on the lattice (trellis) structure illustrated in Figure 3.4. It shows the process of reaching to state s_j at time $t+1$ from

N possible states at time t. The computation shown in Equation 3.20 is repeated for all states $1 \leq j \leq N$, and for a given time $1 \leq t \leq T-1$.

$$\alpha_{t+1}(j) = [\sum_{i=1}^{N} \alpha_t(i) a_{ij}] b_j(o_{t+1}) \tag{3.20}$$

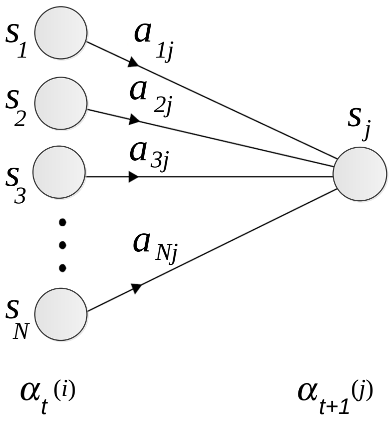

Fig. 3.4 *Illustration of the computation of the forward probability* $\alpha_{t+1}(j)$ *[3]*

- **Termination**: The probability of the observation sequence O given the model parameter λ is equal to the summation of the terminal forward variables $\alpha_T(i)$ for all the possible states.
$$p(O|\lambda) = \sum_{i=1}^{N} \alpha_T(i) \tag{3.21}$$

$$\alpha_T(i) = p(o_1, ..., o_T, q_T = s_i|\lambda) \tag{3.22}$$

Given the model λ and state s_i, the probability of partial observation sequence $O = o_{t+1}, o_{t+2}, ..., o_T$ is found using the backward probability. The backward probability is computed through the following steps [3, 104]:

- **Initialization**: Here, the backward probability is initialised at $t = T$ for all the states in $1 \leq i \leq N$.
$$\beta_T(i) = 1 \tag{3.23}$$

- **Induction**: The backward probability $\beta_t(i)$ at state i and time t, is computed as shown in Equation 3.24. Figure3.5 shows the process of reaching to state s_i at time t from N

possible states at time $t+1$. The computation shown in Equation 3.24 is repeated for all states $1 \leq i \leq N$, and for a given time $T-1 \leq t \leq 1$.

$$\beta_t(i) = [\sum_{j=1}^{N} \beta_{t+1}(j)a_{ij}]b_j(o_{t+1}) \tag{3.24}$$

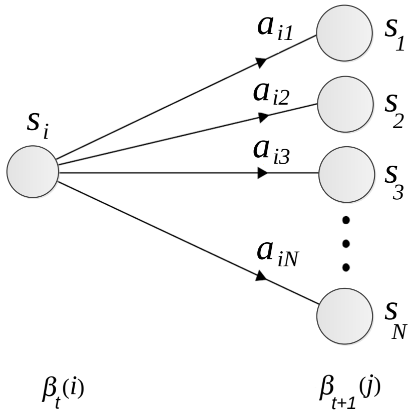

Fig. 3.5 *Illustration of the computation of the backward probability $\beta_t(i)$ [3]*

- **Termination**: The total probability $p(O|\lambda)$ is computed as denoted by Equation 3.25.

$$p(O|\lambda) = \sum_{i=1}^{N} \pi_i \beta_1(i) b_i(o_1) \tag{3.25}$$

The backward recursion part of the forward-backward algorithm is not necessary for evaluating the likelihood of an HMM. The backward computation is a necessary part of the model parameter estimation using the Baum-Welch algorithm which will be introduced in the following section.

3.7.3 Estimating the HMM parameters (Baum-Welch)

Given a model λ and a set of observations, the aim is to adjust the model parameters $\hat{\lambda}$ to maximize the probability $p(O|\lambda)$.

The state sequence is hidden in HMMs, and therefore the HMM parameter estimation problem resembles unsupervised learning using ‘incomplete’ data which can be addressed by the EM algorithm [68]. A generalised implementation of the EM algorithm, known as the Baum-Welch algorithm is applied to the HMM parameter estimation problem [99, 110].

In the general case of EM algorithm, the ‘complete’ data consists of $\varsigma = [O, Q]$, where the hidden state sequence is denoted by $Q = \{q_1, ..., q_T\}$ and the observed data sequence is denoted by state sequence $O = \{o_1, ..., o_T\}$.

The Baum-Welch algorithm estimates the HMM parameters using the EM iterative process that maximizes the probability of the training data given the model, until a local maximum is found.

$$p(O|\hat{\lambda}_{new}) \geq p(O|\lambda_{old}) \tag{3.26}$$

Each iteration of EM consists of the following Expectation and Maximisation steps, which leads to the maximum likelihood estimates of model parameters with respect to objective function $p(O|\lambda)$ [3, 109].

The expressions in the Expectation and Maximisation stages of a GMM-HMM system with multiple component Gaussian PDF are summarised in the following stages.

- **Expectation step**: The posterior probability given the current HMM parameters is computed during the Expectation stage. With the m-th mixture component accounting for o_t, the probability of being in state i at time t is denoted by $\gamma_t(i,m)$ which is shown by Equation 3.27.

$$\gamma_t(i,m) = [\frac{\alpha_t(i)\beta_t(i)}{\sum_{i=1}^{N} \alpha_t(i)\beta_t(i)}][\frac{c_{i,m}\mathcal{N}(o_t|\mu_{i,m}, \Sigma_{i,m})}{\sum_{m=1}^{M} c_{i,m}\mathcal{N}(o_t|\mu_{i,m}, \Sigma_{i,m})}] \tag{3.27}$$

- **Maximisation step**: During the Maximisation stage, the mixture weights, means, and covariances are re-estimated. The ratio between the expected number of times the system is in state i using the m-th Gaussian mixture component and the expected number of times the system is in state i is denoted by a re-estimation term for the mixture coefficient $\hat{c}_{i,m}$. The expected value of the portion of the observation vector o_t allocated to the m-th Gaussian component is denoted by a re-estimation term for the mean vector $\hat{\mu}_{i,m}$. The re-estimation term for the covariance matrix $\hat{\Sigma}_{i,m}$ can be similarly interpreted.

$$\hat{c}_{i,m} = \frac{\sum_{t=1}^{T} \gamma_t(i,m)}{\sum_{t=1}^{T}\sum_{m=1}^{M} \gamma_t(i,m)} \tag{3.28}$$

$$\hat{\mu}_{i,m} = \frac{\sum_{t=1}^{T} \gamma_t(i,m) o_t}{\sum_{t=1}^{T} \gamma_t(i,m)} \tag{3.29}$$

$$\hat{\Sigma}_{i,m} = \frac{\sum_{t=1}^{T} \gamma_t(i,m)(o_t - \hat{\mu}_{i,m})(o_t - \hat{\mu}_{i,m})'}{\sum_{t=1}^{T} \gamma_t(i,m)} \tag{3.30}$$

Next, the value of the output PDF, $b_i(o_t)$, is re-estimated according to Equation 3.16, using the updated values of the mixture weights, means, and covariance matrix [5].

3.8 Modelling context-dependent tied state triphones

Initially, a set of three state single-Gaussian monophone models are created and their means and covariances are trained on the transcription generated based on the pronunciation dictionary.

It is important to ensure that there is enough material to train the HMM state output distribution parameters while capturing important context-dependent information that helps to distinguish between different phone classes [4, 111]. Since the realisation of each phone is greatly influenced by its surrounding phone context, context-dependent triphone models are used as a basis for acoustic modelling.

Context-dependent triphone models contain the information regarding each phone given its left and right adjacent phone context. However, there will be up to N^3 triphones, for a system with N phones and there might not be enough material in the training set for training them. Although most of these triples will be ruled-out by phonological constraints, the standard solution is to apply tree-based state tying to address the data sparsity issue [4, 112].

A decision tree is constructed to group the corresponding states of a set of triphones with same central phone into equivalent classes. In each of these classes, it is assumed that the distributions associated with each of the states are tied. The number of Gaussian mixture components in each state is incremented and the models are re-estimated while there is an improvement in performance.

3.9 Tree based clustering

The tree based clustering starts by mapping a complete set of logical triphones to a reduced set of physical triphones at the state level, during a top-down tree clustering and state tying process. Typically, the choice of which states to tie is made using a binary decision tree. Tree based clustering helps to overcome the data sparsity problem described in the previous section.

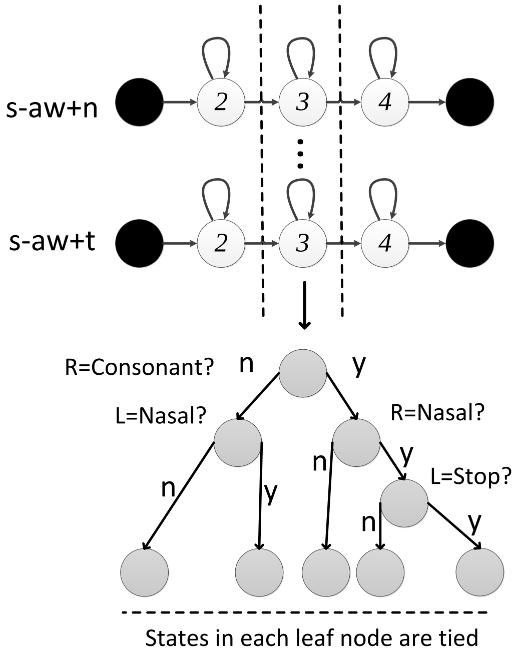

Fig. 3.6 *Decision tree based state tying for center state 'aw' [2, 4]*

A separate tree is built for each phone. The objective is that the terminal nodes of the tree should correspond to sets of triphone contexts that induce the same effect on the realisation of that phone. To achieve this, each non-terminal node of the tree is associated with a question concerning the left or right context in which the phone occurs. These set of predefined questions are attached to the nodes and will result in a phonologically driven binary decision tree.

Decision trees could be applied at the phone level but it is more usual to apply them at the state level. To be able to apply state clustering to all the triphones (logical triphones), one tree is constructed for each state of each phone. HMMs for unseen triphones are then constructed, by using the tied-states corresponding to terminal nodes of a tree based on that triphone's context.

The first step of the tree-building process is to cluster corresponding states of each set of triphones derived from the same monophone, and accumulate all of the HMM states S at the root node of the tree, and assume that they are all tied. Then, the common variance, $\Sigma(S)$, and the common mean, $\mu(S)$, of these tied states are computed for estimating the likelihood of data associated with those accumulated states $L(S)$. The log likelihood, $L(S)$, that a set of training frames F are generated by state S is approximated by Equation 3.31. The posterior probability that each observed frame o_f is produced by the state S is denoted by $\gamma_s(o_f)$.

$$L(S) = \sum_{f \in F} \sum_{s \in S} log(p(o_f|\mu(s), \Sigma(s))\gamma_s(o_f) \tag{3.31}$$

The next task is to find the best phonologically driven question Υ to partition the set of HMM states S into two groups according to the ‘yes’ or ‘no’ answers. For each new group the likelihood of $L(S_{no}(\Upsilon))$ and $L(S_{yes}(\Upsilon))$, and their updated mean and variance are calculated. The increase in the total likelihood of the data corresponds to the accumulated states after splitting the states and is given by Equation 3.32.

$$\Delta L_{\Upsilon} = L(S_{yes}(\Upsilon)) + L(S_{no}(\Upsilon)) - L(S) \tag{3.32}$$

Among all possible questions Υ, the one which maximises the increase in the total likelihood, ΔL_{Υ}, is chosen. At each stage two new nodes are created and this node splitting process is repeated and stops when the increase in the likelihood is below a threshold. To ensure that enough training data is associated with all terminal nodes, we should stop the process when the occupation count is below a threshold. The final decision tree will determine which states should be tied together [4, 112]. An example of decision tree based state tying is illustrated in Figure 3.6. After, this top-down tree clustering is finished and the state tying process is applied, the complete set of triphones is mapped to a reduced number of physical triphones .

3.10 Acoustic modelling using DNN-HMMs

Chapter 4 provides the background needed for the DNN fine-tuning and pre-training procedure. This section uses the background from Chapter 4 to describe the DNN-HMM based acoustic modelling. DNNs are powerful static classifiers and HMMs can successfully model sequential patterns. As a result hybrid DNN-HMM systems are natural candidates for the speech recognition problem. More than two decades ago neural networks were introduced to estimate the HMM state-posterior probabilities given the acoustic observations. However, they were not very popular until a few years ago, when DNN-HMMs were successfully applied to model context-dependent states using large amounts of training data to fine-tune a deep structure with multiple non-linear hidden layers [113–120].

As illustrated in Figure 3.7, HMMs are applied to model the speech dynamics, while each of the DNN's output neurons estimate the posterior probabilities of the HMM's tied-triphone states given the observation, $p(q_t = s|o_t)$, for all possible states s.

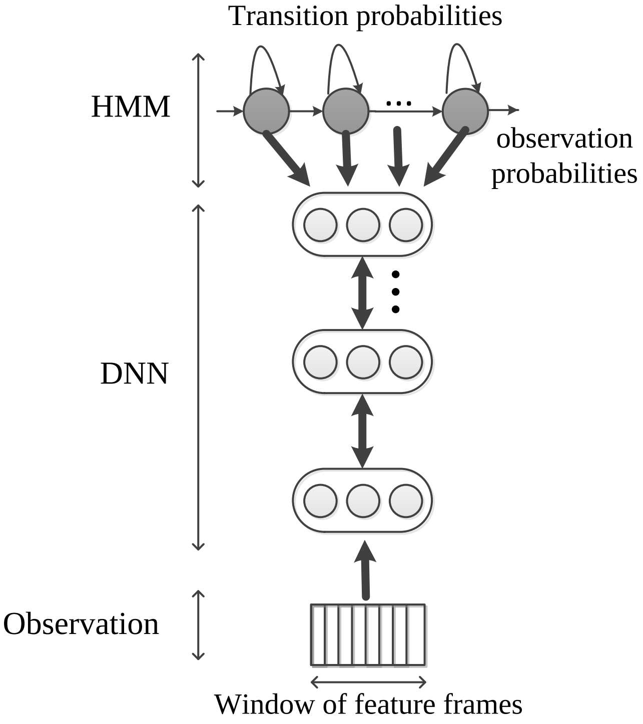

Fig. 3.7 *Structure of a DNN-HMM system [5]*

Recent studies showed that the most important factors that contribute to the GMM-HMM performance are: using a normalised features, using wide window of frames, using enough layers in the DNN, and modelling tied triphone states [5, 7]. Just like the GMM-HMM systems, the Viterbi algorithm can be applied to train the DNN-HMMs. Unlike the GMM based systems were each different state is modelled by a different GMM, a single DNN is trained to estimate all the state probabilities. Also, unlike the GMM systems that accept a single frame as their input, it is important to pass a window of 9 to 13 frames to the DNN systems to exploit their ability to model contextual information [5]. In the following sections we describe training and decoding stages of automatic speech recognition using a DNN-HMM system.

3.10.1 Fine-tuning of a pre-trained DNN-HMM system

For the DNN-HMM training, initially a GMM-HMM system is trained, whichd provides the DNN system with HMM transition probabilities, triphone state alignments, and a decision tree for triphone state tying.

During the DNN fine-tuning all the network parameters are updated using the back-propagation and Stochastic Gradient Descent (SGD) algorithms [35]. Full description of DNN fine-tuning can be found in Section 4.4. The hidden layers of the DNN are initialized during the unsupervised pre-training through layer-wise training of RBMs using the Contrastive Divergence (CD) algorithm [121, 122]. During the generative pre-training of the DNN, the network learns to model the structure of the input data. A full description of the DNN pre-training can be found in Section 4.5. After the generative pre-training, the whole network has to be trained discriminatively. Hence, a randomly initialised softmax layer is added on top of the Deep Belief Network (DBN) (stack of RBMs) with initialised weights to create a DNN with initialised parameters [123].

The Viterbi algorithm is used in the GMM-HMM systems to provide a state level forced alignment of the training data. In order to be able to user these state-frame alignments for training the DNNs, each tied state triphone (senone), is mapped to a senone ID. The value of the senone's mean, determines the senone ID. Using the mapping between features and each senone ID, the feature to senone ID pairs are generated to train the DNN (fine-tune). Also, the senone's prior probability is computed, by dividing the number of frames associated with each senone by the total number of frames.

Using the frame-state alignment adopted from the GMM-HMM system, the DNN system is fine-tuned and the entire network parameters are updated and the prior probability of all the tied-state triphones are computed. A mini-batch SGD is used to minimize per-frame Cross-Entropy (CE) between the training labels and the network output $J_{NLL}(W,b;O,Q)$. Full description training the DNN weight, W, and bias, b, parameters using error back propagation with CE can be found in Section 4.4.2. For each utterance with T frames, Equation 3.33 shows the CE criterion. It has been shown on average, after applying the new model (W',b'), as the frame-level CE reduces (Equation 3.35), the likelihood score of observation sequence O given the word sequence w improves (Equation 3.34). This argument might not be true for every single utterance but on average it is true [5].

$$J_{NLL}(W,b;O,Q) = -\sum_{t=1}^{T} logp(q_t|o_t;W,b) \tag{3.33}$$

$$J_{NLL}(W',b';O,Q) < J_{NLL}(W,b;O,Q) \tag{3.34}$$

$$logp(O|w;W',b') = log\pi(q_0) + \sum_{t=1}^{T} log(a_{q_{t-1}q_t}) + \sum_{t=1}^{T}[logp(q_t|o_t;W',b') - logp(q_t)] \quad (3.35)$$

During the DNN fine-tuning the transition probabilities are re-estimated. The DNN-HMM aims to maximize probability of observation features being in a correct state.

The network is re-trained and each time a new state-frame alignment is generated from the fine-tuned DNN and the DNN-HMM system with updated transition probabilities. The network is re-trained up to a point where there is no improvement in development set recognition accuracy.

3.11 Decoding

The recognition task aims to find the most likely sequence of words. The speech utterances can be modelled using a multiple-level network. In this multiple-level network, each triphone is represented by a network of states at the first level. Then at the next level, each word is represented by a network of triphone states. At the final level, each sentence is represented by a network of word states which are connected together with connection weights corresponding to the language-model probabilities.

The goal of the decoding task is to find the sequence of words corresponding to the most likely path through this multi-level network. This can be achieved by applying a single-pass Viterbi decoding approach. Given the strong constraints provided by the language model, it is feasible to apply the Viterbi algorithm to this multiple-level network. This algorithm, evaluates the probabilities for all valid partial paths from the lowest to the top level. In this section the Viterbi algorithm is described with full details.

Usually a network is represented by a structure, such that the models can be shared among the different hypothesis with common sub-word models. For large vocabulary speech recognition it is important to reduce the number of the hypothesis and remove the paths with low scores. For each time frame, all paths whose likelihood score is not within a certain threshold of the best scoring path are pruned out. Usually, likelihood of all except a small number of states will tend to be small, and unlikely possibilities will be pruned out and the search will be narrowed down [92]. Advances in Weighted Finite State Transducers (WFSTs) [124] showed that it is possible to compose the HMM models, pronunciation dictionary, and language model in a well optimised network. This method can be used for practical applications as it offers an efficient and flexible representation of the HMM topology, lexicon, and language model (refer to [125] for more details).

3.11.1 Decoding HMM state sequences (Viterbi algorithm)

Given an observation sequence $O = \{o_1, ..., o_T\}$, the Viterbi algorithm finds the corresponding single most likely sequence of hidden states, $Q = \{q_1, ..., q_T\}$ (Equation 3.2). In other words, Viterbi algorithm approximates the probability $p(O|\lambda)$, by finding the most likely state sequence Q that maximizes $p(Q, O|\lambda)$. The maximum probability over all partial state sequences ending in state i at time t is denoted by $\delta_t(i)$ and given by Equation 3.36. For each j and t, the array $\psi_t(j)$ keeps track of the previous state with the highest probability. The best state sequence can be found after the initialization stage, recursively using the following stages [3, 104].

$$\delta_t(i) = max_{q_1, q_2, ..., q_{t-1}} p(q_1, q_2, ..., q_t; q_t = s_i; o_1, o_2, ..., o_t|\lambda) \tag{3.36}$$

- **Initialization**: Here, an initial value is chosen for $\delta_t(i)$ and $\psi_1(i)$ at $t = 1$ for state i.

$$\begin{aligned} &\delta_1(i) = \pi_i b_i(o_1), \;\; 1 \leq i \leq N \\ &\psi_1(i) = 0 \end{aligned} \tag{3.37}$$

- **Recursion**: For the current state j at time t, an array $\psi_t(j)$ keeps track of the most likely previous state with highest probability.

$$\begin{aligned} &\delta_t(j) = max_{1 \leq i \leq N}[\delta_{t-1}(i)a_{ij}]b_j(o_t), \;\; 1 \leq j \leq N, 2 \leq t \leq T \\ &\psi_t(j) = argmax_{1 \leq i \leq N}[\delta_{t-1}(i)a_{ij}], \;\; 1 \leq j \leq N, 2 \leq t \leq T \end{aligned} \tag{3.38}$$

- **Termination**: At the end of the observation sequence $O = o_1, ..., o_T$, the best score p^* and the probability of state q_T^* is computed as shown by Equation 3.39.

$$\begin{aligned} &p^* = max_{1 \leq i \leq N}[\delta_T(i)] \\ &q_T^* = argmax_{1 \leq i \leq N}[\delta_T(i)] \end{aligned} \tag{3.39}$$

- **Path backtracking**: At the end of the observation sequence, after backtracking through the most likely predecessor states q_t^*, the optimal HMM state sequence, Q^*, is returned.

$$\begin{aligned} &q_t^* = \psi_{t+1}[q_{t+1}^*], \;\; t = T-1, T-2, ..., 1 \\ &The \;\; best \;\; state \;\; sequence: \;\; Q^* = q_1^*, q_2^*, ..., q_T^* \end{aligned} \tag{3.40}$$

The computation of the Viterbi algorithm is implemented using a lattice structure.The main difference between the forward algorithm and the Viterbi algorithm is that instead of adding up the probabilities from different state sequences coming to the same final state as shown by Equation 3.20, a maximum operation is applied as shown by Equation 3.38 [3, 104].

Up to this point we described the Viterbi algorithm for decoding, which can be applied to the GMM based systems directly. However, for decoding an HMM, the posterior probability, $p(q_t = s|o_t)$, produced by a DNN should be converted to the likelihood, $p(o_t|q_t)$, as shown by Equation 3.41. For each state, q_t, and given the observation o_t, the posterior probability $p(q_t|o_t)$ can be estimated from the DNN as shown by Equation 4.4.

$$p(o_t|q_t = s) = p(q_t = s|o_t)p(o_t)/p(s) \tag{3.41}$$

Term $p(s)$ indicates the state prior probability which is estimated from the training data by dividing the number of frames belong to state s, by the total number of frames. Term $p(o_t)$ can be ignored as its not dependent to the word sequence [5, 126].

3.12 Acoustic model adaptation in GMM based ASR

Any mismatch between training and test data degrades the recognition accuracy. Considerable performance degradations occurs when speaker-independent systems are tested on a speaker that is not well-represented in the training set. Acoustic model adaptation is one solution to this problem. The purpose of adaptation algorithms is to use relatively small amounts of data from each new speaker to modify the parameters of a well-trained HMM without discarding the benefits of training on the full training set and reach the performance level of a speaker dependent system. As a result of acoustic model adaptation, model parameters are updated and become more representative of the adaptation data.

The definition of ‘relatively small’ depends on the task requirements. For instance, in applications related to automated phone systems one can only expect less than a minute of speech. On the other hand, users of a dictation system train their system for at least half an hour, since this system is planned to be used by that individual for a long time. Model-based algorithms can achieve significant reduction in the error rate through acoustic model adaptation, perhaps incrementally as more data is available from a test speaker.

For continuous density HMMs, two of the most popular adaptation techniques are described in this section, namely, MLLR adaptation, and MAP adaptation. Other popular adaptation techniques such as Constrained MLLR (CMLLR) [127, 128], Speaker Adaptive Training (SAT) [129], and eigenvoice speaker clustering approach [130] are briefly described in Appendix D. Each method has its strengths and weaknesses; and based on the application an appropriate method or a combination of different methods (hybrid approach) can be chosen.

3.12.1 Maximum Likelihood Linear Regression (MLLR)

For continuous density GMM-HMM acoustic models, the MLLR adaptation approach uses the adaptation data from a new speaker and updates the speaker independent model parameters to maximize the likelihood of the adaptation data. Since the major differences across different speakers are characterised by the Gaussian means the other HMM parameters are not adapted.

The speaker independent output Gaussian density mean parameters can be adapted using a set of linear regression transformation functions which are applied to the Gaussian components that are grouped by a regression class tree. In this section we describe the regression class tree structure and how it applies the mean transformation to the Gaussian components in each regression class. A single transformation matrix is associated with each regression class which transforms all the mixture components that belong to that class.

Let μ_{im} denote the m-th mean vector for each state i in a Gaussian output density function. The goal of MLLR is to map this speaker independent mean vector μ_{im}, to a new space as given in Equations 3.42 such that the new model can better represent the new speaker's characteristics. Here, A_c is a regression matrix and b_c is a bias vector corresponding to some regression class c.

$$\hat{\mu}_{im} = A_c \mu_{im} + b_c \tag{3.42}$$

Equation 3.42 can also be written as shown in Equation 3.43 where W_c is an the extended transform matrix (Equation 3.44) and ξ_{im} is the extended mean vector (Equation 3.45). Here, the transpose operation is denoted by symbol ' $'$ '.

$$\hat{\mu}_{im} = W_c \xi_{im} \tag{3.43}$$

$$W_c = [b_c, A_c] \tag{3.44}$$

$$\xi_{im} = [1, \mu_{im}^{'}]^{'} \tag{3.45}$$

The extended transform matrix W_c can be tied over multiple number of classes determined by the regression class tree, so that the same transformation can be shared among multiple distributions with similar acoustic characteristics.

The aim is to estimate the transformation matrix such that the likelihood of the adaptation data being generated by each state in the HMM system with updated parameters is maximised. The adaptation data $O_R = \{o_1,...,o_R\}$ is used for training the transformation matrix W_c.

Given the adaptation data O_R with D-dimensional feature vectors, the Gaussian PDF for the adapted system is computed by replacing the speaker independent mean vector in Equation 3.16 by the new speaker adapted mean vector [68, 128, 130, 131].

In MLLR adaptation, a regression tree can be used to group together mixture components such that the same transformation is applied to them. The mixture component groupings are stored in regression classes (terminal nodes) of the binary regression tree.

The regression tree is built using the speaker independent model set. There are two main approaches for defining regression classes; one is a data driven approach based on clustering of mixture components and the other is a knowledge driven approach based on broad phonetic classes (e.g. vowels, silence, nasals).

In the mixture clustering approach, the input data is partitioned based on a centroid splitting algorithm, using a Euclidean distance measure to grow the binary regression class tree. Then, a likelihood measure is applied for comparing the mixture components and mixture components in an HMM set that are similar are grouped together in the same regression class. The number of regression classes (terminal nodes of the tree) are determined empirically and they specify the final component groupings.

In the phonologically driven regression tree clustering approach, a set of pre-defined phonologically driven classes are defined to group the HMM components. As a result, all HMM mixture components that belong to the same broad phonetic class correspond to the same regression class.

Initially, the adaptation feature vectors are aligned to the corresponding Gaussian components. Then, for each of the regression classes the occupation counts are accumulated. The regression class tree is traversed, and the mean transformation is applied to the Gaussian components in each regression class which has sufficient adaptation data. The regression nodes that have insufficient data are pooled back to their parent nodes with sufficient data and the transformation of the parent node will apply to them. Figure 3.8 illustrates a binary regression tree with four regression classes $R = \{r_4, r_5, r_6, r_7\}$. The occupation counts are accumulated for each of the regression classes. When the data for construction of a transformation matrix is insufficient a dotted arrow and node are used. As shown in Figure 3.8, nodes $\{r_5\}$, $\{r_6\}$, and $\{r_7\}$ have insufficient data and they have to pooled back to nodes $\{r_2\}$, $\{r_3\}$, and $\{r_3\}$ respectively, where there is sufficient adaptation data. Consequently, the transforms W_2 W_3 W_4 are generated for nodes $\{r_2\}$, $\{r_3\}$, and $\{r_4\}$ receptively. As a result when the transformed model set is required the transformation matrices W_2, W_3 and W_4 are applied to regression classes $\{r_5\}$, $\{r_6, r_7\}$ and $\{r_4\}$ respectively.

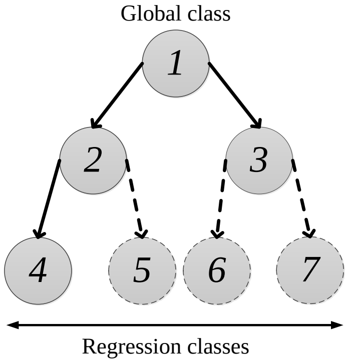

Fig. 3.8 *A binary regression tree with four regression classes [2]*

Studies show that, as more adaptation data becomes available, a better performance is achieved through applying transformation to the group of Gaussian components that belong to the same ‘dynamic’ class determined by the mixture clustering approach, rather than the same ‘static’ class determined by the broad phone class approach. For a small amount of adaptation data there is not much difference between the two approaches. Using MLLR adaptation, it is possible to adapt distributions for which there are no observations, as a result of tying transformations across a number of mixture components. Also, in some cases when there is very limited adaptation data available, the global adaptation is applied which is the same as a tree with just a root node [128, 131–134].

3.12.2 Maximum A Posterior (MAP)

Maximum A Posterior (MAP) adaptation, also known as Bayesian adaptation, uses a small amount of adaptation data to modify the model parameters, while exploiting the prior knowledge. As a result, the prior probability density does not allow large parameter deviation, unless the parameter values estimated from the adaptation data have high certainty [135].

Given the prior model distribution $p(\lambda)$, the MAP estimate of a HMM parameter λ, requires finding values of λ that maximizes the posterior distribution $p(\lambda|O)$. Here, the observation data of length R is denoted by O. This can be expressed in terms of likelihood $p(O|\lambda)$ using the Bayes’ theorem given by Equation 3.46.

$$\hat{\lambda} = argmax_{\lambda} p(\lambda|O) = argmax_{\lambda} p(O|\lambda)p(\lambda) \tag{3.46}$$

When MAP is applied to speaker adaptation, the prior distribution is normally the speaker independent model. When there is no prior information available, $p(\lambda)$ has a uniform distribution, and Equation 3.46 will represent the formula for the Maximum Likelihood (ML) estimation and aims to choose a value for λ that maximizes the likelihood of the adaptation data $p(O|\lambda)$. However, the ML approach only works well when there is sufficient data available for estimating the model parameters and it is unreliable when the data are sparse. As a result, the prior knowledge is incorporated into the MAP adaptation formula to prevent this issue. The EM algorithm can be used as the ML to estimate the HMM parameter.

Where $\mu_{i,m}$ represents the prior mean, and τ is a balancing factor between prior mean and the ML mean estimate (determined empirically), the MAP update formula for state i and mixture component m and observation sequence of length R is defined in Equation 3.47. When the amount of adaptation data is large, then the MAP's mean estimate, $\hat{\mu}_{i,m}$, will move towards the mean of the adaptation data, $\bar{\mu}_{i,m}$, and when it is small, the mean MAP will be closer to the speaker independent component mean, $\mu_{i,m}$ [132, 135]. Here, $N_{i,m}$ is the probability of occupying the mth Gaussian mixture component of the state i at time t for the adaptation data with R observations where $1 \leq t \leq T_r$ and $1 \leq r \leq R$ (Equation 3.48).

$$\hat{\mu}_{i,m} = \frac{N_{i,m}}{\tau + N_{i,m}} \bar{\mu}_{i,m} + \frac{\tau}{\tau + N_{i,m}} \mu_{i,m} \tag{3.47}$$

$$N_{i,m} = \sum_{r=1}^{R} \sum_{t=1}^{T_r} \gamma_t^r(i,m) \tag{3.48}$$

A comparison between the Gaussian mean formula (Equation 3.29) and the MAP mean update formula (Equation 3.47) shows that the MAP adaptation effectively interpolates the parameters obtained from the adaptation data with the parameters from the original prior. As the amount of adaptation data increases the MAP estimate relies less on this prior distribution and converges towards to the ML estimate.

MAP needs more data to outperform MLLR, because MLLR adaptation is defined based on the pooled Gaussian transformation, and MAP adaptation is defined at the component level and needs to update each component separately. During the MAP adaptation, only the parameters that have been seen during the adaptation will be updated [130].

3.13 Evaluating ASR performance

Once the test data has been decoded into a sequence of words, the recognition performance can be evaluated using the percentage Word Error Rate (%WER) given by Equation 3.49 or by percentage word accuracy given by Equation 3.50, where, N is the total number of words in the test utterance. The occurrences of three types of errors determine the ASR performance, namely number of times that a word is omitted, an extra word is recognised, and a wrong word is recognised which correspond to deletions (D), insertions (I), and substitutions (S) respectively.

$$WER = \frac{D+S+I}{N} \times 100 \tag{3.49}$$

$$Accuracy = 100 - WER \tag{3.50}$$

3.14 Summary

A conventional ASR system consists of the following units, namely acoustic signal processing that converts the audio signal from time to frequency domain while removing the noise, acoustic modelling which generates the acoustic modelling score from a combination of acoustic and phonetic knowledge. Language modelling that generates the language model score by approximating the probability of a word sequence, and the decoding component which generates a sequence of words with highest score as a result of combining the acoustic and language model scores.

In this chapter different units of a conventional ASR system and two of the most popular acoustic model adaptation techniques for the GMM based ASR systems, namely MAP and MLLR were presented.

Chapter 4

Deep neural networks background

4.1 Introduction

ASR has been dominated by GMM based systems for almost four decades. DNNs have recently become one of the important modelling algorithms in the modern speech recognition systems [7]. Over the last decade, DNN based systems have achieved high accuracy due to their discriminative nature compared to GMM based systems in predicting the context-dependent HMM states [7, 34, 35, 136].

We start this chapter by introducing simple forms of neural networks, namely single-layer and multi-layer perceptrons. Then, we present DNNs, their training criteria, and their parameter estimation algorithm (back-propagation).

Next, we describe Deep Belief Networks (DBNs) and explain how they can be used for pretraining a DNN through initialising its model parameters.

This chapter provides the background for the DNN-HMM acoustic modelling described in Section 3.10, where the hybrid DNN-HMM systems, their fine-tuning and pretraining are introduced. Please note that the majority of notations in this chapter are based on Li Deng's book [5].

4.2 Single-Layer Perceptron (SLP)

An SLP is the simplest form of a neural network. A psychologist Frank Rosenblatt gave the name 'perceptrons' to a group of artificial neural network models which were proposed in late 1950s. A single-layer perceptron consists of an input layer that receives the input data, and an output layer which send the output to the user. There are a set of weighted connections between the input layer and the output layer as shown in Figure 4.1. For each node the weighted sum of the inputs is computed, and then compared with a sigmoid type of threshold function. A sigmoid function $\sigma(.)$ is defined in Equation 4.1. The sigmoid function is one of the most popular activation functions.

$$\sigma(z) = \frac{1}{1+e^{-z}} \tag{4.1}$$

Based on the result of the comparison the system will perform a specific action. In a single-layer perceptron, for a set of n inputs, the weights define a decision hyperplane in an $n-1$ dimensional space that can only divide the data into two classes. When a classification problem can be solved by using a single hyperplane the input data is called linearly separable. Therefore, the application of single-layer perceptrons is limited to classifying linearly separable data.

In late 1969s, Minskey and Papert [137] published a book which strongly criticized the perceptron models and listed major limitations of single-layer perceptrons. They claimed that SLPs can only learn linearly separable patterns and they are not capable of learning an XOR function. Their claims caused researchers to move on from perceptron type machine learning algorithms for pattern recognition.

It was not until 1986, that Rumelhart [138], popularized the use of multi-layer perceptrons for pattern classification, by introducing a convergent error back-propagation algorithm for adaptively updating the weights using the gradient.

Two years later, Hornik [139] showed that its possible to use Multi-Layer Perceptrons (MLPs) to approximate any function, when a sufficient number of hidden units is available between the input and the output units. This work has once again brought back the attention to neural network type algorithms by showing the strength of multi-layer neural networks [104]. In the next section, multi-layer perceptrons are described.

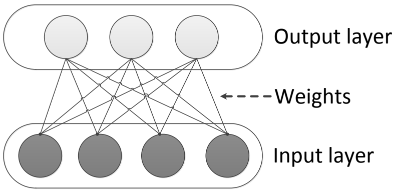

Fig. 4.1 *A single-layer perceptron*

4.3 Multi-Layer Perceptron (MLP)

Unlike SLPs, in MLPs it is possible to address the non-linearly separable problems [140]. An MLP consists of an input layer containing the input features, at least one hidden layer (middle layer), and an output layer. The value of each node in hidden layers is equal to the weighted sum of the input units after passing through a sigmoid threshold function.

During the MLP training process, the input data is propagated in a forward direction one layer at a time, and the outputs of the units in one layer becomes the input to the units in the next layer. The hidden layers enable the network to model the non-linear patterns by bringing complexity to the system's architecture.

Figure 4.2, shows an MLP with one hidden layer. The aim of the MLP training is to adjust the weights to minimize the error between the predicted and the desired outputs of the system using the back-propagation algorithm [104]. In the next section this algorithm is described.

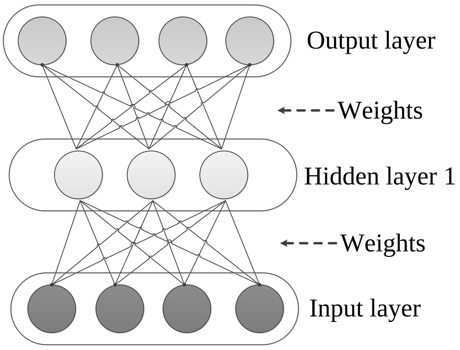

Fig. 4.2 *A multi-layer perceptron with one hidden layer*

4.4 DNN training (fine-tuning)

A Multi-Layer Perceptron (MLP) with many hidden layers (at least two) is called a Deep Neural Network (DNN). DNNs can model data with complex non-linear structure. Backward propagation of errors (back-propagation) is the standard procedure for training (fine-tuning) the neural networks. It is based on the Stochastic Gradient Descent (SGD) to minimize the network error.

4.4.1 Forward propagation

Lets assume we have a training set with M samples $=\{(o^m, y^m)|0 \leq m < M\}$, where y^m is the corresponding desired output vector for the m-th observation vector o^m.

In a network, with $L+1$ layers, the data propagation is done from the input visible layer shown by $\ell = 0$ to the last layer (output layer) denoted by $\ell = L$.

In the forward propagation stage, the output of each transformation is the input of the next transformation. Just like in an MLP, the value of each node in hidden layers is equal to the weighted sum of the outputs of the units in the previous layer. Each layer ℓ, consists of N_ℓ units called neurons.

The excitation vector is the vector of inputs to a layer, and the activation vector is the vector of outputs. The excitation vector of the ℓ-th layer, z^ℓ, is calculated by addition of the bias vector, b^ℓ, to the result of the multiplication of the weight matrix, W^ℓ, and the previous layer's activation vector, $v^{\ell-1}$ (Equation 4.2).

$$z^\ell = W^\ell v^{\ell-1} + b^\ell \quad , \quad for \;\; 0 < \ell \leq L \tag{4.2}$$

The activation vector of the ℓ-th layer, v^ℓ, can be calculated by applying a sigmoid activation function $\sigma(.)$ (Equation 4.3) to the excitation vector z^ℓ.

$$v^\ell = \sigma(z^\ell) = \frac{1}{1+e^{-z^\ell}} \quad , \quad for \;\; 0 < \ell < L \tag{4.3}$$

The value of each neuron, i, in the output layer, v_i^L, represents the probability that the observation vector o belongs to the i-th class, $p_{dnn}(i|o)$, where class $i \in \{1, ..., C\}$. Here, the probability, $p_{dnn}(i|o)$, is found by normalizing the excitation vector with a softmax function $softmax(.)$ as shown in Equation 4.4. Here, the i-th element of the excitation vector z^L is denoted by z_i^L.

$$v_i^L = p_{dnn}(i|o) = softmax_i(z^L) = \frac{e^{z_i^L}}{\sum_{j=1}^{C} e^{z_j^L}} \tag{4.4}$$

For an observation vector o, during the DNN forward computation process, the output of the network v^L is determined using Equation 4.4 in terms of the activation vectors v^ℓ for layers $\ell = \{1, .., L-1\}$ and the model parameters $\{W, b\} = \{W^\ell, b^\ell | 0 < \ell \leq L\}$.

4.4.2 Error back-propagation

In order to train the DNN model parameters, a training criteria has to be defined. Lets assume that the term $J(W,b;o,y)$ denotes the loss function given the observation vector o with its corresponding output vector y, and the model parameters $\{W,b\}$.

A popular empirical criteria for classification task, is the Cross-Entropy (CE) represented by J_{CE}. For the observation o in the training set, the empirical probability that the observation o belongs to class i is denoted by $y_i = p_{emp}(i|o)$ and it is often denoted using a hard class label c as shown in Equation 4.6. The probability estimated by the DNN that the observation o belongs to class i, is denoted by $v_i^L = p_{dnn}(i|o)$. Consequently, the CE can be interpreted as the negative logarithm of probability that the observation o belongs to class c as $-\log v_c^L$. As shown by Equation 4.5, the CE criterion can be written in the form of a Negative Log Likelihood loss function J_{NLL}.

$$\begin{aligned} J_{CE}(W,b;o,y) &= -\sum_{i=1}^{C} y_i log v_i^L \\ &= -\sum_{i=1}^{C} p_{emp}(i|o) log p_{dnn}(i|o) \\ &= -log v_c^L \\ &= J_{NLL}(W,b;o,y) \end{aligned} \tag{4.5}$$

$$y_i = I(c=i) = \begin{cases} 1, & if \;\; c=i \\ 0, & otherwise \end{cases} \tag{4.6}$$

To be able to do error back-propagation, the model parameters should be updated using the mini-batch Stochastic Gradient Descent (SGD) algorithm which estimates the gradient with respect to model parameters based on a small batch (block of N input vectors) of randomly selected training samples [5, 7, 136, 141].

The update formulas for mini-batch SGD are denoted by Equation 4.7. Here, the batch size and the learning rate are denoted by M_b and ε respectively. At iteration t, the average weight matrix gradient is denoted by ΔW_t^ℓ and the average bias vector gradient is represented by Δb_t^ℓ estimated from the training batch with M_b samples as shown in Equation 4.8.

$$\begin{aligned} W_{t+1}^\ell &= W_t^\ell - \varepsilon \Delta W_t^\ell \\ b_{t+1}^\ell &= b_t^\ell - \varepsilon \Delta b_t^\ell \end{aligned} \tag{4.7}$$

$$
\begin{aligned}
\Delta W_t^\ell &= \frac{1}{M_b} \sum_{m=1}^{M_b} \nabla_{W_t^\ell} J_{CE}(W,b;o^m,y^m) \\
\Delta b_t^\ell &= \frac{1}{M_b} \sum_{m=1}^{M_b} \nabla_{b_t^\ell} J_{CE}(W,b;o^m,y^m)
\end{aligned}
\tag{4.8}
$$

The mini-batch SGD algorithm is a compromise between the SGD and batch training algorithms. Not only it is easy to apply in a parallel manner within the mini-batch (like mini-batch training algorithm), but also it is capable of getting out of local optima due to its noisy estimation of gradient (like SGD algorithm), and consequently, compared to SGD it needs less time to reach convergence.

To be able to start the back-propagation process, the gradient of the training criterion J_{CE} with respect to the networks output v^L should be computed. Then the CE criteria is coupled with softmax output layer L. The gradient of training criterion with respect to DNN parameters leads to estimation of the global error which is denoted by Equation 4.9.

$$
\begin{aligned}
\nabla_{W_t^L} J_{CE}(W,b;o,y) &= \nabla_{z_t^L} J_{CE}(W,b;o,y) \frac{\partial z_t^L}{\partial W_t^L} = (v_t^L - y)(v_t^{L-1})^T \\
\nabla_{b_t^L} J_{CE}(W,b;o,y) &= \nabla_{z_t^L} J_{CE}(W,b;o,y) \frac{\partial z_t^L}{\partial b_t^L} = (v_t^L - y)
\end{aligned}
\tag{4.9}
$$

Now, the error back-propagation can be performed for hidden layers $0 < \ell < L$. We start from the hidden layer preceding the Softmax, and evaluate the gradient of the training criterion J_{CE} with respect to the DNN's model parameters and moving towards the first layer (Equation 4.10). The term $\sigma^{'}(.)$ denotes the first derivative of the sigmoid activation function and it is computed as shown by Equation 4.11. The symbol '.' indicates the dot product in the following equations.

$$
\begin{aligned}
\nabla_{W_t^\ell} J_{CE}(W,b;o,y) &= \nabla_{v_t^\ell} J_{CE}(W,b;o,y) \partial v_t^\ell / \partial W_t^L = [\sigma^{'}(z_t^\ell).e_t^\ell](v_t^{\ell-1})^T \\
\nabla_{b_t^\ell} J_{CE}(W,b;o,y) &= \nabla_{v_t^\ell} J_{CE}(W,b;o,y) \partial v_t^\ell / \partial b_t^L = \sigma^{'}(z_t^\ell).e_t^\ell
\end{aligned}
\tag{4.10}
$$

$$
\sigma^{'}(z_t^\ell) = (1 - v_t^\ell).v_t^\ell \tag{4.11}
$$

Here, the error signal at layer ℓ is denoted by e_t^ℓ and it can be back-propagated from the output layer $\ell = L$ and proceed towards the first layer $\ell = 1$ (Equation 4.12).

$$e_t^{\ell-1} = \nabla_{v_t^{\ell-1}} J(W,b;o,y) = \begin{cases} (W_t^\ell)^T e_t^\ell, \;\; if \;\; \ell = L \\ (W_t^\ell)^T [\sigma^{'}(z_t^\ell).e_t^\ell], \;\; if \;\; \ell < L \end{cases} \tag{4.12}$$

Finally, the SGD update formulas (Equation 4.7) are used. During each training 'epoch' (pass over whole training set), for each batch of the input training data M_b, the SGD parameter update is performed.

The back-propagation algorithm explained in this section can be summarised through the following stages [5, 104].

- **Initialisation**: In this stage, the weight matrix, W_0^ℓ, and the bias vector, b_0^ℓ, for each layer are initialised. It is important to initialise these parameters with small random weights and biases, to prevent symmetry, otherwise it is possible to cycle through the same values for weights without reaching a convergence.

- **Forward propagation**: In the forward pass the input propagates from the input layer to the output layer and the activation vector, v_ℓ, is calculated for each layer as shown in Equation 4.3 for layers $0 < \ell < L$, and by Equation 4.4 for layer $\ell = L$.

- **Back-propagation**: In the backward pass the estimated error shown in Equation 4.10 back-propagates up through the network in a layer-wise manner over all the training data to adjust the weight and bias parameters according to the SGD update formula given by Equation 4.7.

- **Iteration**: The weight and bias parameters are adjusted after each iteration over all the training set (according to Equation 4.7). Here, each iteration of parameter update is called one 'epoch' of training. This iterative parameter update is continued until the cost function reaches a certain point where it stops changing or a certain number of epochs are passed.

4.4.3 Practical considerations

There are a number of practical considerations that we have to take into account during the DNN fine-tuning process. In this section, three of the most important ones are pointed out [5].

- **Data preprocessing**: In DNN training, it is very important to apply normalization techniques, such as cepstral mean and variance normalisations, to reduce channel distortions and be able to use the same learning rate across all weight dimensions.

- **Learning rate and batch size**: The DNN's convergence time and the resulting model is influenced considerably by the combination of the batch size and the learning rate. The batch size, M_b, and the learning rate, ε, shouldn't be too large to guarantee the convergence. Choosing a large learning rate causes fluctuation around the optimum weight values and fails to converge. On the other hand, choosing a very small learning rate results in a very slow convergence. The batch size and learning rate values are determined empirically.

- **Network structure**: Since each layer of a DNN acts as a feature extractor for the previous layer, the number of neurons per layer should be large enough to capture the important patterns. Having too many neurons per layer will cause over fitting to the training data and having not enough neurons will cause under fitting. Stacking too many hidden layers will also cause over fitting, since each layer bring an additional constraint, and having very few number of layers will cause under fitting.

  The size of the network is determined empirically according to the size of the training data. Generally, it is easier to find a good configuration in a wide and deep model rather than a narrow and shallow network. One possible reason is that the former model contains a large number of local optima that can perform equally well.

4.5 DNN pre-training using Contrastive Divergence (CD)

Unsupervised DNN model parameter initialisation (pre-training) facilitates the subsequent discriminative fine-tuning of DNNs. It is empirically confirmed that a pre-trained DNN system achieves a higher performance compared to a DNN that uses random initialization. Since the DNN is highly non-linear, when there is not enough training material available to fine-tune the network, the initialisation may considerably affect the final model. Therefore, unsupervised pre-training can prevent the system from falling into a local minima during the fine-tuning process, by introducing a useful prior to the system [142–145]. In this section the DNN pre-training algorithm is described [5, 6].

4.5.1 Restricted Boltzmann Machine (RBM)

Restricted Boltzmann Machines (RBMs) are a type of a neural network, proposed three decades ago by Geoffrey Hinton [146]. The DNN model parameters can be initialised using the RBM algorithm.

An RBM consists of two layers; one hidden layer which contains binary stochastic hidden units (represents the input features) and one visible layer which contains stochastic visible units. The hidden units are usually modelled with a Bernoulli distribution $h \in \{0,1\}$. The visible units either take on the binary values $v \in \{0,1\}$ and are modelled with a Bernoulli distribution, or take on real values and are modelled with a Gaussian distribution. As shown in the Figure 4.3, there are a set of weights W associated with the connections between the visible and hidden units, and there is no visible-visible or hidden-hidden connections between the neurons in the same layer. An RBM assigns a joint probability $p(v,h)$ over the visible units v and hidden units h, defined in terms of an energy function $E(v,h)$ as shown in Equation 4.13. Here, Z is a normalization factor computed as shown in Equation 4.14. The energy function for a Gaussian-Bernoulli RBM is computed, given the visible vector v with Gaussian distribution, hidden vector h with Bernoulli distribution, the weight matrix W, the hidden layer's bias vector b and the visible layer's bias vector c, according to Equation 4.15.

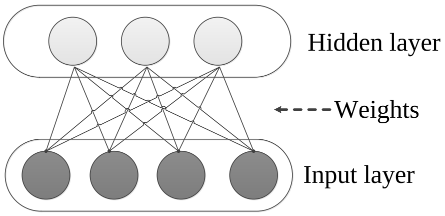

Fig. 4.3 *Illustration of the RBM structure*

$$p(v,h) = \frac{e^{-E(v,h)}}{Z} \tag{4.13}$$

$$Z = \sum_{v,h} e^{-E(v,h)} \tag{4.14}$$

$$E(v,h) = -b^T v - c^T h - v^T W h \tag{4.15}$$

Given the constraint that there are no intra-layer connections, and knowing that the hidden layer has a Bernoulli distribution, $h_j \in \{0,1\}$ and the visible layer has a Gaussian distribution, the probability for some input data, given the hidden layer is denoted by Equation 4.16. Here,

I is the identity matrix. The activation probability for a single neuron, $p(v_i|h)$, is summarised in Equation 4.17

$$p(v|h) = N(v; b + W^T h, I) \tag{4.16}$$

$$p(v_i = 1|h) = \sigma(b + W^T h) \tag{4.17}$$

Similarly, the probability of the hidden layer, given the visible layer is defined as in Equation 4.18 yielding Equation 4.19.

$$p(h|v) = \prod_i p(h_j|v) \tag{4.18}$$

$$p(h_j = 1|v) = \sigma(c + Wv) \tag{4.19}$$

Using Equation 4.18 allows us to initialize a neural network with forward propagation through the RBM hidden units. As shown in Equation 4.20, the Negative Log Likelihood (NLL) criterion should be minimized for training an RBM. Here, the update parameters are listed in Equation 4.21, where ε is the learning rate. From the training batch with M_b samples, at iteration t the average weight matrix gradient, ΔW_t, and the average bias vector gradient for the visible layer, Δb_t , and the hidden layer, Δc_t, are represented by the Equation 4.22 [5].

$$J_{NLL}(W, b, c; v) = -logp(v) \tag{4.20}$$

$$\begin{aligned} W_{t+1} &= W_t - \frac{\varepsilon}{M_b} \Delta W_t \\ b_{t+1} &= b_t - \frac{\varepsilon}{M_b} \Delta b_t \\ c_{t+1} &= c_t - \frac{\varepsilon}{M_b} \Delta c_t \end{aligned} \tag{4.21}$$

$$\begin{aligned} \Delta W_t &= \rho \Delta W_{t-1} + (1-\rho) \frac{1}{M_b} \sum_{m=1}^{M_b} \nabla_{W_t} J_{NLL}(W, b, c; v^m) \\ \Delta b_t &= \rho \Delta b_{t-1} + (1-\rho) \frac{1}{M_b} \sum_{m=1}^{M_b} \nabla_{b_t} J_{NLL}(W, b, c; v^m) \\ \Delta c_t &= \rho \Delta c_{t-1} + (1-\rho) \frac{1}{M_b} \sum_{m=1}^{M_b} \nabla_{c_t} J_{NLL}(W, b, c; v^m) \end{aligned} \tag{4.22}$$

To update the RBM parameters, it is important to maximize $logp(v)$ with respect to model parameters $\{W, b, c\}$. The update rule of the RBM weights, requires taking the gradient of $logp(v)$ as denoted by Equation 4.23. Here, the expectation, $\langle v_i h_j \rangle_{data}$, is computed from the training data based on the frequency that visible unit v_i, and the hidden unit h_j, fire together, and $\langle v_i h_j \rangle_{model}$ is the same expectation computed from the model distribution. For an RBM, unlike the DNN, computing the gradient of the log likelihood of the data respect to model parameters is not computationally feasible.

$$\nabla w_{ij} J_{NLL}(W, b, c; v) = -[\langle v_i h_j \rangle_{data} - \langle v_i h_j \rangle_{model}] \tag{4.23}$$

Contrastive Divergence (CD) is an efficient approach to approximate the RBM's gradient, which was initially developed by Hinton and fully described in [121, 122]. In CD the gradient approximation with respect to the visible-hidden weights is carried out as show in Equation 4.24 [5].

$$\begin{aligned} \nabla w_{ij} J_{NLL}(W, b, c; v) &= -[\langle v_i h_j \rangle_{data} - \langle v_i h_j \rangle_{\infty}] \\ &\approx -[\langle v_i h_j \rangle_{data} - \langle v_i h_j \rangle_{1}] \end{aligned} \tag{4.24}$$

When direct sampling of observations is not feasible, the Gibbs sampling algorithm, approximates an observation sequence from a specified multivariate probability distribution. Here, the model expectation, $\langle v_i h_j \rangle_{model}$, in Equation 4.23, is replaced by applying the Gibbs sampler for one step, $\langle v_i h_j \rangle_1$, or for more steps till infinity, $\langle v_i h_j \rangle_\infty$.

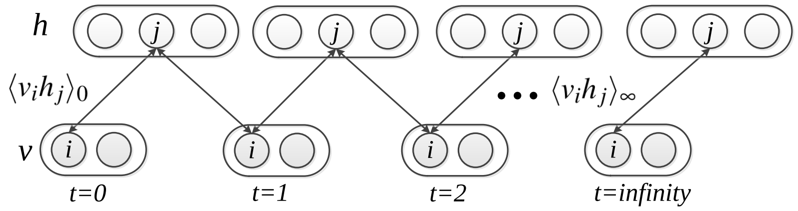

Fig. 4.4 *Illustration of the contrastive divergence approach for RBM learning [5, 6]*

The sampling process for the Contrastive Divergence (CD) approach is represented by Figure 4.4. Then, based on the posterior probability of the visible sample, $p(v|h)$, a hidden sample is generated as denoted by Equation 4.19. After that, a visible sample is produced for the Gaussian-Bernoulli RBMs based on the posterior probability $p(v|h)$, using the hidden sample generated in the previous stage as denoted by Equation 4.16. This process can be repeated for many steps. However, the Gibbs sampler is often run for one step to estimate

$\langle v_i h_j \rangle_1$ because running the Gibbs sampler for infinite number of steps is not feasible. As a result the model expectation $\langle v_i h_j \rangle_{model}$ can be approximated by Equation 4.25. As shown in Equation 4.26, in CD, the sampling approach is applied to generate a hidden vector ($\sim$ means sample from the model), while the expected value of the posterior distribution, $E(v|h)$, is used to generate a visible vector [5].

$$\langle v_i h_j \rangle_{model} \approx \langle v_i h_j \rangle_1 = v_i^1 h_j^1 \tag{4.25}$$

Here v_0 is a sample from the training set, h^0 is a sample from the probability $p(h|v^0)$. The mean-field algorithm, can be applied to generate, v^1 which is a sample from the probability $p(v|h^0)$, and h^1 which is a sample from the probability $p(h|v^1)$.

$$\begin{aligned} h^0 &\sim p(h|v^0) \\ v^1 &= E(v|h^0) = p(v|h^0) \\ h^1 &= E(h|v^1) = p(h|v^1) \end{aligned} \tag{4.26}$$

Similar steps can be applied to find the $\langle v_i h_j \rangle_{data}$, as shown in Equation 4.27.

$$\langle v_i h_j \rangle_{data} \approx \langle v_i h_j \rangle_0 = v_i^0 p_j(h|v^0) \tag{4.27}$$

The update rules for the model parameters $\{W, b, c\}$ are computed as shown in Equation 4.21 based on the gradient estimation denoted by Equation 4.28. In a Gaussian-Bernoulli RBM, the expected value of the posterior distribution $E(v|h)$, is calculated using Equation 4.16 [5].

$$\begin{aligned} \nabla_w J_{NLL}(W, b, c; v) &= -[\langle hv^T \rangle_{data} - \langle hv^T \rangle_{model}] \\ \nabla_b J_{NLL}(W, b, c; v) &= -[\langle v \rangle_{data} - \langle v \rangle_{model}] \\ \nabla_c J_{NLL}(W, b, c; v) &= -[\langle h \rangle_{data} - \langle h \rangle_{model}] \end{aligned} \tag{4.28}$$

4.5.2 Deep Belief Network (DBN)

Pre-training may help in optimization and results in generalization error reduction [147–149]. The Deep Belief Network (DBN) consists of a stack of RBMs, such that the hidden layer of each RBM acts as the visible layer for the next RBM. Thus, the number of the DBN's hidden layers is equal to the number of trained RBMs [35]. After training an RBM on the input data, a weight matrix W is generated. For each data vector v a new vector of expected hidden neuron activations h is produced, which is then used as the next RBM's input data. During the unsupervised layer-wise training of the DBN, the weights from one RBM are applied to extract the features from the output of the previous RBM. This process is repeated until the top two layers of the DBN are reached.

Since the conditional probability, $p(h|v)$, of the RBMs and the DNNs with a sigmoid activation function have the same form, the weights from all the hidden layers of the DBN can be used as the DNN's initial weights [123].

The only difference between the DNN and the DBN is that the former has labels. As a result, after unsupervised pre-training a softmax output layer (randomly initialised) is added on top of the DBN. Then during the DNN fine-tuning, the back-propagation algorithm is applied to fine-tune all the weights in the network discriminatively [35]. Over-fitting is a serious problem in DNNs [150]. Hence, during the fine-tuning, to prevent neural networks from over-fitting to the training set one popular approach is to apply early stopping using validation data [150].

4.6 Summary

In this chapter we described the fine-tuning and pre-training procedure for a simple DNN system with a softmax output layer. Figure 4.5 illustrates the process that leads to fine-tuning of a pre-trained DNN system.

The generative pre-training of the DNNs is carried out through training of a stack of RBMs (DBN) using the CD algorithm. Part (a) of this Figure shows that first the visible layer v is initialised with observation data x. Then, a RBM is trained using this visible layer. The output layer of the first RBM h^1 becomes the input layer to the second RBM with output layer h^2. Part (b), shows how a stack of RBMs form a DBN. The network parameters learnt during the pre-training stage, will be used to initialize the DNN parameters, and it is hoped this would allow the rapid progress of the discriminative fine-tuning, and reduce possible over fitting. In part (c), a softmax layer, which contains all the training targets, is added on top of the stack of the DBN to form a pretrained DNN. During the network fine-tuning the network parameters are updated in a layer-wise manner by applying the back-propagation and SGD algorithms.

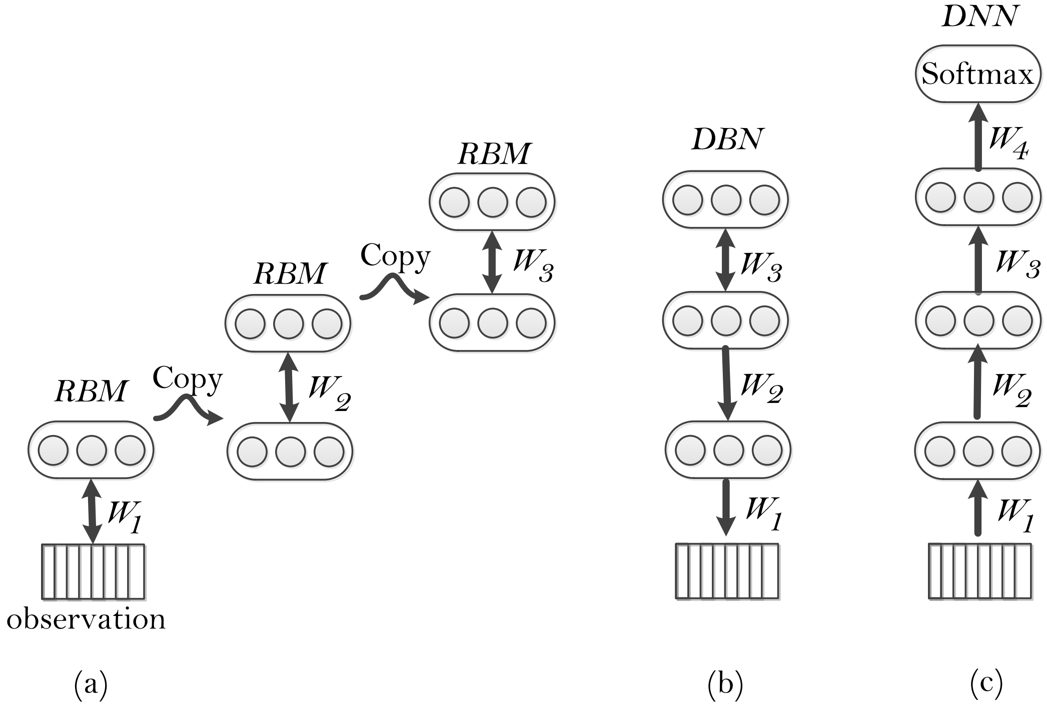

Fig. 4.5 *A layer-wise pre-training using a stack of RBMs [7]*

Chapter 5

Speech corpora

5.1 Introduction

This thesis reports the results of ASR and AID experiments on British English accented speech. Two corpora were used in the majority of these experiments, namely the 'Accents of the British Isles' corpus (ABI) and the British English version of the Wall Street Journal corpus (WSJCAM0). These corpora are described in this section.

5.1.1 The Accents of the British Isles (ABI) corpus

The Accents of the British Isles (ABI) corpus [12] is a British English corpus, recorded by the speech group of the University of Birmingham, Its purpose is to investigate the effect of different regional accents on speech recognition performance. The utterances were recorded at a sample rate 22,050 samples per second and 16 bit resolution in a range of environments in libraries and community centres using both far-field and close-talking microphones.

The first and second set of the ABI database, namely ABI-1 and ABI-2, are collections of speech recordings from 285 and 262 speakers from 14 and 13 different regions of the British Isles, respectively. In each region the objective was to record speech from 10 male and 10 female subjects who were born in that region and have lived there for all of their lives, and, if possible, whose parents had also lived in the region for all of their lives. The age distribution of the speakers is between 16 and 79.

The ABI corpus covers a range of typical application words, sentences and phrases. The content is designed with a number of uses in mind and aimed to capture accent-specific phenomena. The content of the recordings falls into two main categories. The first is the application data category, which provides examples for a range of applications over a wide range of accents to improve ASR performance on such applications. The second is the phonetically motivated category, which provides data for training accent-specific acoustic models in order to improve ASR performance in general. The details of the recording details are the same in ABI-1 and ABI-2, except that in ABI-2 an additional set containing 22 SCRIBE sentences was recorded from each speaker. Each speaker reads through 20 prompt texts, within the following categories.

- **Application data:**
  - Game commands: 60 short phrases containing game commands with typically one or two words (e.g: 'change view', 'grab image', 'toggle source', 'select left'). The ABI file names under the game commands category are, namely 'gca', 'gcb', and 'gcc'.
  - Catalogue codes: Short phrases containing 10 sequences of 4 alphabetic characters (letters), and the international radio operator's alphabet (e.g:'G P Y O', 'golf', 'papa', 'yankee', 'oscar'). The ABI file name under the catalogue codes category is 'cc'.
  - PIN numbers: Short phrases containing 10 4-digit sequences (e.g: 'four zero nine one'). The ABI file name under the PIN numbers category is 'pin'.
  - Equipment control: long phrases containing equipment specific commands (e.g: 'climate control seventy one degrees', 'navigation select route home'). The ABI file names under the equipment control category are, namely 'eca', and 'ecb'.

- **Phonetically motivated data:**
  - Careful words: Short phrases containing consonant-vowel-consonant syllables used to highlight vowel sounds (e.g: 'hoed' (to rhyme with showed), 'hoid' (to rhyme with 'void'), 'howd' (to rhyme with 'loud'). There is a set of 19 of these words, repeated 5 times in random order. The ABI file names under the careful words category are, namely 'cwa', 'cwb', 'cwc', 'cwd', and 'cwe'.
  - Short passages: Long phrases containing a short 'accent diagnostic' story ('When a sailor in a small craft...') in form of 20 phonetically balanced sentences. The ABI file name under the short sentences category is shortsentence.
  - Short sentences: Long phrases with 3 paragraphs containing the sentences from SCRIBE 1 (e.g: 'Gary attacked the project with extra determination', 'I itemise all accounts in my agency'). The ABI file names under the short passages category are, namely 'shortpassagea', 'shortpassageb', and 'shortpassagec'.

- Short phrases: A set of short phrases (e.g: ‘it‘s so sweet’, ‘while we were away’, ‘thin as a wafer’, ‘has a watch’, ‘roll of wire’). It consists of 18 phonetically rich phrases with three or four words in each. The ABI file name under the short phrases category is ‘shortphrase’.

In the work reported in this thesis, from both corpora we only used the data from the close-talking microphones and a subset of utterances, namely the ‘shortpassagea’, ‘shortpassageb’, and ‘shortpassagec’ (SPA, SPB and SPC), the ‘shortsentences’ and the ‘shortphrases’ with average duration of 43.2, 48.1, 53.4, 85 and 34.5 seconds respectively. This is summarised in Table 9.1.

Table 5.1 *Portion of files used from the ABI-1 corpus in this research*

Code	File name	Average length (seconds) (including silence)	Application AID	Application ASR
SPA	shortpassagea	43.2s	test	test
SPB	shortpassageb	48.1s	train	adapt
SPC	shortpassagec	53.4s	train	adapt
shortphrase	shortphrase	85s	train	adapt
shortsentence	shortsentence	34.5s	train	adapt

Set one of the Accents of the British Isles (ABI-1) corpus contains data from 13 different ‘accent regions’ and standard (southern) British English (sse). The sse speakers were selected by a phonetician. A complete list of ABI-1 accent regions can be found in Table 5.2. The ABI-1 accent regions fall into four broad ‘accent groups’, namely Scottish (SC: shl, gla), Irish (IR: uls, roi), Northern English (NO: lan, ncl, lvp, brm, nwa, eyk) and Southern English (SO: sse, crn, ean, ilo). Please note that in our experiments we included the nwa (North Wales) in the northern English set.

Table 5.2 *Accents and their corresponding accent codes in the ABI-1 corpus*

Code	Location	Code	Location
sse	Standard Southern English	uls	Ulster
crn	Cornwall	lan	Lancashire
ean	East Anglia	ncl	Newcastle
ilo	Inner London	lvp	Liverpool
shl	Scottish Highlands	brm	Birmingham
gla	Glasgow	nwa	North Wales
roi	Republic of Ireland	eyk	East Yorkshire

The second set of ABI corpus (ABI-2) consists of speech from 13 British accent regions that are not covered in ABI-1 plus some additional sse speakers. A complete list of ABI-2 accent regions can be found in Table 5.3. The ABI-2 accent regions fall into four broad 'accent groups', namely Scottish (SC: edn), Northern English (NO: les, ddl, htp, lds, srb, sot), Southern English (SO: brs, sos, smr, hfd), and Wales (WA: crf, crd).

Table 5.3 *Accents and their corresponding accent codes represented in the ABI-2 corpus*

Code	Location	Code	Location
sse	Standard Southern English	htp	Hartlepool
brs	Bristol	hfd	Hereford
crf	Caernarfon	lds	Leeds
crd	Cardiff	srb	Shrewsbury
les	Leicestershire	sos	Southend-on-Sea
ddl	Dudley	sot	Stoke-on-Trent
edn	Edinburgh	smr	Somerset

Using ABI-1 for AID and ASR

In both accent recognition and speech recognition experiments, a development set is required to train the system parameters. Examples of such system parameters are, namely LLR fusing coefficients for AID, and grammar scale factor, word insertion penalty, and number of regression classes for ASR. However, in the ABI corpus, there is no development set for training these system parameters.

For AID experiments, we apply a 3-fold cross validation for setting the training, test, and development sets. During this procedure two subsets are used for training and development set and the remaining subset for testing. We divided the ABI-1 speakers into three folds. One fold contains 94 and the other two folds contain 95 speakers. Accent and gender are distributed uniformly across all three folds. In each round, the distribution of speakers in each fold is such that the training, development, and test sets had no speaker in common. This process was repeated three times to cover all the ABI-1 speakers in the test phase. The SPA recordings were chosen for testing. Finally in the recognition stage, the scores from each round are combined together and the final performance is reported.

Unless it is noted otherwise, for all the ASR experiments the WSJCAM0 development set is used, and we divided the ABI-1 data in to 2-folds, each time half of the data is used for training and the other half is used for testing. This process is repeated two times to cover all the ABI-1 speakers in the test phase. Finally in the recognition stage, the scores from each round are combined together and the final performance is reported. In each experiment the SPA recordings were use for testing, and the SPB, SPC, shortphrase, and shortsentence recordings were used for adaptation (refer to Table 9.1). The SPA recordings were chosen for testing because we wanted our AID experiments be comparable to our ASR results and also the AID results reported by Hanani [56], and Demarco [17, 63].

5.1.2 WSJCAM0 corpus

The WSJCAM0 corpus [13] is a British English speech corpus suitable for large vocabulary speech recognition purpose. It was gathered by Cambridge University and contains the material from a subset of prompting texts from the Wall Street Journal. The utterances were recorded at a sample rate of 16,000 samples per second and 16 bit resolution in a quiet room recording environment using a far-field desk microphone and a head-mounted close-talking microphone. The minimum age of the speakers is 18 and majority of them are between 18 and 28 [13, 151].

The WSJCAM0 corpus is a British English equivalent of the WSJ0, which is a US American English corpus. The training set was selected randomly in paragraph units taken from the WSJ0 training set, and it comprises 90 utterances from each of 92 speakers. The development set consists of 18 speakers each reading 90 utterances from WSJ0 training material. The test set utterances are also taken from the WSJ0 test set, and it contains data from 48 speakers each reading 40 sentences including only 5,000 word vocabulary limit (referred as SI-dt-o5), and another 40 sentences including only 64,000 word vocabulary limit. The detailed transcriptions of all the utterances are available.

For our test experiments we use the 5k vocabulary test set (SI-dt-o5). Our training data (WSJT) comprises the WSJCAM0 training set which includes in total 15.50 hours of speech (recorded using the close talking microphone). Our development set (WSJD) is taken from the WSJCAM0 development set and has duration of 2.25 hours.

In all of our experiments the following codes were used when referring to training and development (Table 5.4).

Table 5.4 *Code names used for WSJCAM0 corpus*

Code	Data section	Length
WSJT	WSJCAM0 training set	15.50 hours
WSJD	WSJCAM0 development set	2.25 hours
SI-dt-o5	WSJCAM0 5k vocabulary test set	1.2 hours

5.2 Summary

In this chapter two British accent speech corpora were described, namely ABI and WSJCAM0 corpus. The details of the ABI-1 and ABI-2 accent distribution are also presented in this chapter.

Chapter 6

Accent recognition experiments and visualisation of accent space

6.1 Introduction

In real applications, a computationally efficient AID system can be added to an ASR system to select the relevant accent-specific acoustic model or pronunciation dictionary based on the user's accent and mitigate the problems caused by regional accents for ASR.

This chapter presents the performance of our unsupervised AID systems and compares them against DeMarco's i-vector [17, 63], Hanani's phonotactic [56], and Hanani's acoustic-phonotactic fused [56] systems. The supervised ACCDIST-SVM AID system used in this work (developed by Hanani et al. [56]) is also compared with the ACCDIST system proposed by Huckvale [72].

As mentioned earlier in Section 5.1.1, all these systems are tested on 'shortpassagea' (SPA) utterances and trained on namely the , 'shortpassageb', and 'shortpassagec' (SPB and SPC), the 'shortsentences' and the 'shortphrases' with average duration of 43.2, 48.1, 53.4, 85 and 34.5 seconds respectively from ABI-1 unless mentioned otherwise. A 3-fold cross validation procedure was done for setting the training, test, and development sets (Section 5.1.1).

In this research, accent recognition systems are developed for two purposes, namely (1) visualisation and understanding of accent space, (2) accent-specific acoustic model selection for ASR systems.

6.2 Accent identification system description

In this section we describe three different AID systems.

6.2.1 Phonotactic system

For the phonotactic approach, an SVM-based PPRLM is designed as shown in Figure 6.1 and described in Section 2.4.1.

Our phonotactic system applies 14 accent-specific phone recognisers and one general English recogniser trained on the WSJCAM0 to the SPA utterances from ABI-1. Each accent-specific phone recogniser is trained on WSJCAM0 training set and adapted to one accent from the ABI-1 corpus using the MLLR approach. For adaptation, the ABI-1 training subset (includes SPB, SPC, shortphrase, and shortsentence) was used to create 15 phone recognisers with accent-specific acoustic models.

All phone recognisers use 39 dimensional MFCCs. Given our 3-fold cross validation sets, for each of the three test sets we created a bigram triphone level language model which was built based on the phone sequence probabilities rather than from words derived from the WSJCAM0 and ABI-1 training subsets. Please note that, no common speaker exists between the test subset and the training subset that was used to create its corresponding phone level bigram. These bigram triphone grammars are used during the phone recognition process. A triphone dictionary with 8875 triphone entries from a phoneme set of size 44 is constructed using the WSJCAM0 and ABI-1 training subsets and using the BEEP pronunciation dictionary (Section 7.2). All phone HMMs are 5 state HMMs with 3 emitting states without state-skipping, with an 8 component GMM per state. The output of phone recognisers is a sequence of phones used to create accent-specific N-gram (N-gram supervectors). For each N-gram and each of our 15 phone recognisers, we can, in principle, construct a separate phonotactic AID system.

Let D_n be the number of different N-grams that occur in the training data. For each utterance, a D_n dimensional phonotactic supervector is produced whose i-th entry is the relative frequency p_i of the i-th N-gram in the set. The resulting D_n dimensional supervector is then evaluated with the various class SVMs to obtain a classification score. For our 4-gram language model supervectors are of dimension $D_{n=4} = 21,696$, this value is chosen empirically as for larger dimensions of D_n there were not enough examples of certain 4-grams to be able to use its information to discriminate between different accents.

The accent identification rate of 14 accent-specific PRLMs plus WSJCAM0 PRLM with $N = 2,3,4$ and their fusion are presented in Table 6.1. As can be seen by increasing the order of the N-grams from $N = 1$ to $N = 4$, the accuracy of all systems improves consistently.

However, for language models with $N = 5,6$ the accuracy reduces presumably because more training data would be needed to train the N-gram frequency supervectors of larger dimensions. The results show that for different orders of N-gram ($N = 2,3,4$) on average all phonotactic systems perform at a similar accuracy level (between 57.9% and 43.9%) with the ilo accented phone recognizer outperforming the other systems slightly on average (57.9%). Each phone recogniser performs best on a few accents and doesn't perform adequately on some other accents.

This experiment was also carried out with fused $N = 2,3$ and $N = 2,3,4$, but the accuracy was lower than those in fused $N = 3,4$. As can be seen, even fusion of phonotactic systems for different combination of N-grams ($N = 3,4$) did not provide further improvement in the identification rate compared to the 80.65% accuracy achieved by fusion of 4-grams from 15 systems only.

The results reported in Table 6.1 show that the fusion of several phonotactic systems with different phone recognizers provides a higher accuracy compared to any of the individual systems, and the best performance of 80.65% belongs to the fused 4-grams.

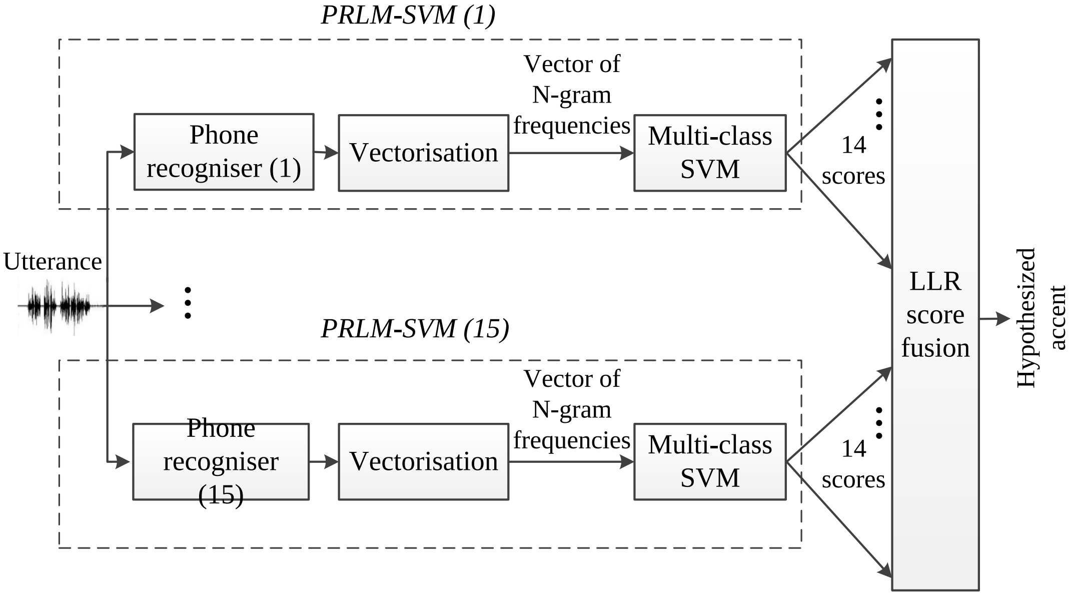

Fig. 6.1 *Using LLR to fuse the scores from* 15 *parallel PRLM-SVM AID systems*

The best phonotactic AID result is obtained by applying the LLR fusion to the outputs of 15 individual phonotactic systems with 4-grams, using Brummer's multi-class linear logistic regression (LLR) toolkit [53]. The confusion matrix for the best phonotactic AID experiment

Table 6.1 *Summary of the AID accuracies for 15 PRLM systems and their fusion*

Accuracy of phonotactic systems (%)	N-gram	2-gram	3-gram	4-gram	3,4-gram fused
brm	Northern	39.44	40.12	52.11	54.54
eyk		52.09	56.69	61.63	64.78
lan		48.61	54.94	56.35	62.33
lvp		47.53	47.54	53.87	61.61
ncl		44.36	54.21	52.48	57.74
nwa		45.41	54.20	53.52	61.97
ilo	Southern	51.74	57.02	64.79	67.59
sse		49.64	49.64	60.22	61.63
ean		46.45	50.34	56.34	61.27
crn		52.09	56.69	61.63	64.78
roi	Irish	45.40	55.99	57.04	61.97
uls		39.08	45.07	55.30	59.51
shl	Scottish	44.35	49.64	58.45	62.67
gla		38.73	49.67	53.88	56.70
WSJCAM0	-	49.30	57.72	60.18	67.95
Fusion of 15 systems	English	72.54	79.93	80.65	79.59

is shown in Table 6.2 (Acc. refers to the AID accuracy). In this table, majority of accents are correctly recognised.

However, it is not clear why a few speakers with ilo accent are recognised as brm. A large number of speakers from various accents (ean, brm, eyk, nwa, shl, crn) appear to exhibit sse (standard southern English) accent, this may be due to the speakers' social or educational status.

The lowest identification error rate of 10% occurred for gla (Glaswegian), uls (Ulster), and ncl (Newcastle) accents which can be due to the fact that these accents are very different from other accents in terms of their phonotactic properties. The lowest accuracy of 68% belongs to the ean (East Anglia) accent, as a considerable population of its speakers were identified to exhibit sse accent.

6.2.2 I-vector system

The theory behind our i-vector AID system is fully described in Section 2.4.2 and its main stages are summarised in Figure 2.2. Safavi [152] created the original version of this i-vector system for the speaker verification task using the Microsoft toolbox [153], and during this research we applied further modifications to adopt this to our accent recognition task.

The i-vector system is trained with 19 MFCCs plus 49 Shifted-Delta Cepstral coefficients (SDC) with a 7-3-1-7 configuration [154], giving a total of 68 features per frame (described in Appendix E). Studies showed that incorporating SDC features as inputs to language/accent

Table 6.2 *Confusion matrix for the best phonotactic system (fusion of 4-gram results from 15 PRLMs)*

Accent code	Accent group	Acc.	brm	eyk	lan	lvp	ncl	nwa	ilo	sse	ean	crn	roi	uls	shl	gla
brm	Northern	75%	15	0	0	0	0	0	1	1	2	0	1	0	0	0
eyk		76%	0	19	2	0	1	1	0	1	0	1	0	0	0	0
lan		76%	0	3	16	0	0	1	0	0	0	0	0	0	0	1
lvp		85%	0	1	0	17	0	1	1	0	0	0	0	0	0	0
ncl		90%	1	0	0	1	18	0	0	0	0	0	0	0	0	0
nwa		81%	0	0	0	1	1	17	0	1	1	0	0	0	0	0
ilo	Southern	71%	2	0	0	0	0	0	15	0	3	0	1	0	0	0
sse		69%	0	0	0	0	0	2	0	11	2	1	0	0	0	0
ean		68%	0	0	0	0	0	0	0	6	13	0	0	0	0	0
crn		85%	0	0	0	0	0	0	1	1	0	17	1	0	0	0
roi	Irish	84%	0	0	0	0	0	1	0	0	0	0	16	2	0	0
uls		90%	0	0	0	0	0	0	0	0	0	0	2	18	0	0
shl	Scottish	86%	0	0	0	0	0	0	0	1	0	0	1	0	19	1
gla		90%	0	0	0	0	0	0	0	0	0	0	0	0	2	18

recognisers improves the recognition accuracy and our study confirms this. The values chosen for our SDC coefficients were suggested to be the best for an acoustic based accent recognition [56, 155]. Our acoustic-based accent recognizer used band-limited speech (0.23 to 5.25 kHz) rather than the full frequency bandwidth as it has been shown that this band contains more accent information and less speaker specific information [156].

In our system, the UBM was trained on the training subset of the ABI-1 corpus using various number of UBM components and T-matrix ranks (number of UBM components: 512, 1024 and T-matrix rank: 200, 400, 800). For each utterance, an i-vector is produced and a linear SVM is applied to identify the accents of speakers. The average identification rate over SPA utterances (test subset of ABI-1) after 3-fold cross validation is reported in Table 6.3. The best accuracy (76.76 %) is achieved using the i-vector system that has a UBM with 512 Gaussian mixture components, and a total variability T-matrix of rank 800.

Table 6.3 *Summary of the i-vector AID results for different numbers of UBM components and T-matrix ranks*

Accuracy (%)	T-matrix rank: 200	T-matrix rank: 400	T-matrix rank: 800
UBM:256 GMM	65.30	65.35	69.38
UBM: 512 GMM	67.95	74.30	76.76
UBM: 1024 GMM	70.78	74.29	75.35

Table 6.4 *Confusion matrix for the proposed i-vector system (T-matrix rank of 800, UBM with 512 GMM componenets)*

Accent code	Accent group	Acc.	brm	eyk	lan	lvp	ncl	nwa	ilo	sse	ean	crn	roi	uls	shl	gla
brm	Northern	80%	16	0	0	0	0	1	0	1	1	1	0	0	0	0
eyk		84%	1	21	2	0	1	0	0	0	0	0	0	0	0	0
lan		76%	1	0	16	0	1	1	1	0	1	0	0	0	0	0
lvp		85%	0	0	1	17	0	2	0	0	0	0	0	0	0	0
ncl		65%	0	0	2	1	13	0	0	0	1	0	0	0	2	1
nwa		52%	1	4	1	0	1	11	0	0	0	2	0	0	1	0
ilo	Southern	57%	2	1	3	0	0	0	12	0	2	0	0	0	0	1
sse		69%	0	2	0	0	0	1	2	11	0	0	0	0	0	0
ean		84%	1	1	0	0	0	0	1	0	16	0	0	0	0	0
crn		55%	0	1	0	0	1	1	3	1	1	11	0	0	1	0
roi	Irish	78%	0	0	0	0	0	0	0	0	0	0	15	4	0	0
uls		90%	0	0	0	0	0	0	0	0	0	0	2	18	0	0
shl	Scottish	100%	0	0	0	0	0	0	0	0	0	0	0	0	22	0
gla		95%	0	0	0	0	1	0	0	0	0	0	0	0	0	19

The AID results for the most accurate i-vector accent recognition system are summarised in Table 6.4. In this table, most of the recognition results are as expected. Unlike the phonotactic system, the i-vector system does not mis-recognise a large number of speakers from other accents as sse. Also it achieves a much better accuracy on the Scottish accents (gla, shl), in fact it achieves the highest accuracy for these two Scottish accents. The performance of the i-vector system on Scottish accents is the highest among all the AID systems reviewed in this section (even the ACCDIST-SVM system).

However, a number of Irish accents (roi, gla) are recognised as each other. Additionally, nwa and crn accents are recognised as other accents within their accent region (northern and southern respectively) and their recognition rate falls by approximately 35% by applying the i-vector rather than the phonotactic method, and leaves them as the most mis-recognised accents in the table. In addition to that, there is a considerable confusion between the recognition of northern and southern English accents, for example a large number of ilo speakers are identified to exhibit northern accents (brm, eyk, lan), and a few brm speakers are mis-classified as having southern accents (sse, ean, crn).

6.2.3 Acoustic-phonotactic fused system

Our best text-independent (unsupervised) accent identification rate of 84.87% is achieved by fusing our best acoustic and phonotactic systems which have individual accuracies of 76.76% and 80.65%. Our fused AID system consists of an acoustic system based on i-vectors (T-matrix of rank 800, and 512 component UBM) and 15 parallel PRLM phonotactic systems, each described with full details in sections 6.2.2 and 6.2.1 respectively. The fusion is carried out using Brummer's multi-class LLR fusion toolkit [52]. Figure 6.2 illustrates our acoustic-phonotactic fused system.

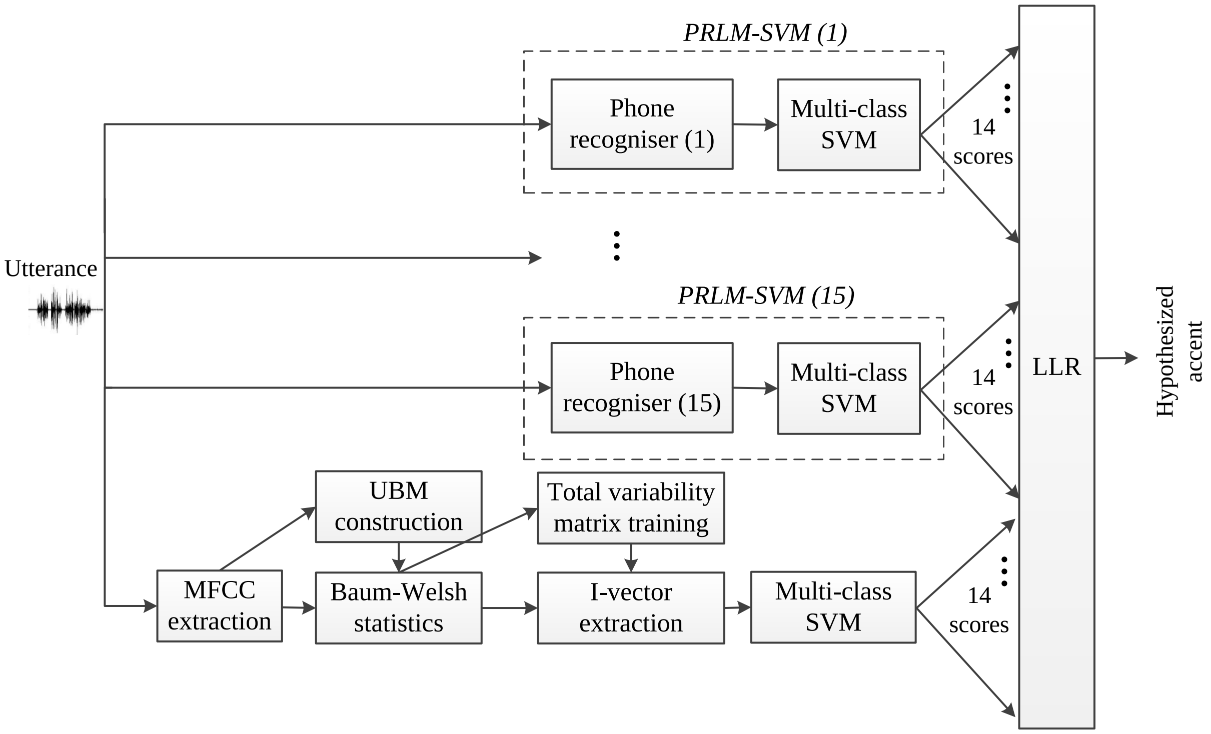

Fig. 6.2 *Fusing the scores from* 15 *parallel PRLM-SVM and the i-vector systems using LLR*

Table 6.5 shows the confusion matrix corresponding to the fusion of i-vector and phonotactic systems. The lowest identification rate 71% belongs to the nwa and it can be seen that speakers from this region are mainly mis-recognized as speakers with a lvp accent, due to similarity of their accent properties. The best identification rate of 95% is achieved for lan, gla and shl. For nwa and crn that were previously largely mis-classified by the i-vector system as other northern or southern accents respectively, this fusion had a very positive effect.

Table 6.5 shows that, after the fusion, the pattern in the mis-recognitions is more similar to the phonotactic system which is the more accurate system rather than the i-vector's, and different northern accents are quite often mis-recognised as each other, for example ean is largely mis-classified as sse.

Table 6.5 *The confusion matrix for the acoustic-phonotactic fused AID system (Acc. refers to accuracy)*

Accent code	Accent group	Acc.	brm	eyk	lan	lvp	ncl	nwa	ilo	sse	ean	crn	roi	uls	shl	gla
brm	Northern	80%	16	0	0	0	0	0	0	2	2	0	0	0	0	0
eyk		92%	0	22	1	0	0	0	0	1	0	1	0	0	0	0
lan		95%	0	2	19	0	0	0	0	0	0	0	0	0	0	0
lvp		85%	0	1	0	17	0	1	1	0	0	0	0	0	0	0
ncl		85%	0	1	0	1	17	1	0	0	0	0	0	0	0	1
nwa		71%	1	0	0	2	1	15	0	0	1	0	0	0	0	0
ilo	Southern	77%	1	1	0	1	0	0	17	1	2	0	0	0	0	0
sse		75%	0	0	1	0	0	0	1	12	1	0	0	0	0	0
ean		75%	0	1	0	0	0	0	0	4	15	0	0	0	0	0
crn		85%	0	1	0	0	0	0	3	0	0	17	0	0	0	0
roi	Irish	84%	0	0	0	0	1	0	0	0	0	0	16	2	0	0
uls		90%	0	0	0	0	0	0	0	0	0	0	2	18	0	0
shl	Scottish	95%	0	0	0	0	0	0	0	1	0	0	0	0	21	0
gla		95%	0	0	0	0	0	0	0	0	0	0	0	0	1	19

6.2.4 ACCDIST-SVM system

The ACCDIST-SVM text-dependent AID system is fully described in Section 2.4.3 and summarised in Figure 2.4. This system was implemented by Hanani[56]. During this research, the supervectors generated by the ACCDIST-SVM system are used for the accent feature visualisation experiment, and the identification result of this system is also used in our acoustic model selection for the speech recognition experiments.

In Hanani's ACCDIST-SVM system, a transcription of each SPA recording from ABI-1 corpus was force-aligned with the speech data, and the most common vowel triphones were found. This phone level transcription is produced using the BEEP dictionary. Here, the $P = 105$ most common triphones across all speakers in the training data were found (P empirically selected). During the vectorisation stage, each vowel-triphone is expressed in the form of a 41 dimensional vector and used in constructing the vowel distance tables for each speaker. Each distance table of dimension 105×105 contains 5460 distance entries. For every utterance, a 5460 dimensional supervector is constructed as a result of vectorization of the speaker distance tables. A multi-class SVM with correlation distance kernel is applied to the supervectors of all accents each of size 5460.

Using 43s of speech (approximate length of an SPA recording including silence), the text-dependent (supervised) ACCDIST-SVM system achieves 95% accuracy. Hanani's ACCDIST-SVM has the lowest error rate compared to the other accent classification techniques which were applied to ABI-1, and fully described in [17, 56, 63, 72].

Table 6.6 *Confusion matrix for the ACCDIST-SVM based AID system (Acc. refers to accuracy)*

Accent code	Accent group	Acc.	brm	eyk	lan	lvp	ncl	nwa	ilo	sse	ean	crn	roi	uls	shl	gla
brm	Northern	95%	19	0	0	0	0	0	0	1	0	0	0	0	0	0
eyk		96%	0	24	1	0	0	0	0	0	0	0	0	0	0	0
lan		100%	0	0	21	0	0	0	0	0	0	0	0	0	0	0
lvp		100%	0	0	0	20	0	0	0	0	0	0	0	0	0	0
ncl		90%	0	0	0	0	18	2	0	0	0	0	0	0	0	0
nwa		95%	0	0	0	0	0	20	0	0	0	0	1	0	0	0
ilo	Southern	91%	1	0	0	0	0	0	19	0	0	0	1	0	0	0
sse		94%	0	0	0	0	0	0	0	15	1	0	0	0	0	0
ean		79%	0	0	0	0	0	0	0	4	15	0	0	0	0	0
crn		75%	0	0	0	0	0	1	0	3	1	15	0	0	0	0
roi	Irish	95%	0	0	0	0	0	1	0	0	0	0	18	0	0	0
uls		100%	0	0	0	0	0	0	0	0	0	0	0	20	0	0
shl	Scottish	100%	0	0	0	0	0	0	0	0	0	0	0	0	22	0
gla		95%	0	0	0	0	0	0	0	0	0	0	0	0	1	19

Table 6.6 summarises the recognition results for the ACCDIST system which is the most accurate AID system. In this table, the majority of the recognition results are correct. However, there are a few mis-classifications within different northern accents, for example eyk and ncl are recognised as lan and nwa respectively which might be due to geographical proximity of these regions.

Additionally, two southern accents, namely ean and crn are largely mis-recognised as sse which might be due to social status or educational factors influencing the accents of the speakers. Accuracy of 100% is achieved in recognition of lan, lvp, uls and shl accents. In general ACCDIST was highly successful in recognising Scottish, Irish, and southern English accents, and the majority of confusions in classifications occurred in recognition of different southern English accents.

6.3 Comparison of experimental results

In this section the performance of our acoustic, phonotactic, and acoustic-phonotactic fused unsupervided AID systems are compared against DeMarco's i-vector [17, 63], Hanani's phonotactic, and Hanani's acoustic-phonotactic fused [56] systems respectively. The supervised ACCDIST AID system [56] used in this work is also compared with the ACCDIST system proposed by Huckvale [72].

Comparison of the proposed phonotatic system with that of Hanani

Hanani uses four parallel Multi-lingual PRLM systems comprising of four phone recognisers with N-gram ($N = 2,3,4$) language models trained on non-English (Czech, Hungarian, Russian) and English (trained on ABI-1 corpus material) data [56]. In his system, each language has a different language-specific phoneme set. In total Hanani's phonotactic fused system with accuracy of 82.1% consists of 4 parallel PRLM systems, with 2-gram, 3-gram and 4-gram (12 systems). The dimension of Hanani's phonotactic AID supervectors after the fusion of N-grams with $N = 2,3,4$ for English, Czech, Russian, and Hungarian are 300523, 328230, 290205, and 272894. The dimension of the supervectors produced by Hanani's phonotactic system after fusing all 12 systems is equal to 1,191,852.

Our English multi-accented parallel PRLM based phonotactic system uses one general English (trained on WSJCAM0), and 14 accent-specific (trained on WSJACAM0 and MLLR adapted to ABI-1 accents) phone recognisers, with a 4-gram language model (in total 15 PRLM systems). The accuracy of our phonotactic fused system is 80.65% and the dimension of the supervectors produced by our phonotactic system after fusing the 4-grams from all 15 systems is equal to 325,440 (15 × 21,696).

Hanani's and our proposed phonotactic fused results for different N-grams are shown in columns 3 to 5 of Table 6.7. These results suggest that our phonotactic fused system has outperformed that of Hanani's with relative AID error rate reduction of 24.18%, 14.43%, and 1.3% for N-grams language models with N = 2, 3, and 4 respectively.

For recognising accents of ABI-1 SPA utterances, Hanani's system (fusion of 4 language specific systems with $N = 2,3,4$) results in 1.76% relative improvement in accuracy compared to our phonotactic system (fusion of 15 systems with $N = 4$). However, our system works with supervectors of much lower dimensionality compared to that of Hanani. Hence, it is more practical for real-time applications, where the online classification of these supervectors is required for AID.

Comparison of the proposed acoustic-based system with Demarco's and Hanani's

Demarco's acoustic fused AID classifier [17] combines the results from 630 i-vector systems and hence it is more complex than ours. A set of projection methods are applied to different i-vector systems constructed using various T-matrix ranks (100, 150, 200, 250, 300, 350, and 400) and multiple UBMs with GMM components of different sizes (64, 128, 256, 512, and 1024).

Demarco's i-vector implementation was then analysed under various other projection methods in [17], using LDA, SDA (Semi-supervised Discriminant Analysis) [157], NCA

Table 6.7 *Summary of Hanani's and our proposed phonotactic AID results*

Accuracy(%)		2-gram	3-gram	4-gram	2,3,4-gram Fused
Czech	Hanani's	37	62.7	68.7	70.5
Hungarian		43	65	71	73
Russian		48.2	62	63	69
English		38.6	53	58	61
Phonotactic fusion		55	68.4	79.6	82.1
Accuracy(%)		2-gram	3-gram	4-gram	3,4-gram Fused
Phonotactic fusion	Proposed	72.54	79.93	80.65	79.59

(Neighbourhood Component Analysis) [158], and RLDA (Regularized LDA) [159]. Further more, at a final level, a genetic algorithm was employed over all projection methods, all GMM component sizes and all T-matrix sizes, to produce a final selection of i-vector classifiers, termed 'weak learners' (fusing 630 from 2520 possible systems), which, when used with a majority voting classifiers, provides a further improved AID performance of around 81%. Demarco assumed that each of these projection methods extracts different 'aspects' of the accents in question.

Despite the high accuracy of Demarco's i-vector system, it is not well suited for the practical applications due to its high complexity. Hanani's acoustic fused accent recognition system comprises 27 complex acoustic systems with a 4096 component UBM, namely, a GMM–UBM, a GMM-SVM-GMM, a GMM–SVM, 12 language specific GMM-uni-grams and 12 language specific GMM-bi-grams. His GMM-uni-grams and GMM-bi-grams results comprise 12 language specific phone recognisers trained on, namely English, Arabic, Farsi, French, German, Hindi, Japanese, Korean, Mandarin, Spanish, Tamil and Vietnamese from the CallFriend corpus [16].

Table 6.8 summarises the results from three acoustic systems applied to the AID task. Compared with the 630 fused i-vector systems proposed by Demarco, our simple i-vector system offers significant simplification at a cost of a 5% increase in error rate. The choice of which system to use depends on the application. It is also worth noting that Demarco's single best i-vector system has accuracy of 68% (UBM with 256 GMMs, and T-matix rank of 300) which is less accurate than our best single i-vector system with 76.76 % accuracy [63]. Our single i-vector based acoustic system achieved a higher accuracy compared to Hanani's more complex acoustic system.

Table 6.8 *Summary of results for the acoustic-based AID systems*

	Accuracy (%)	Number of systems fused together
Hanani's acoustic fused	75.60	27
Demarco's acoustic fused	81.05	630
Proposed acoustic system	76.76	1

Comparison of proposed acoustic–phonotactic fused system with Hanani's

Hanani's acoustic-phonotactic fused system is obtained by fusing 27 acoustic systems and 12 PRLM phonotactics using Brummer's multi-class LLR fusion toolkit. His system comprises 29 phonotactic and acoustic based AID systems, namely a GMM–UBM, a GMM-SVM-GMM, a GMM–SVM, 12 language specific GMM-uni-grams and GMM-bi-grams, and 4 language specific PRLM systems with 3 order of N-grams ($N = 2,3,4$).

Hanani's acoustic fused, phonotactic fused, and acoustic-phonotactic fused systems have identification rates of 75.6%, 82.1%, and 89.60% respectively. His acoustic-phonotactic fused system achieves the highest accuracy compared to all the unsupervised AID systems tested on ABI-1 and reported in this work.

Our acoustic-phonotactic fused system with 85.87% accuracy, consists of 16 systems, namely one i-vetor and 15 phonotactic based AID systems with identification rates of 76.76% and 80.65% respectively. The dimensions of the supervectors used in our i-vector and phonotactic AID systems are 800 and 325,440 (15 × 21,696) dimensions respectively.

Table 6.9 *Summary of the AID results for the acoustic–phonotactic fused systems*

Accuracy (%)	Acoustic	Phonotactic fused	Acoustic–phonotactic fused
Hanani's system	75.60 (fused)	82.14	89.60
Proposed system	76.76 (single)	80.65	85.87

Due to the lower dimensionality of supervectors and smaller number of fused systems in our acoustic-phonotatic system compared to that of Hanani, the on-line computational costs, for LLR score fusion and SVM classification, would be much lower. Hence, our less complex system is advantageous over that of Hanani in real-time applications.

Comparison of the Hanani's ACCDIST-SVM system with Huckvale's system

Supervised AID systems require the transcription from labelled data. Hanani's ACCDIST-SVM and Huckvale's ACCDIST system are both supervised AID systems. Hanani's ACCDIST-SVM system achieves 95% accuracy over all ABI-1 SPA utterances. Huck-

Table 6.10 *ABI-1 regions and corresponding broad accent groups*

Broad accent groups	Northern						Southern				Irish		Scottish	
Accent regions	brm	eyk	lan	lvp	ncl	nwa	ilo	sse	ean	crn	roi	uls	shl	gla

vale's system has an identification rate of 92.3% over all 'shortsentence' utterances from ABI-1.

The ACCDIST method proposed by Huckvale [72] (Section 2.4.3) is less flexible than Hanani's ACCDIST-SVM as it requires each utterance to correspond to exactly the same phone sequence. For training and testing, Hanani's ACCDIST-SVM uses SPA utterances, while Huckvale's ACCDIST uses the shortsentence utterances from ABI-1.

For construction of the vowel distance table the ACCDIST-SVM uses the realisation of vowels in triphone contexts, unlike the ACCDIST system that uses the realisation of vowels in specific word contexts. In addition, for determining the closest accent group to the test speaker, the ACCDIST-SVM classifier is based on SVMs with correlation distance kernel, rather than correlation distance.

6.4 Visualisation of the AID feature space

In this section we show the visualisation maps for three AID systems, namely phonotactic, i-vector, and ACCDIST-SVM. These visualisation results are based on our proposed approach for visualisation of the AID feature space which has been explained in Section 2.5.

In this section, Table 6.10 summarises different accent regions and their correspond broad accent groups to further facilitate our analysis.

6.4.1 Phonotactic feature space

Given the phonotactic supervectors (described in Section 2.4.1) containing the 4-gram language model frequencies from the phonotactic based AID trained on WSJCAM0 data (described in Section 6.2.1), we aim to illustrate the visualisation of different accent groups and analyse their distributions in this space.

Figure 6.3 shows a 2-dimensional projection of an example phonotactic feature space. This feature space is constructed as a result of applying EM-PCA and LDA to the phonotactic supervectors of length $D = 21,696$ and reducing the dimension to $n = 65$ and $n = 2$ respectively. For each accent region, 1-standard-deviation contours from the mean value represent utterances of that accent in the accent space.

Three overlapping clusters corresponding to northern and southern England and Scotland, are formed in the phonotactic feature space. All the northern accents, namely ncl, nwa, lvp, lan, eyk are located close to each other except for the Birmingham data which is separated from the southern and northern England clusters. This observation is consistent with the recognition results shown in Table 6.2, as all the northern accents have been mis-recognised as each other but the brm accent was often mis-classified as a southern accent.

This might be due to social status or educational level of the speakers, as the material for the Birmingham accent was recoded in the University of Birmingham. In fact, this is the case for the Birmingham (brm) accent cluster in all the visualisation results; despite the fact that linguists consider this accent as a northern accent, its cluster always lies separately somewhere between other northern and southern accents.

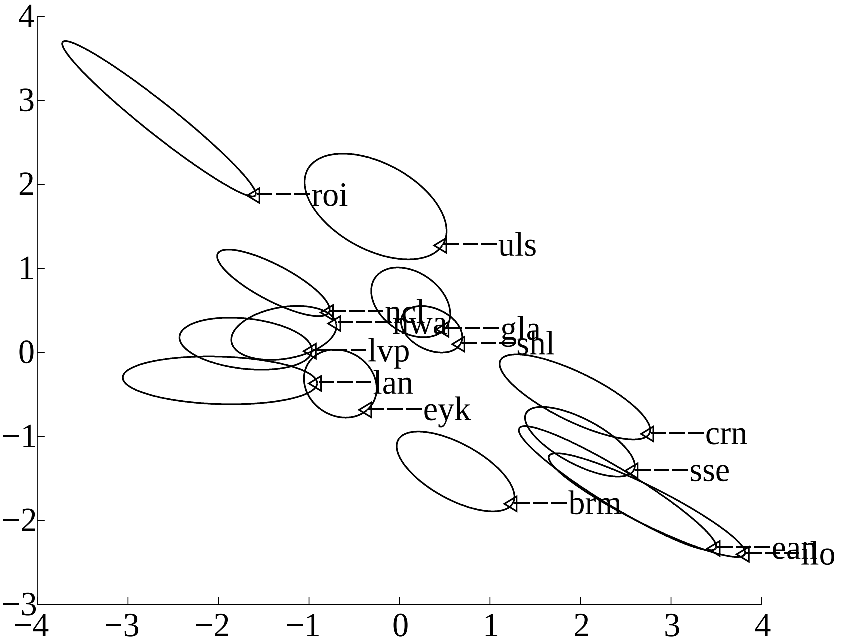

Fig. 6.3 *Visualization of the phonotactic based AID feature space*

There is a region at the top of the figure corresponding to the two Irish accents roi and uls. Geographical proximity of Liverpool, Lancashire, and East Yorkshire is evident from the proximity of their accent groups (lvp, lan, eyk). The two Scottish accents are located very closely to each other in the centre of the space, clearly separated from Irish, northern and southern England accents. Similar to the ACCDIST-SVM visualisation space (Figure 6.5) crn and ean have a considerable overlap with sse, which emphasises that these two accents exhibit similar accent properties as the standard southern English (sse) accent. Table 6.2

shows that frequently speakers with ean and crn accents are recognised to exhibit sse accent, and speakers with sse accent are mis-recognised as either ean or crn.

In Figure 6.3 there are a few unexpected phenomena. Table 6.2 results shows that the two Irish accents, roi and uls, are mis-recognised as each other a few times (despite their similarity). Here, uls is located more closely to gla accent rather than roi, which may reflect the historical and social connections between Glasgow and Belfast. Additionally, ilo and ean have a large overlap despite their differences from the linguistic point of view. Infact even the results from Table 6.2 shows that a number of ilo speakers are incorrectly classified as ean.

In spite of the major differences between the two Scottish accents (gla, shl) and the rest of the accent groups, they are located in the middle of the phonotactic accent space. This did not occur in the ACCDIST-SVM or i-vector accent space shown in Figures 6.5 and 6.4 respectively, as both Scottish accents (gla, shl) are positioned in a far corner of the graph close to one another and far from the other accent groups, as expected due to their considerably different accent properties.

6.4.2 I-vector feature space

Given the 800 dimensional supervectors from the our proposed i-vector system with 512 component UBM and T-matrix of rank 800, we visualise the distributions of accent groups in a two dimensional accent space. Here, the 800 dimensional i-vector space $D = 800$ is projected onto 666 and 2 dimensions using the EM-PCA and LDA algorithms respectively. For each accent region, 0.7-standard-deviation contours from the mean value represent utterances of that accent in the accent space. Figure 6.4 shows a representation of the AID i-vector space for the ABI-1 corpus.

The confusion matrix and visualisation plot for the i-vector system shows that these features are quite successful in separating Irish and Scottish accents. Interestingly, even in two dimensions the major accent properties of the i-vectors can be observed.

Three accent clusters corresponding to northern, Irish and Scottish accents are present in this visualisation map, however, there is no separate cluster for the southern English accents. Looking at the confusion Table 6.4 and the visualisation map, suggests that the i-vector features are not very strong in capturing differences between different northern accents and mis-recognise them as other northern or southern accents. The confusion table shows that a large number of utterances with northern English accent are mis-recognised as southern and a considerable number of southern accented utterances are recognised as northern which leads to a large overlap in the visualisation space between two accent groups. The social or educational factors for some of the northern speakers could be the reason for them being incorrectly identified as southern accents.

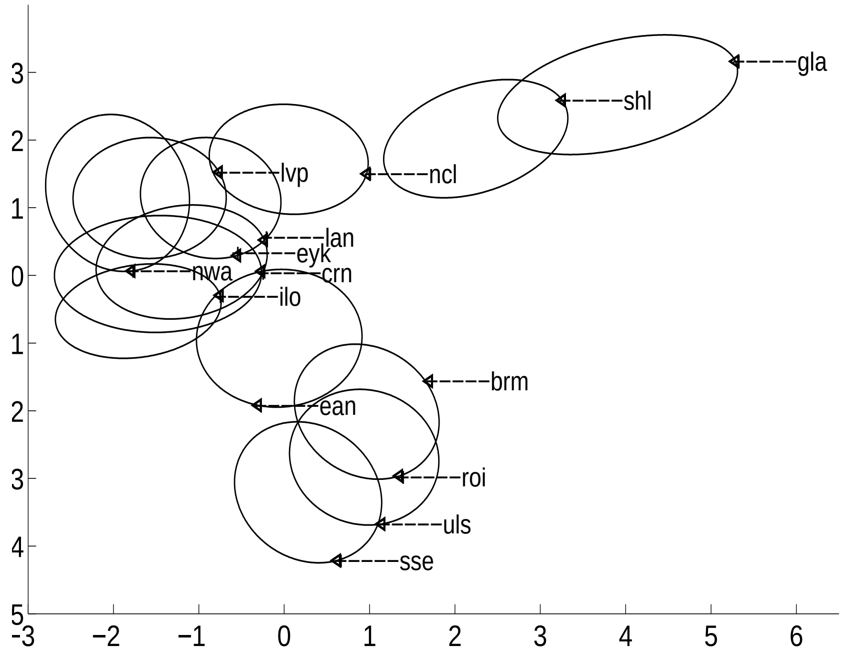

Fig. 6.4 *Visualization of the i-vector based AID feature space*

In the top right, a cluster can be seen for the two Scottish accents gla and shl and these two accents have the biggest distance from the rest of accent groups. Referring to Table 6.4 they are also more distinguishable compared to the rest of accents and correspond to the highest AID performance.

The clusters for the two Irish accents roi and uls are situated on the bottom left side of the visualisation map with large overlap as expected. Similarity between these two accents is also evident from the confusion matrix. In fact, four out of five times the mis-recognition of these accents was due to one Irish accent being mis-recognised as the other Irish accent.

Southern accents are scattered around the centre of the figure and overlap with northern and Irish accent groups. One reason might be the fact that in each accent region we expect to see members who exhibit hints of Southern accent in their speech which could be influenced by social factors.

6.4.3 ACCDIST-SVM feature space

ACCDIST-SVM supervectors (described in Section 2.4.3) contain the vowel-distances for each utterance produced by the ACCDIST-SVM AID system (described in Section 2.4.3). We use the supervectors from the ACCDIST-SVM system to visualise the distributions of accent groups in a two-dimensional ACCDIST-SVM accent space (Figure 6.5).

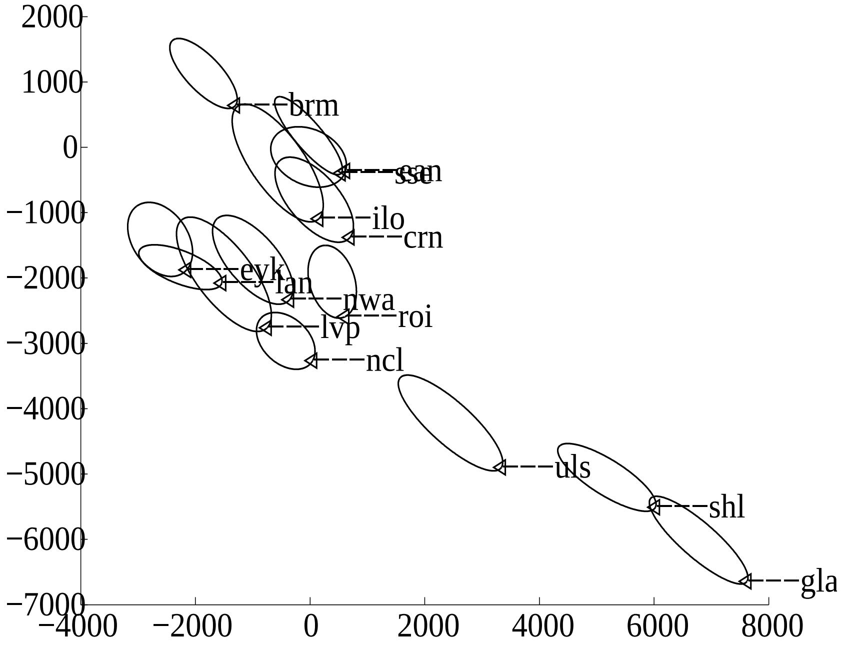

Fig. 6.5 *Visualization of the ACCDIST-SVM based AID feature space*

For visualisation of the ACCDIST-SVM accent space, first, EM-PCA is applied to the supervectors of length $D = 5460$ to reduce the dimensions to $n = 145$ (this value was chosen empirically), then LDA is applied to reduce the dimension to $n = 2$. In the two-dimensional projection of the ACCDIST-SVM supervector space shown in Figure 6.5, each accent group is shown by 1-standard-deviation contours from the mean of the supervectors in that accent.

Figure 6.5 shows 3 distinct clusters, corresponding to northern England, southern England and Scotland, but there is no separate cluster for the Irish accents. The proximity of Belfast (uls) to the Scottish accents (gla and shl) rather than Dublin (roi) can be seen in both phonotactic (Figure 6.3) and ACCDIST-SVM (Figure 6.5) visualisation maps. This proximity may reflect social influences.

The North Wales (nwa) recordings were made in Denbigh (roi accent), which is close to Liverpool, and this explains their location in Fig.6.5. The large overlap among clusters of southern English accents (crn,ilo,sse ean) is evident from the recognition results from Table 6.6.

Unexpected features of Figure 6.5 include the grouping of Birmingham (brm) with the southern English accents, and the positioning of the Dublin (roi) data between the southern and northern English accents.

6.5 Summary and conclusions

In this chapter we investigated the performance and the recognition results for three state-of-the-art AID systems, namely ACCDIST, phonotactic, and i-vector on the ABI-1 corpus. Then the results of fusing the acoustic and phonotactic systems are analysed. As each approach explores a different clue in speech for identification of regional accents, a considerable improvement is gained in recognition accuracy by fusing acoustic and phontactic systems. The AID approach that gives the best AID performance is the supervised ACCDIST-SVM method. This is also the method that leads to the accent space that is most consistent with our understanding of the relationships between the different accents in the ABI-1 corpus.

The summary of unsupervised AID performance on ABI-1 SPA utterances can be found in Table 6.11. Our i-vector system, with 76.76% accuracy, was fused with a phonotactic system, with accuracy 80.65%, in order to add knowledge of phone frequencies to the final AID system, with accuracy of 84.87%, and it helped to reduce the relative recognition error rate by 9.5%. Regarding the phonotactic system, by analysing the results from Hanani's multi-lingual phonotactic fused system provides higher accuracy than our multi-accent English phonotactic fused system. From this we conclude that it is possible that a phonotactic system can improve further by introducing additional phones from other languages (e.g., Czech, Hungarian, Russian).

Table 6.11 *Summary of the acoustic (Ac.) and phonotactic (Phon.) based AID results*

AID systems	Acoustic systems	Phonotactic systems	Final fused AID performance
DeMarco et. al. [17]	AID (630 Ac.) 81.05%	—	Fused (630 Ac.) 81.05%
Hanani et. al. [56]	AID (27 Ac.) 75.60%	AID (12 Phon.) 82.14%	Fused (27 Ac. & 12 Phon.) 89.60%
Our proposed AID Systems	AID (1 Ac.) (T=800, 512 GMM) 76.76%	AID (15 Phon.) 80.65%	Fused (1 Ac. & 15 Phon.) 84.87%

The research presented in this chapter demonstrates that by using a carefully crafted automatic accent identification system whose properties match that of the target problem, we can achieve identification performance at a similar level of those complex AID systems proposed in the literature. In this research the supervectors generated by AID systems were used in visualisation of AID accent space. In addition to that, AID systems will be used by ASR systems in Chapters 7 and 8 for accent-specific acoustic model selection.

Chapter 7

Supervised model selection for accent-robust ASR (GMM-HMM)

7.1 Introduction

In this chapter we initially investigate the effect of accented speech from the ABI-1 corpus on the performance of a basic recogniser trained on southern English speech from WSJCAM0 corpus. Then, we investigate different acoustic model adaptation approaches to mitigate for the effect of accent on speech recognizers. Our acoustic model adaptation methods are concerned with adaptation to a new user's regional accent using MAP and MLLR approaches. In Chapter 6 we showed that, with on average 43.2s of speech (including silence), an individual's accent can be determined with 95% accuracy using supervised AID (ACCDIST-SVM). Thus, we apply this supervised AID system for selecting an appropriate accent-dependent acoustic model. We compare our proposed methods for accent adaptation with conventional speaker adaptation. Our experiments in this chapter answers the following research questions.

- Given limited data from the test speaker, is it better to use that training sample for supervised speaker adaptation, or to use that data for AID and identify a suitable accent-dependent ASR system? or do both?

- Three approaches to accent-dependent modelling are investigated. Is it better to use the data from neighbouring speakers in 'AID space', the 'true' accent of the user (if it is known) or the result of AID to build a suitable acoustic model for ASR?

- For accent adaptation, will MAP provide a better accuracy or MLLR?

7.2 Baseline speech recognition system

In this chapter and Chapter 8, we implemented our British English speech recognizer using the Hidden Markov Model Toolkit (HTK) [2]. It is a phone-decision tree tied triphone HMM based system with 5500 tied states, each associated with an 8 component Gaussian Mixture Model (GMM). The choice of number of GMMs per state is empirically driven. It was trained on the SI training set (92 speakers, 7861 utterances) of the WSJCAM0 corpus of read British English speech [13]. We used the British English Example Pronunciations (BEEP) dictionary [160], extended to include all of the words in the ABI-1 corpus. Our language model consists of the weighted combination of the 5k WSJ0 bigram language model and a bigram language model based on the ABI-1 corpus.

Feature extraction

Initially the waveform is down-sampled to 8 kHz and 25 ms Hamming windows with 10 ms spacing are applied to the signal. Our 39 dimensional MFCC feature vectors comprise 12 cepstral coefficients plus the 0th cepstral coefficient, delta, and acceleration. Cepstral mean value normalisation was applied to the features to reduce the effect of noise.

Pronunciation dictionary

The BEEP dictionary contains all the words that occur in the WSJCAM0 corpus and is designed for large vocabulary ASR [160]. This dictionary contains the phonemic transcriptions of approximately 250,000 English words (including words with multiple pronunciations, and words with non-letter symbols).

The ABI-1 corpus contains some words which were not included in BEEP. The phone level transcription of those words are borrowed from the Carnegie-Mellon University Dictionary (CMU) pronunciation dictionary [161]. Therefore, 61 words with their corresponding transcription were added from CMU to our dictionary (including the words with multiple pronunciations).

The phone set of the BEEP and CMU dictionaries are composed of 44 and 39 phones respectively. During our pronunciation dictionary formation, we first used the BEEP dictionary (containing British standard English pronunciations) and then extracted 61 words from the CMU dictionary to cover some words that are in ABI-1 and not in BEEP.

Language model

If we build an N-gram language model based on the limited sentences of the ABI-1 corpus, the resulting language model will be so restrictive that it limits the underlying acoustic models in dealing with accented speech, and conflicts with the purpose of this work which is to address these acostic-level problems. Conversely, we can not use a grammar trained on WSJCAM0 alone, because it will not contain some of the ABI vocabulary. The solution used in this research is as follows. In order to have a system which works reasonably well for both WSJCAM0 and ABI-1 we combined both WSJ0 and ABI-1 language models.

The experiments reported in this research use a weighted combination of the 5k WSJ0 bigram language model and a bigram language model based on the ABI-1 corpus (excluding the SPA utterances that are used as the development data), so that for a given bigram b the bigram probabilities are combined as shown by Equation 7.1. Since the probabilities in the bigram language models are in log_{10} format their values need to be anti-logged for the computation in Equation 7.1. For different values of λ the combined probability $P_{comb}(b)$ is computed and its value is converted to log_{10} format.

$$P_{comb}(b) = \lambda P_{ABI}(b) + (1-\lambda)P_{WSJ0}(b) \quad where \;\; \lambda \in [0,1] \tag{7.1}$$

In order to design a system which is equally appropriate for the test data from WSJCAM0 and ABI-1, we need to choose a value for lambda which gives equal accuracy to the test data from both corpora. In this section, for our test experiment, we chose half the test utterances from the 5k vocabulary SI-dt-o5 test set in WSJCAM0 corpus, and from ABI-1 we chose the data from sse accent (SPA utterances) as our test utterances.

As shown in Figure 7.1 the accuracy of both WSJCAM0 and ABI-1 systems varies by variation of lambda. For large values of lambda the resulted combined language model gives higher weight towards ABI-1 language model and therefore the accuracy of system becomes higher for ABI-1 test data and lower for WSJCAM0 test data. For lower values of λ the combined language model is biased towards the WSJCAM0 language model so the accuracy on the WSJCAM0 test set is higher.

The choice of $\lambda \in [0,1]$ was determined empirically as 0.175, so that the bigram probabilities are strongly biased towards WSJ0. With this bigram language model we achieve similar error rates of 10.4% on the WSJCAM0 test set and 10% on the sse utterances from ABI-1. The same dictionary and language model was used in all experiments.

For the case where we only use the ABI-1 bigram language model, or equally the combined language model with lambda $\lambda = 1$, the accuracy reaches the highest value of 95.59% for ABI-1. On the other hand, the result of the test on the WSJCAM0 data with only

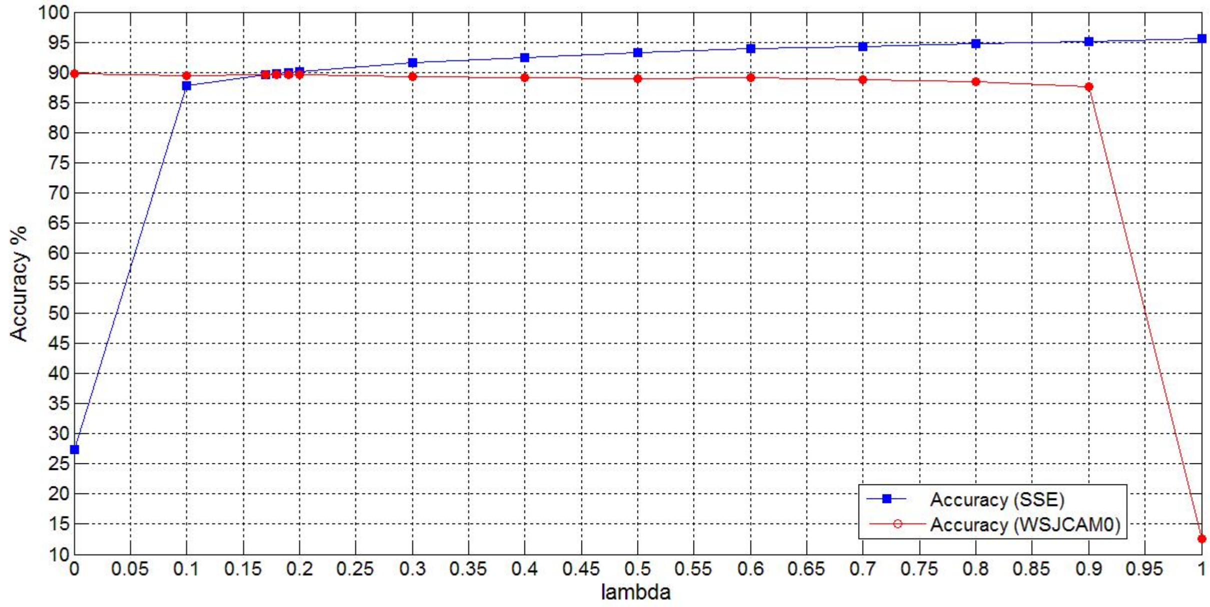

Fig. 7.1 *The effect of changes in the value of λ on the recognition results for the WSJCAM0 SI-dt-o5 test set and all sse utterances from ABI-1*

WSJ0 language model, or equivalently with a combined language model with $\lambda = 0$ shows that the accuracy of system reaches the highest value of 89.74% on WSJCAM0 test set.

7.3 Speech corpora

Here, we summarise key information regarding our speech databases, namely WSJCAM0, and ABI-1.

WSJCAM0 corpus

WSJCAM0 is a British English speech corpus [13]. In our experiments we refer to the training and developement parts of it as WSJT and WSJD respectively (summarised in Table 7.1). In this thesis, we refer to the baseline system trained on the WSJCAM0 training set as the baseline ASR system.

Table 7.1 *Code names for the WSJCAM0 training and developement sets*

Code	Average length	Data section
WSJT	15.50 hours	of WSJCAM0 training set
WSJD	2.25 hours	of WSJCAM0 development set

The ABI-1 Speech Corpus

The ABI-1 speech corpus [12] represents 13 different regional accents of the British Isles, and standard (southern) British English (sse).

For each regional accent, 20 people (normally 10 women and 10 men) were recorded. The subjects were born in the region and had lived there for all of their lives. The experiments in this paper focus on a subset of these texts, namely the 'shortpassages' (SPA, SPB, SPC), the 'shortsentences' and the 'shortphrases'. These are described below and summarised in Table 7.2:

- 'SPA', 'SPB' and 'SPC' are short paragraphs, of lengths 92, 92 and 107 words, respectively, which together form the accent-diagnostic 'sailor passage'. The corresponding recordings have average durations 43.2s, 48.1s and 53.4s.
- 'Short sentences' are 20 phonetically balanced sentences (e.g. 'Kangaroo Point overlooked the ocean'). They are a subset of the 200 Pre-Scribe B sentences (a version of the TIMIT sentences for British English), chosen to avoid some of the more 'difficult' of those sentences, whilst maintaining coverage (146 words, average duration 85.0s)
- 'Short phrases' are 18 phonetically rich short (three- or four-word) phrases (e.g.'while we were away') containing English phonemes in particular contexts in as condensed form as possible (58 words, average duration 34.5s)

Table 7.2 *Code names for the utterances used from ABI-1 corpus in this research*

Code	File name	Average length (seconds) (including silence)	Application ASR
SPA	shortpassagea	43.2	test
SPB	shortpassageb	48.1	adapt
SPC	shortpassagec	53.4	adapt
shortphrase	shortphrase	85	adapt
shortsentence	shortsentence	34.5	adapt

7.4 Experiments on supervised acoustic model adaptation

We applied two types of adaptation, namely speaker adaptation and accent adaptation (accent-dependent acoustic model selection using a supervised AID system). We explore the effectiveness of these two approaches on the accented speech recognition task.

During the adaptation stage, when the adaptation data comes from the test speaker the task is called speaker adaptation, and when the adaptation data comes from the same accent as the test speaker's the task is called accent adaptation. Where the transcription of the data is available the process is called supervised adaptation, and where there is no transcription provided it is unsupervised adaptation.

- **Supervised speaker adaptation:** For each speaker we conducted supervised (using the correct transcription) speaker adaptation (using MLLR or MAP) with 48.1s (SPB), 101.5s (SPB+SPC), 136s (SPB+SPC+'Shortphrases') and 221s (SPB+SPC+ 'Shortphrases'+'Shortsentences') of speaker-dependent data.

- **Supervised accent adaptation:** For each subject in the ABI-1 corpus, the SPA recording was used as test data, and a gender- and accent-dependent model was created by applying supervised accent adaptation to the baseline WSJCAM0 system (using MLLR or MAP). Adaptation used the SPB, SPC, 'shortsentences' and 'shortphrases' (section 5.1.1) data from 9 other subjects with the same gender and accent as the test speaker (approximately 31.5 minutes of speech). In this chapter we applied ACCDIST-SVM as our unsupervised AID (described in Section 6.2.4).

Three approaches to accent-dependent modelling are investigated: using the 'true' accent model, choosing a model using AID, and building a model using data from neighbouring speakers in 'AID space'.

The WSJCAM0 database mainly consists of standard British English speakers, so we expect to find a lower error rate using the baseline system as the accent groups get closer to the sse accent. For accent groups such as gla which are very distant from sse we expect to see the baseline trained on southern English accent has a low accuracy, but when we provide the relevant adaptation data to this system we expect a considerable improvement in the results. For accents that are more similar to the accent properties of the training set we don't expect a large improvement in accuracy after adaptation as the system parameters are already trained on similar data.

In our experiments we report the experiment results in terms of the Word Error Rate (%WER) or the Average Relative WER Reduction (%AWR). For each experiment exp_i the average %WER for all the 14 accents are computed and stored in $\%WER_{exp_i}$. The average %WER reduction for exp_2 relative to exp_1 is denoted by % AWR(exp_1, exp_2) or simply %AWR, and computed as shown by Equation 7.2.

$$\%AWR(exp_1, exp_2) = [(WER_{exp_1} - WER_{exp_2})/WER_{exp_1}] * 100 \tag{7.2}$$

Baseline experiment on the ABI-1 corpus (B0)

We used the baseline WSJCAM0 speech recognition system with the extended WSJ0 5k bigram grammar to recognise the SPA recording for each subject in the ABI-1 corpus. The purpose of this experiment was to measure the effect of regional accent on the performance of a 'standard' British English ASR system.

Adapting to sse accent (B1)

We were concerned that performance improvements resulting from accent adaptation might actually be due to adaptation to the ABI-1 task. Since the recordings in WSJCAM0 are already close to sse, by adapting the baseline system using the ABI-1 sse adaptation data and then testing on all of the ABI-1 accents we can measure the amount of task adaptation. This is the purpose of B1.

Accent-dependent models using 'true' accent (B2)

In these experiments we use the 'true' accent of each ABI-1 subject to apply the true accent-dependent models. The aim of this and the following experiment is to investigate whether it is better to rely on speaker adaptation, or the accent labels defined by the criteria used to select speakers in each region or to the accent label defined by our accent recognizers for the accent specific acoustic model selection.

Supervised speaker adaptation (S0)

The accent-dependent ASR experiments based on AID that follow use AID results from 43.2s of speech. This raises two questions: (1) Is it better to use this speech for AID, so that an accent-dependent model can be selected, or directly for speaker adaptation? (2) How much speech from an individual is needed to achieve results from speaker adaptation that are comparable with the use of an accent-dependent model? (3) Or both? To answer these questions we conducted speaker adaptation experiments for each ABI-1 subject, using supervised (S0) adaptation.

Accent-dependent models chosen using supervised AID (S1)

In these experiments, for each subject, speech recognition is performed using the accent-adapted model corresponding to the result of AID for that speaker, using supervised ACCDIST-SVM AID (S1).

We explore whether the ASR performance will change when an AID system identifies the accented acoustic model rather than using the accent labels defined by the criteria used to select speakers in each region.

ASR Model based on *N* closest speakers in AID space (S2)

Each of these approaches (B2 and S1) to acoustic model selection treat regional accents as well-defined, disjoint phenomena with clear boundaries, whereas in reality this is not the case. Individuals who were born in the same region and have lived there for all of their lives, can still exhibit quite different patterns of pronunciation, and most users will have lived in several different locations during their lifetime. There is clearly considerable variation within an accent group and near ‘accent boundaries’ there may be individuals whose speech exhibits patterns of pronunciation associated with several regional accents. This is likely to be typical of individuals who have lived in many different geographical regions. This is the motivation for the final techniques that are investigated. In our AID systems we identify the set of N speakers who are ‘closest in accent space’ to the new speaker, again using just 43.2s of speech. All of the data associated with these N speakers is then used to create an ASR model.

The aim of this experiment is to investigate whether it makes any difference if the adaptation data comes from the N closest speakers in the accent space or from speakers who speak the same accent as the test speaker according to AID.

In (S2), each ABI-1 speaker s, is represented as a phonotactic supervector V_s. Given a test speaker s and speaker t the correlation distance $C(V_s, V_t)$ is calculated between V_s and V_t for each ABI-1 subject t, and the N speakers for which the correlation distance $C(V_s, V_t)$ is largest are identified. A new model is then constructed by adapting the baseline WSJCAM0 model using the adaptation data from these N speakers. The values of $N = 3, 9, 12, 18, 27, 36$ where tested and $N = 9$ (S2, 31.5 minutes of adaptation speech) was chosen after 2-fold cross-validation.

7.5 Visualisation of the accent space (ACCDIST-SVM)

The ACCDIST-SVM AID approach is fully described in Section 6.2.4. This supervised AID system is based on the ACCDIST-SVM measure [72].

In this chapter we use a list of different accent regions and their corresponding broad accent groups for analysing the ASR results (shown in Table 6.10). In addition to that, we will use the ACCDIST-SVM accent space visualisation result (Figure 6.5) to decide which accents are similar. In our analysis, overlapping clusters are assumed to correspond to similar accents. Since the baseline ASR system is trained on southern English accented data (WSJCAM0) we expect high recognition accuracy for accents that are similar to sse and their corresponding accent cluster is located closely to the sse accent cluster. The accents whose accent clusters is far from that of the sse are considered as difficult accents. We expect to see low accuracy for those accents whose cluster is far from sse due to the large mismatch between the training and test accent properties.

7.6 Results and discussion

In this section we apply various techniques in an attempt to improve the accuracy of the speech recogniser on accented data. We analyse the results of the supervised speaker and accent adaptation using either MAP or MLLR approaches. We investigate whether MAP provides a low WER for the given task or MLLR.

The detailed results of all of the experiments described in Section 7.4, are shown in Figures 7.2 and 7.3. Speech recognition experiments are carried out using a baseline ASR system trained on WSJCAM0 (B0). The percentage word error rates (%WER), after applying different techniques to the accented ASR task, for each regional accent are shown in the graphs and labelled with their corresponding approach.

In Figures 7.2 and 7.3, the accents are ordered according to the performance of the baseline system, trained on WSJCAM0, and tested on the accented test data (B0).

Please note that the majority of code names in this chapter contain letter S which stands for Supervised. The supervised can be used in the context of either supervised speaker adaptation or supervised accent adaptation.

7.6.1 Supervised speaker and accent adaptation using MAP

In this section we apply MAP adaptation to the speaker and accent adaptation task and analyse the results shown in Figure 7.2, using the knowledge from Figure 6.5 regarding the accent distributions.

7.6.2 Baseline experiment (B0)

The poorest (59 %WER) is for the Glasgow accent (gla), which is also the furthest from sse in Figure 6.5. Error rates tend to be higher for the northern English accents, and lower for the southern accents, which is also consistent with the accent visualisation Figure 6.5.

The ACCDIST AID feature space showed that the brm accent is closer to southern accents than northern. Interestingly, the baseline recognition result for brm (14.51 % WER))

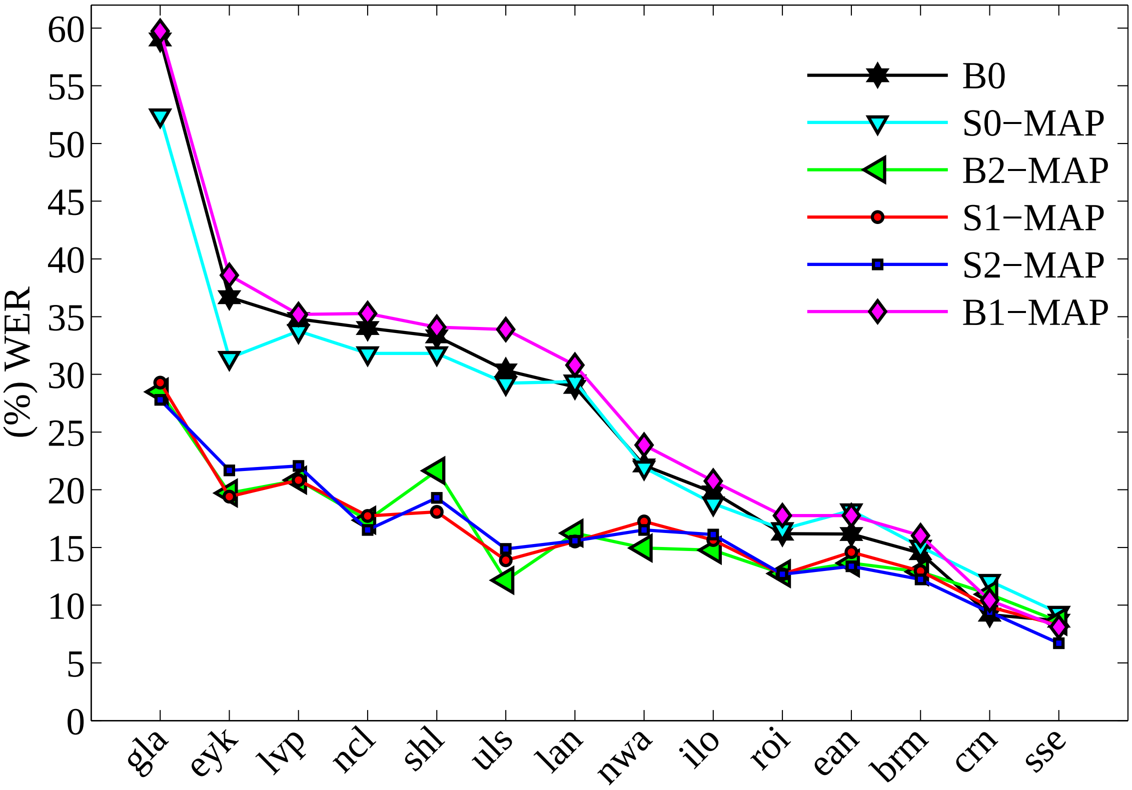

Fig. 7.2 *Comparison of results (supervised MAP adaptation)*

confirms that this accent matches the training set properties more than other northern accents as it achieves a relatively lower error rate compared to them.

The word error rates for the Scottish Highland (shl) and Ulster (uls) accents are grouped with the northern English accents, and are not as poor as one might predict from Figure 6.5.

Adapting to sse accent (B1)

The graph labelled B1 in Figure 7.2 shows the result of MAP adaptation using the sse data. Recall that the purpose of this experiment is to show that subsequent performance gains obtained by adapting to accented data in the ABI-1 corpus result from accent, and not task adaptation. Overall, relative increase in error rate is 4% compared to the baseline. As one would expect, sse performance is almost unchanged. This gives confidence that the improvements reported in this work are indeed due to accent adaptation and not due to adaptation to the ABI-1 task.

True accent model selection (B2)

The results of adapting to the 'true' accent of the speaker is shown in graph labelled B2 in the figure. The relative reduction in WER varies between 51.7% (gla) and 0.7% (sse), with an average reduction of 38.1% across all accents.

Supervised speaker adaptation (S0)

Graph S0 in Figure 7.2 shows results for supervised speaker adaptation of the baseline (B0) with 48s of speech. The reduction in error rate relative to the baseline (B0) for supervised speaker adaptation is 3%. This small reduction in error rate is due to insufficient adaptation material for updating the model parameters using MAP to match the properties of adaptation data. Using a small amount of adaptation data, the model parameters after MAP adaptation will not change much compared to the prior baseline model parameters. Only, for the 'difficult' accents (gla up to shl) some improvement in performance is observed, but for the remainder of the accent regions 'easier' the performance is almost unchanged. For instance, the highest relative %WER reduction of 14.34% and 11.15% is observed for eyk and gla respectively. Surprisingly, for the rest of the accent regions MAP adaptation didn't provide considerable improvement in the performance (less than 6.5% relative WER reduction).

AID based accent model selection (S1)

The result of choosing the accent model returned by ACCDIST-SVM supervised AID, rather than the 'true' accent, is shown in the graph labelled S1 (37.8% relative WER reduction compared to B0). Since the supervised ACCDIST-SVM accent identification has identification accuracy of 95% one would expect the performance in S1 to be similar to B2, and this is the case.

Adaptation using data from *N* closest speakers in ACCDIST-SVM space (S2)

The final graph (S2) is for adaptation using all data from the N closest ABI-1 speakers to the test speaker, according to the correlations between their ACCDIST-SVM supervectors (S2). For MAP adaptation we chose the value of $N = 18$ closest neighbours empirically amongst values of $N = \{3, 9, 12, 18, 36, 54\}$ and achieved WER reduction of 38.18%. The results are similar to B2 (adaptation to the 'true' accent) and slightly more accurate compared to S1 (adaptation to the supervised AID accent).

This is disappointing. By definition, an ABI-1 speaker's 'true' accent is determined by the fact that he or she has lived all of their life in the the location where they were born.

However, for some of the ABI-1 accents there are, subjectively, large differences between speakers, for example due to economic and social factors.

Hence one might expect that using AID to choose an accent-dependent model (S0), would result in lower WER. Further, if a speaker is close to the boundary of an accent region in AID space, one might expect that building a model from the speech of the closest other speakers in AID space would lead to an advantage. However, there is no evidence for this in the current study.

Summary of the results for supervised adaptation using MAP

Table 7.3 summarises the results presented in Figure 7.2 after applying the MAP adaptation to the speaker and accent adaptation tasks. The results in this table are averaged over all accents.

Table 7.3 *Summary of the results after applying supervised adaptation using MAP technique*

Experiment	Code	%WER	%AWR
Baseline	B0	25.97	—
Adapting to sse accent	B1	27.30	-5.13
Correct accent adapt	B2	16.07	38.12
Supervised speaker adapt (48s)	S0	25.13	3.23
Supervised accent adapt (using AID)	S1	16.14	37.85
N closest speaker adapt (N=18)	S2	16.05	38.20

As can be seen from the Table 7.3, the *N* closest adaptation technique, S2, with 16.05% accuracy has the highest accuracy. After that the second and third highest accuracies belong to supervised accent adaptation using true accent label of the test speaker (B2) and the accent label chosen by the AID system using 43.2s of data (S1). Although the *N* closest speaker adaptation approach uses more data for MAP adaptation, the improvement in AID accuracy compared with B2 with 16.07% and S1 with 16.14% accuracy is not considerable. Using a small amount of data for the speaker adaptation experiment using MAP has resulted in poor gain in accuracy for S0 compared to the accent adaptation experiment which use larger amount of data which comes from other speakers with similar accent properties.

7.6.3 Supervised speaker and accent adaptation using MLLR

The detailed results of all of the experiments are shown in Figure 7.3 for MLLR adaptation. In this section we describe the accented speech recognition results after applying MLLR adaptation to the speaker and accent adaptation tasks and analyse the results shown in Figure

7.3, using the knowledge from Figure 6.5 regarding the accent distributions. Please note that for MLLR speaker and accent adaptation tasks we use 20 and 100 regression classes respectively (chosen empirically).

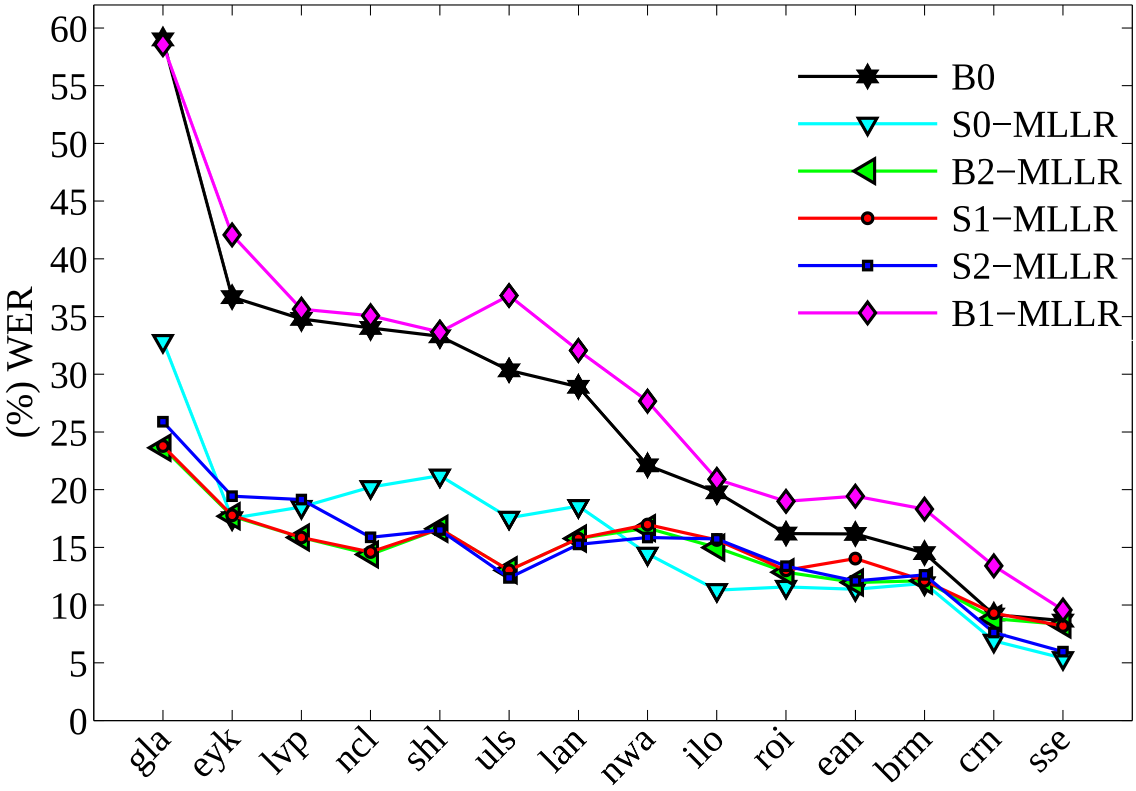

Fig. 7.3 *Comparison of the baseline, supervised speaker adaptation, and supervised accent adaptation results using MLLR*

.

Adapting to sse accent (B1)

The graph labelled B1 in Figure 7.3 shows results of the MLLR adaptation using the sse data. The relative increase in error rate is 9.58% compared to the baseline. Interestingly, the increase in relative %WER compared to baseline B0 is higher for the MLLR adaptation compared to the MAP adaptation.

Recall that the purpose of this experiment is to show that subsequent performance gains obtained by adapting to accented data in the ABI-1 corpus result from accent, and not task adaptation. Overall, performance is 10% poorer than the baseline. As one would expect, sse performance is almost unchanged.

True accent model selection (B2)

The results of adapting to the 'true' accent of the speaker is shown in the graph labelled B2 in the figure. The relative reduction in error rate varies between 59.96% (gla) and 3.84% (sse). Across all accents an average reduction of 44.28% is much higher than that achieved by MAP adaptation (38.1%).

Supervised speaker adaptation (S0)

Graph S0 in Figure 7.3 shows results for supervised speaker adaptation of the baseline (B0) with 48s of speech. Across all accents, the reduction in error rate relative to the baseline (B0) for supervised speaker adaptation using an MLLR system with 20 regression classes is 38.85% which is significantly higher than that achieved by MAP adaptation (3%). This small reduction in error rate is due to lack of enough adaptation material for updating MAP parameters to match the properties of adaptation data. This confirms that even with a small amount of adaptation data from the test speaker, MLLR can achieve a significant gain in accuracy, while this is clearly not the case for MAP.

Improvement in accuracy is observed among all accent regions. For the 'difficult' accents such as eyk and gla speaker adaptation using MLLR provides the highest relative WER reduction of 52.25% and 44.28% respectively, while a smaller relative improvement of 24.65% (crn) and 37.71% (sse) is observed for 'easier' accents.

AID based accent model selection (S1)

During the AID-based accent adaptation, we apply AID to SPA utterances of length 43.2s to identify their regional accent. Based on the accent of the speaker an accented acoustic model (trained off-line) will be selected. The result of applying a supervised AID, to select the accent model rather than using the 'true' accent, is shown in the graph labelled S1 (43.16% relative WER reduction compared to B0). This result is much higher WER reduction (43.16%) than what we achieved using MAP adaptation (37.8%). As the AID system has a high accuracy (95%) the performance in S1 is similar to B2.

Adaptation using data from *N* closest speakers in ACCDIST-SVM space (S2)

The final graph (S2) is for adaptation using all data from the *N* closest ABI-1 speakers to the test speaker, according to the correlations between their ACCDIST-SVM supervectors (S2). For MLLR adaptation we chose the value of $N = 9$ closest neighbours empirically amongst values of $N = \{3, 9, 12, 18, 36, 54\}$ and achieved WER reduction of 42.89% compared to baseline B0. The results are slightly less accurate compared to the B2 (adaptation to the 'true' accent) and S1(adaptation to the supervised AID accent). Again, the WER reduction is not as considerable as we expected according to our hypothesis.

Summary of the results for supervised adaptation using MLLR

Table 7.4 summarises the results presented in Figure 7.3 after applying the MLLR adaptation to the speaker and accent adaptation tasks. Please note that these results are averaged across all of the accents. The baseline experiment (B0), is described earlier in Section 7.6.1.

Table 7.4 *Summary of the results after applying supervised adaptation using the MLLR technique*

Experiment	Code	%WER	%AWR
Baseline	B0	25.97	—
Adapting to sse accent	B1	28.72	-10.50
Correct accent adapt	B2	14.47	44.28
Supervised speaker adapt (48s)	S0	15.88	38.85
Supervised accent adapt (using AID)	S1	14.76	43.16
N closest speaker adapt (N=9)	S2	14.83	42.89

For all the supervised accent and speaker adaptation experiments, applying MLLR adaptation (Table 7.4) rather than MAP (Table 7.3) has given a lower WER. Table 7.4 shows that the supervised accent adaptation approach using the true accent labels of the test speaker (B2) will result in the lower WER (14.47%) than that achieved by speaker adaptation using MLLR. Selecting an acoustic model trained on data with similar accent properties as the test speaker (B2 and S1) rather the acoustic model trained on data from *N* closest speakers to the test speaker in the accent space (S2) will result in a lower WER. The second and third highest accuracy are for the S1 and S2 experiments. similar to the supervised MAP adaptation results in table 7.3 Supervised speaker adaptation result is not as good as that of the supervised accent adaptation (B2 and S1). Using 48s of data from the test speaker, the result achieved by supervised speaker adaptation using MLLR (Table 7.4) is still better than that of the MAP (Table 7.3). In this section we have shown that it is better to use a test speaker's data for AID-based accent adaptation (S1) rather than for speaker adaptation (S0).

Supervised speaker adaptation using more than 48s of data (S0')

In this section we investigate how much data is actually needed for the supervised speaker adaptation to match that of the supervised accent adaptation.

Table 7.5 *Supervised speaker adaptation results using up to 221s of data versus accent adaptation results using only 43s of the speaker’s data*

Experiment		B0	S0	S0’			S1
Length of data from test speaker		—	48s	102s	136s	221s	43.2s
MAP	%WER	25.97	25.13	21.40	20.11	15.57	16.14
	%AWR	—	3.23	17.60	22.56	40.04	37.85
MLLR	%WER	25.97	15.88	14.17	13.81	12.30	14.76
	%AWR	—	38.85	45.44	46.82	52.63	43.16

Table 7.5 compares the performance of supervised AID based accent adaptation using 43.2s of speech (S1) to select a suitable acoustic model and speaker adaptation using more than 48s and up to 221s of data (S0’) to adapt to the test speaker’s speech, using MAP and MLLR. Comparing the S0’ and S1 results, we can conclude that the data required to achieve a similar result to AID based adaptation with speaker adaptation is greater by a factor of 3.53 and 1.1 for MAP and MLLR respectively.

This result has a valuable impact for real time systems where there is no prior data available from the test speaker, we can achieve similar or even better results by identifying an acoustic model that matches his or her accent. To further reduce the online computational cost and speed up the process, we could apply previously trained accent-specific acoustic models, so that given an unknown test speaker the AID system can recognise his or her accent and choose a model that roughly matches his or her accent properties.

7.7 Summary and conclusions

In this chapter it is shown that, using an ASR system trained on the WSJCAM0 corpus of British English speech, error rates can be up to seven times higher for accented speech (e.g. Glaswegian accent (gla)) than for standard southern English (sse).

Given an average of 43.2s of data from a new speaker, three alternative approaches to supervised accent-dependent modelling were investigated, namely using the acoustic model for the ‘true’ accent, using the acoustic model for the accent chosen by a supervised AID system, and building a model using data from the *N* closest speakers in the supervised ‘AID feature space’. Table 7.6 summarises the results presented in Figures 7.3 and 7.2, and reports the results of MLLR and MAP adaptation applied to the speaker and accent adaptation tasks. Despite the fact that there exist some outliers in each accent, and choosing the data from *N* closest speakers might seem like a good approach to accommodate such outliers with more personalised models, the results showed that there is no particular advantage in adopting this technique using the MLLR approach. Using MAP adaptation for adapting to the *N* closest speakers in the accent space provided a slightly better result compared to the AID based or correct accent adaptation approaches.

These results show that all three accent adaptation methods give significantly better than the performance obtained with the baseline, accent-independent model. Compared with the baseline WSJCAM0 system, the relative reduction in ASR error rate is 44.28% and 38.12% for accent-dependent models (with true accent labels) using MLLR and MAP respectively. We also demonstrated that using the 43.2s of speech to identify an appropriate accent-dependent model using AID gives lower WER than speaker adaptation, which has high commercial value when we don't have enough material from the user for speaker adaptation. In such cases we can simply use an AID system to select the relevant accented acoustic model (trained offline) that matches the user's regional accent.

From the results it is clear from these results that MLLR gives lower WER than MAP on this application. Hence in future experiments adaptation will be performed using MLLR. In most practical applications, unsupervised adaptation approaches are preferred over supervised ones. In the next chapter we will investigate the performance of a range of unsupervised acoustic model adaptation techniques.

Table 7.6 *Summary of the baseline, supervised speaker adaptation and supervised accent adaptation results using MAP and MLLR techniques*

	Experiment	Code	%WER	%AWR
	Baseline	B0	25.97	—
MAP	Adapting to sse accent	B1	27.30	-5.13
	Correct accent adapt	B2	16.07	38.12
	Supervised speaker adapt (48s)	S0	25.13	3.23
	Supervised accent adapt (using AID)	S1	16.14	37.85
	N closest speaker adapt (N=18)	S2	16.05	38.20
MLLR	Adapting to sse accent	B1	28.72	-10.50
	Correct accent adapt	B2	14.47	44.28
	Supervised speaker adapt (48s)	S0	15.88	38.85
	Supervised accent adapt (using AID)	S1	14.76	43.16
	N closest speaker adapt (N=9)	S2	14.83	42.89

Chapter 8

Unsupervised model selection for accent-robust ASR (GMM-HMM)

8.1 Introduction

In the previous chapter we investigated three alternative supervised techniques for accent-dependent acoustic model selection, namely using the 'true' accent model, choosing the model using an AID system, and building a model using data from the *N* closest speakers in the 'AID feature space'.

We showed that using 43.2s of speech to identify an appropriate accent-dependent model outperforms supervised speaker adaptation using MLLR and MAP. The supervised speaker adaptation requires between 1.1 and 5.1 times more data to achieve a similar result to accent adaptation using MLLR and MAP respectively. In all our experiments, MLLR adaptation achieved a higher accuracy compared with MAP. Hence, the experiments in this chapter are only carried out using MLLR.

This chapter investigates text-independent (unsupervised) techniques to compensate for the effects of regional accents of British English on ASR. A set of experiments are proposed to answer the following questions.

- Given a small amount of speech from a speaker, is it better to use this speech for unsupervised AID, so that an accent-dependent model can be selected, or directly for the speaker adaptation? or the combination of both?

- How will the accuracy of the unsupervised AID affect the accuracy of ASR after accent-dependent model selection? Is it better to use a computationally expensive but more accurate AID system (Demarco's i-vector, or phonotactic systems)? or will a simpler AID which is not as accurate perform equally well (our proposed i-vector system)?

- How much speech from an individual is needed to achieve WER from unsupervised speaker adaptation that are comparable with the use of an accent-dependent model selected using unsupervised AID?

8.2 Experiments on unsupervised acoustic model adaptation

The description of our baseline British English speech recognizer, feature extraction, language model, and speech corpora are available in Section 7.2.

- **Unsupervised speaker adaptation:** For each speaker we conducted unsupervised (transcription derived from the baseline WSJCAM0 ASR) MLLR speaker adaptation with 48.1s (SPB), 101.5s (SPB+SPC), 136s (SPB+SPC+'Shortphrases') and 221s (SPB+SPC+ 'Shortphrases'+'Shortsentences') of speaker-dependent data.

- **Unsupervised accent adaptation:** The unsupervised accent adaptation is in fact an acoustic model selection technique based on the knowledge of the speaker's accent produced by an unsupervised (text-independent) accent identification system. For each subject in the ABI-1 corpus, the SPA recording was used as test data, and a gender- and accent-dependent model was created by applying supervised MLLR accent adaptation to the baseline WSJCAM0 system. Adaptation used the SPB, SPC, 'shortsentences' and 'shortphrases' data from 9 other subjects with the same gender and accent as the test speaker (approximately 31.5 minutes of speech).

Here, we applied our phonotactic AID system (described in Section 6.2.1), our i-vector AID system (described in Section 6.2.2), and Demarco's i-vector AID system (described in Section 6.2.2) as our unsupervised AID systems (described in Section 6.3).

We introduce a set of experiments to compare the results of applying unsupervised speaker adaptation and unsupervised accent adaptation techniques. We explore the effectiveness of each approach on the accented speech recognition task.

For accent-dependent modelling, we investigate the effectiveness of phonotactic, Demarco's i-vector, and our i-vector decisions on acoustic model selection. These three systems are all unsupervised (text independent) and will be compared against adaptation to acoustic model selected based on the 'true' accent of the speaker. The purpose is to investigate the significance of the accuracy of the AID systems (used for acoustic model selection) on the ASR recognition accuracy.

In addition to that, we investigate whether choosing the *N* closest speakers from the unsupervised phontactic AID accent space rather than the supervised ACCDIST-SVM AID accent space, will improve the ASR accuracy compared to selecting one acoustic model based on the speaker's identified accent.

In the following experiments we try to address these questions with insight from our phonotactic visualisation results which illustrates the distributions of accents in the accent feature space. The ASR experiment results will be reported in terms of %WER or the Average Relative WER Rate Reduction (%AWR). The baseline experiment (B0), the experiment for adapting to sse accent (B1), and the 'true' accent model selection (B2) experiment are described earlier in Sections 7.6.1 and 7.6.3.

Please note that the majority of code names in this chapter contain letter U which stands for Unsupervised. The term 'unsupervised' can be used in the context of either the unsupervised speaker adaptation or the unsupervised accent adaptation.

Unsupervised speaker adaptation (U0)

To measure the improvement in ASR WER by applying unsupervised speaker adaptation to the baseline, we conducted speaker adaptation experiments for each ABI-1 subject using unsupervised (U0) MLLR adaptation. Later, we compare this improvement with what is achieved using the accent adaptation.

Accent-dependent models chosen using Demarco's i-vector AID system (U1)

In these experiments, for each subject, speech recognition is performed using the accent-adapted model corresponding to the result of AID for that speaker, using Demarco's i-vector AID (U1). Demarco's i-vector system, with an accuracy of 81.05%, is described in Section 6.3.

Accent-dependent models chosen using phonotactic AID (U2)

In these experiments, for each subject, speech recognition is performed using the accent-adapted model corresponding to the result of AID for that speaker, using phonotactic based AID (U2). Our phonotactic AID system, with accuracy of 80.65%, is fully described in Section 6.2.1.

The purpose of this experiment is to investigate the ASR recognition accuracy by using a phonotactic AID system rather than an acoustic AID system (Demarco's AID). These two

AID systems rely on different features to recognise the speaker's accent and despite their similar average accuracies, for a number of speakers their accent identification result will be different.

Accent-dependent models chosen using our i-vector AID (U3)

In these experiments, for each subject speech recognition is performed using the accent-adapted model corresponding to the result of AID for that speaker, using our i-vector AID (U3). Our i-vector system, with accuracy of 76.76%, is fully described in Section 6.2.2.

The idea behind this experiment is to measure the changes in ASR accuracy by choosing an AID system with lower accuracy for the accent-specific acoustic model selection task rather than the more accurate AID systems which are often computationally expensive or supervised (text-dependent). This experiment is of high commercial use for developers who are interested in a faster and computationally efficient AID system for acoustic model selection.

ASR Model based on N closest speakers in phonotactic AID space (U4)

In U4, each ABI-1 speaker s, is represented as a phonotactic supervector V_s. Given a test speaker s and speaker t the correlation distance $C(V_s,V_t)$ is calculated between V_s and V_t, and the N speakers for which the correlation distance $C(V_s,V_t)$ is largest are identified. A new model is then constructed by adapting the baseline WSJCAM0 model using the adaptation data from these N speakers. The values of $N = 9, 18, 36, 54$ where tested and $N = 18$ (approximately 66.2 minutes of adaptation speech from other speakers) was chosen after 2-fold cross-validation.

The hypothesis is that an acoustic model created based on data from neighbouring speakers in the accent space can better match speakers who are on the boundary between different accent regions or might exhibit a mixture of accent properties.

The result of adaptation to the N closest speakers in the phonotactic accent space will be compared against that achieved by the ACCDIST-SVM system.

Accent-dependent models chosen based on 'true' accent followed by unsupervised speaker adaptation (BU)

For each speaker, model selection is applied, based on the 'true' accent (BU) of the speaker, to obtain an 'accent adapted' acoustic model. Unsupervised MLLR speaker adaptation is then applied to that model and recognition is performed.

The aim is to compare the results achieved by this approach with that of unsupervised model selection and unsupervised speaker adaptation.

Accent-dependent models chosen using phonotactic AID followed by unsupervised speaker adaptation (UU)

For each speaker, a model selection is applied, based on the accent-determined by phonotactic AID system (UU), to obtain an 'accent adapted' acoustic model. Unsupervised MLLR speaker adaptation is then applied to that model and recognition is performed.

This system is expected to exploit the speech from the test speaker for both selection of the accent-dependent acoustic model and unsupervised speaker adaptation. We expect this system to work much better than a system that only relies on either speaker adaptation or accent adaptation.

8.3 Unsupervised AID and visualisation of the accent space (phonotactic system)

The phonotactic AID system is fully described in Section 6.2.1. This unsupervised AID system exploits the phonotactic supervectors containing the 4-gram language model frequencies to identify different British English accents. The phonotactic supervectors are projected to a two dimensional space, and can be seen in Figure 6.3. In this chapter we use the intuition from the phonotactic accent space visualisation map to analyse the ASR results. A list of different accent regions and their corresponding broad accent groups can be found in Table 6.10.

8.4 Results and discussion

In this section we apply various unsupervised techniques in an attempt to improve the accuracy of accented speech recognition. We report and analyse the results of unsupervised speaker and accent adaptation and the combination of both using MLLR. A detailed breakdown of results of individual systems is shown in Figure 8.1 for different British regional accents. The accents are ordered on the horizontal axis according to the baseline B0 results.

Table 8.1 summarises the results presented in Figure 8.1 after applying MLLR adaptation to the speaker and accent adaptation tasks. Experiments B0, B1, and B2 are fully described in Section 7.6.3. The remainder of the experiments will be discussed in this section.

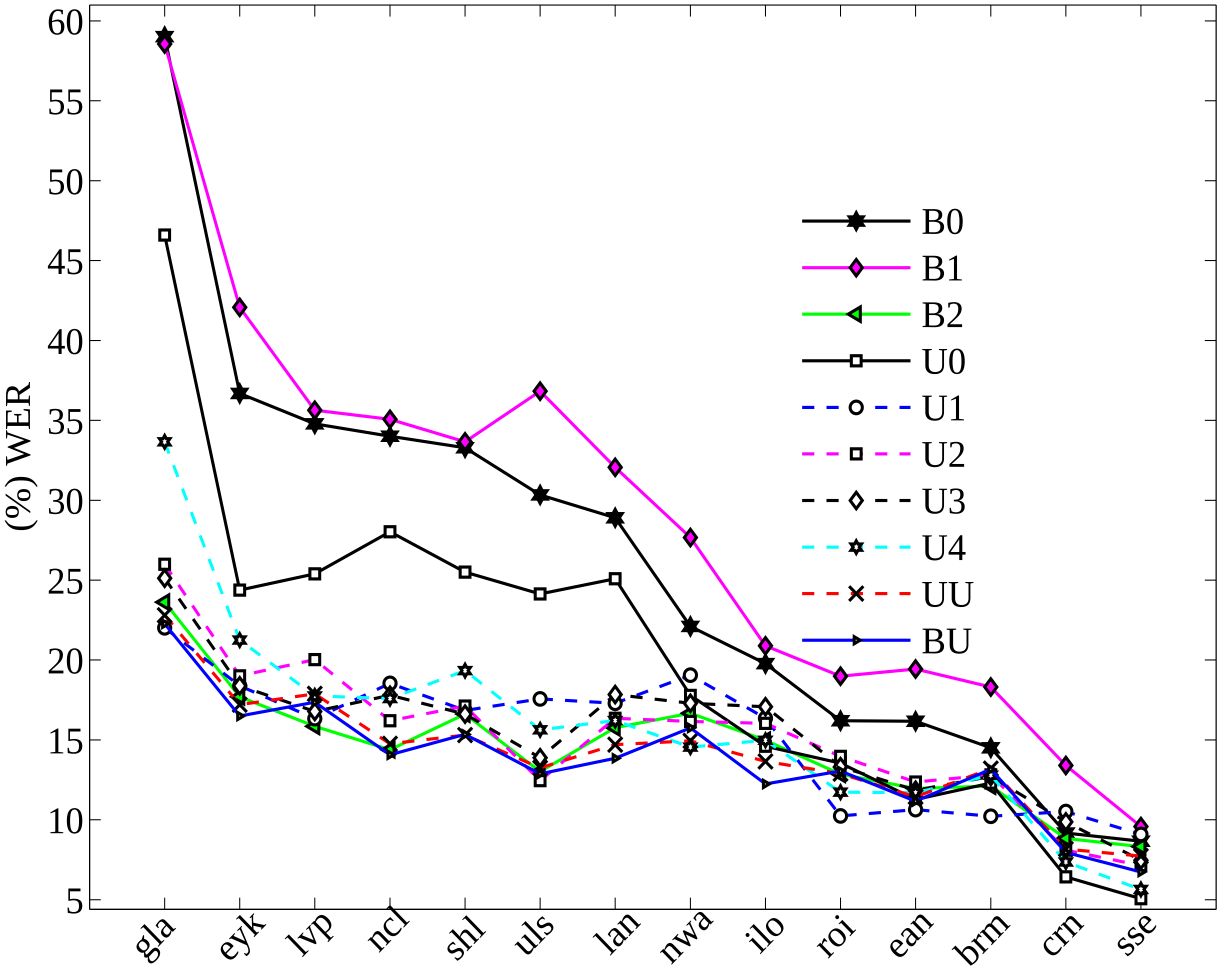

Fig. 8.1 *Comparison of the baseline, unsupervised speaker adaptation, and accent adaptation results*

Unsupervised speaker adaptation (U0)

Graph U0 in Figure 8.1 shows results for unsupervised speaker adaptation of the baseline (B0) with 43.2s of speech. The relative reduction in error rate relative to the baseline (B0) for unsupervised speaker adaptation is 21.83%. It can be noted that there is an improvement in ASR performance across all accents when compared to the baseline (B0). However the relative improvement seems to be greater for the 'easier' accents (sse up to lan), and gets narrower after that. Presumably, this is due to the poor baseline ASR performance on the more 'difficult' accents which results in an incorrect transcription.

Table 8.1 *Summary of the results for proposed unsupervised adaptation approaches*

Experiment	Code	%WER	%AWR
Baseline	B0	25.97	—
Adapting to sse accent	B1	28.72	-10.50
Correct accent adapt	B2	14.47	44.28
Unsupervised speaker adapt (48s)	U0	20.30	21.83
Unsupervised accent adapt (using Demarco's i-vector)	U1	15.22	41.39
Unsupervised accent adapt (using phonotactic AID)	U2	15.27	41.22
Unsupervised accent adapt (using our i-vector)	U3	15.43	40.59
N closest speaker adapt (N=18, phonotactic AID space)	U4	15.73	39.45
Unsupervised accent and speaker adapt (using phonotactic AID)	UU	14.14	45.57
adapt to correct accent label and unsupervised speaker adapt	BU	13.74	47.10

Unsupervised accent adaptation using Demarco's i-vector (U1) AID and proposed phonotactic AID (U2)

Our proposed phonotactic AID (with 80.65% accuracy) and Demarco's i-vector (with 81.05% accuracy) are described in Sections 6.2.1 and 6.3 respectively. The results of choosing the accent model returned by AID, rather than the correct accent, are shown in the graphs labelled U1 (Demarco's i-vector AID) and U2 (phonotactic AID) and summarised in 8.1. Surprisingly, although the AID error rates for both AID systems are quite high, the corresponding ASR %WERs is close to those obtained with the 'correct' accent model (B2). Specifically, the WER obtained with the 'true' accent model is 14.47%, compared with 15.22% and 15.27% for model selection using Demarco's i-vector (U1) and the phonotactic based AID (U2) systems, respectively. Despite the fact that phonotactic and i-vector systems rely on different properties for identifying speakers accent and give different labels to a number of speakers, the WERs that result from using these methods for model selection are very similar (approximately 15%).

Unsupervised accent adaptation using our i-vector AID system (U3)

Our proposed i-vector AID (with 76.76% accuracy) is described in Section 6.2.2

In Figure 8.1 graph U3 shows the results for unsupervised accent adaptation of the baseline (B0) using our simple i-vector AID system that is much simpler compared to those in U1 and U2. In the U3 system, the average word error rate reduction compared to the baseline B0 is 40.59% which is only 1.77% and 1.94% less than that achieved by the more complicated phonotactic system and Demarco's ivector systems.

The phonotactic AID result is achieved by fusion of 15 PPRLM systems while Demarco's system is obtained by fusing 630 i-vector systems. Our i-vector approach is much simpler and consists of only one system, which makes it computationally more efficient and faster.

Adaptation using data from *N* closest speakers in phonotactic space (U4)

The results of adapting the baseline (B0) to the data from the $N = 18$ closest speakers to the test speaker in the phonotactic feature space is shown in the graph labelled U4.

Using this approach we attempted to create models that more accurately describe the test speaker's accent property when he or she exhibit features that might not be consistent with his or her 'true' accent.

However, similarly to the *N* closest speakers approach in the ACCDIST-SVM system, there is no evidence of improvement from adaptation using data from the *N* closest speakers compared to accent-dependent model selection using an AID.

Figure 8.1 shows that the relative %WER reduction is higher for the 'easier' accents (from sse up to lan) compared to U2, but gets lower after that. This shows that for easier accents its possible to gain more advantage by adapting to the data from *N*-closest speakers rather than using accented acoustic models chosen using AID. For the *N* closest adaptation in phonotactic space we chose the value of $N = 18$ closest neighbours empirically amongst values of $N = \{3, 9, 12, 18, 36, 54\}$.

Comparing the result of adaptation to the *N* closest speakers in the ACCDIST-SVM accent space (with 14.83 %WER) and phonotactic accent space (with 15.73% WER) shows that the former has achieved a higher accuracy. This might be due to the fact that the ACCDIST method is supervised and relies on more robust measures to find neighbouring speakers in the AID space (forced alignment of the vowel triphone's distance tables) rather than the phone frequency measure used in phonotactic system.

Accent adaptation followed by unsupervised speaker adaptation (BU and UU)

As can be seen from the Figure 8.1, for the difficult accents which are typically far from sse in the visualisation map, the baseline B0 performs poorly, and applying an unsupervised speaker adaptation U0 on top of this poor baseline did not provide much improvement over the baseline.

In order to take the most advantage from applying the unsupervised speaker adaptation, in this experiment we applied unsupervised accent adaptation prior to that. This way, by adapting the baseline B0 to the accent specific acoustic model, we make sure that our unsupervised speaker adaptation is carried out on a much better baseline. This is of high importance for more difficult accents such as gla where the baseline performs poorly.

The unsupervised accent adaptation part of these experiments can be carried by choosing the accent model using either the phonotactic AID or the correct accent label of the test speaker.

In both cases when the systems are followed by speaker adaptation, as the accent moves further away from sse the relative improvement caused by accent adaptation followed by speaker adaptation increases, BU and UU, compared to the unsupervised speaker adaptation, U0.

In addition to that, considerable %WER reduction of 7.4% (UU) and 5.04% (BU) occurs compared to that of the only accent adapted systems, U2 and B2 respectively.

The top two highest ASR accuracies among all the unsupervised accent and speaker adaptation experiments proposed in this section, belong to BU and UU respectively. Where, the former and the latter systems have reduced the average %WER by up to 47.10% and 45.57% respectively. It is also worth nothing that using accent labels to choose the accent models in BU system rather than the AID in UU system has relatively reduced the %WER by 2.83%.

Unsupervised speaker adaptation using more than 48s of data (U0’)

In this section we investigate how much data is actually needed for the unsupervised speaker adaptation to match that of the unsupervised accent adaptation.

We assert that the comparisons made in these experiments are ‘fair’. Although the underlying accent-dependent acoustic models (trained offline) have certainly been trained with far more data than that used for speaker adaptation, the key *practical* question is whether a good accent-dependent model can be reliably selected using the same small amount of data that is available for speaker adaptation. We are evaluating the consequences of using this small amount of data in different ways.

Regarding speaker adaptation, Table 8.2 compares the performance of AID based unsupervised model selection using 43.2s of speech and unsupervised speaker adaptation using more than 48s and up to 221s of speech (U0’). The unsupervised model selection approaches are presented for our most accurate and least accurate AID systems, namely Demarco’s i-vector AID system and our i-vector based AID system with accuracies of 81.05% and 76.76% for experiments U1 and U3 respectively.

The performance obtained with unsupervised speaker adaptation using additional data is never as good as unsupervised model selection. Even when the unsupervised speaker adaptation (U0’) is given five times more adaptation data from the test speaker, and we

Table 8.2 *aSR results for unsupervised speaker adaptation versus unsupervised accent adaptation*

Experiment		B0	U0	U0’			U1	U3
Length of data from test speaker		—	48.0s	102s	136s	221s	43.2s	43.2s
MLLR	%WER	25.97	20.30	18.75	18.99	17.83	15.22	15.43
	%AWR	—	21.83	27.80	26.87	31.34	41.39	40.59

use our simplest AID system (U3), unsupervised speaker adaptation cannot achieve similar results.

8.5 Summary and conclusions

This chapter investigates whether the notion of ‘regional accent’ can be used explicitly to improve ASR performance using a series of unsupervised adaptation approaches. Table 8.1 summarises the results presented for unsupervised speaker and accent adaptation in this section.

Given an average of 43.2s of data from a new speaker, three alternative approaches to unsupervised acoustic model adaptation are investigated:

- Using the acoustic model for the accent chosen by one of the three different unsupervised AID systems, namely Demarco’s i-vector based AID system, phonotactic based AID system, and our i-vector based AID system. These systems rely on different features for accent identification and have different level of complexity and accuracy. We investigated whether any of these factors will affect the ASR performance.
- Using the data from the *N* closest speakers in the phonotactic accent space for adaptation
- Using models chosen according to the speaker’s accent followed by speaker adaptation

The results shows that all unsupervised accent adaptation approaches significantly outperform the baseline. The relative reduction in ASR error rate compared with the baseline WSJCAM0 system is 47.10% for models chosen according to the ‘correct’ accents followed by speaker adaptation, and between 41.39% and 40.59% for ASR models selected using the unsupervised AID systems. We observed that, despite our hypothesis that choosing the data from *N* closest ABI-1 speakers to the test speaker will not improve the performance compared to accented acoustic model selection performed by AID.

The performance obtained with different AID systems for model selection does not appear to be very sensitive to the accuracy of different AID systems, similar %WER is obtained using different AID systems with error rates ranging from 23.24% to 18.30%. Given the apparent insensitivity of ASR performance to AID error rate, it will be interesting to apply the same methods to shorter enrolment utterances, since it would be desirable for an ASR system not to require around 40 seconds of data for reliable model selection. Choosing the accent model using either the phonotactic AID or the correct accent label of the test speaker followed by the unsupervised speaker adaptation gives us up to 45.57% and 47.10% relative WER reductions compared to the baseline, which are the highest ASR accuracies compared to that of all the unsupervised accent and speaker adaptation experiments proposed in this section. The relative improvements in the former and latter system are similar to that of the supervised speaker adaptation (reported in Table 7.5) using twice and three times adaptation data from the test speaker (using MLLR).

Results show that using a total of 48s of data for 'true' accent adaptation followed by unsupervised speaker adaptation (UU)) will outperform the supervised speaker adaptation (S0) with approximately five times more adaptation data from the test speaker.

Chapter 9

Speech recognition performance on various multi-accented data sets (DNN-HMMs)

9.1 Introduction

During the period of this research, the field of ASR has become dominated by the use of DNN-HMM systems for acoustic modelling. As explained in Chapters 3 and 4, the power of the DNN-HMM approach stems from the ability to compute discriminative state-observation probabilities using a DNN [7]. DNN-HMM systems have been shown to be superior to GMM-HMM systems in their ability to accommodate variability in speech signals. Therefore, it is natural to ask how DNN-HMM systems can accommodate regional accents. The purpose of this chapter is to explore this question.

In Chapters 7 and 8, we showed the effect of accented speech on automatic speech recognition accuracy, and we conducted a study to address the problems caused by regional accents by using accent identification and selecting an accent-dependent acoustic model.

In this chapter, we start by comparing the performance of our baseline GMM-HMM and DNN-HMM systems for recognizing accented speech, then we answer a number of questions.

- Is it better to use a GMM-HMM system to recognise accented speech and apply accent plus speaker adaptation to it or it is better to use a DNN-HMM system and rely on its ability to accommodate the variability in multi-accent data?

- If DNN-HMM systems provided superior results on accented speech compared to GMM-HMMs, is it better to train a generalized DNN using a large amount of data from diverse accents, or is it better to use accent information from a test speaker and use a smaller but accent-specific training data set?

- In case the generalized DNN outperforms the system trained on smaller accent-specific data, which accent groups should be included?

- Is it possible to achieve further improvement in the DNN-HMM system by incorporating a large amount of accent-diverse data in the pre-training phase?

- Is adding data with wide accent diversity in the pre-training stage as crucial as in the fine-tuning stage?

9.2 DNN performance on various multi-accented data sets

We used the Kaldi open source toolkit [162, 163] to build a DNN-HMM based speech recognizer. This section describes the baseline DNN-HMM system. Then it explores the changes in the ASR accuracy after supplementing the training set with different types of Extra Training Material (ETM). Finally, it investigates the changes in ASR performance after supplementing the pre-training dataset with different types of Extra Pre-training Material (EPM).

Our ETM can be added in two different ways: it can be added to match the accent characteristics of the test speaker or it can be added regardless of the regional accent of the test speaker. Experiments in this section can be categorised in one of the following three categories, namely (1) adding no extra ETM (baseline system), (2) adding accent-dependent ETM, (3) adding accent-independent ETM, and (4) adding accent-independent EPM. By accent-independent we mean the accent of the additional ETM or EPM will not be chosen based on the knowledge of the test speaker's accent and it remains the same for all test speakers. These experiments are then categorised into different sub-categories based on the variation in their extra training material in terms of the size, the accent diversity, and the type of accent or broad accent. For each experiment we describe the 'accent diversity' in terms of the number of regional accents it contains in its data set.

The ETM and EPM are added from either the ABI-1 corpus or WSJCAM0 development set (WSJD). Since our EPM does not need to be transcribed, in one of the experiments we used the material from the ABI-2 corpus which contains 13 additional accents compared to ABI-1 (Section 5.1.1).

9.2.1 Description of code names chosen for each experiment

In this section, a specific code name is allocated to each experiment. For instance, the code corresponding to all the experiments reported in this chapter starts with D, which refers to DNN. The second letter in the experiment code name can be 0, S, and U which corresponds to the Baseline, Supervised, and Unsupervised experiments. In the supervised experiments the test speaker's accent is known in advance. Thus, this information is used for adding the additional training material from the same accent as the test speaker. These experiments are called accent-dependent experiments. On the other hand, in the unsupervised experiments the test speaker's accent is considered unknown, and the accent property of the extra training material is independent of the test speaker's accent. These experiments are called accent-independent experiments. The third character of the code can be a number for the experiments carried out at the fine-tuning level, or can be letter P for the experiments carried out at the Pre-training level.

9.2.2 Baseline DNN-HMM system, no ETM added (D0)

The baseline DNN-HMM ASR system is trained on the WSJCAM0 training set (WSJT), and the test material comes from the ABI-1 corpus. This is a baseline to compare the results of other experiments when ETM is added to the WSJCAM0 training set.

Initially, the GMM-HMM system is trained on 39 dimensional mean and variance normalized MFCCs plus delta and acceleration, with 6 GMMs per state. Then hidden layers of our DNN-HMM system are initialized with stacked RBMs, that are trained using the CD algorithm one layer at a time as described in Section 4.5.

After pre-training using 5 epochs, our system is fine-tuned using mini-batch SGD to minimize per-frame cross-entropy between the HMM state target posterior probabilities and network output. Here, the learning rate is 0.008, the mini-batch size is 256. In this stage the GMM-HMM system provides the triphone tree and the alignment of context dependent states to frames to be able to a train the DNN to classify frames into HMM state PDFs. The DNN fine-tuning and pre-training has been described with full details in Chapter 4.5, and the hybrid DNN-HMM approach to ASR is presented Section 3.10.

Our DNN system is trained with 13 dimensional mean/variance normalized filterbank features, spliced using context of +/- 7 frame next to the central frames, it has 5 hidden layers, with 1024 neurons with sigmoid activation, 195 inputs and 1619 dimensional output layer with softmax activation.

We used the BEEP dictionary [160], extended to include all of the words in the ABI-1 corpus (Section 7.2). Our language model is obtained from a weighted combination of the 5k WSJ0 and ABI-1 corpus bigram language models (Section 7.2).

9.2.3 Adding accent-dependent ETM to the training set

In this section we describe experiments that investigate the effect of adding additional material to the WSJCAM0 training set. This additional material is either from the same accent as the test speaker, or from a broad accent group that includes the test speaker's accent. The experiments are as follows:

ETM with fixed size and accent diversity, using a single accent (DS1)

In this experiment, the additional training material is taken from the same accent as the 'true accent' of the test speaker. Thus, if the true accent of the test speaker is gla, then the additional training material will be taken from the gla set, excluding the test speaker. This scheme is illustrated in Figure 9.1. In order that the results are all comparable, the amount of additional training material is 40 minutes in all cases.

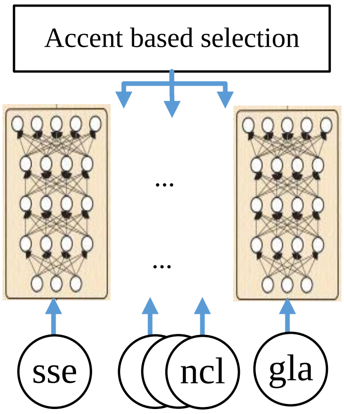

Fig. 9.1 *Accent-specific training using data from 14 accent regions (40 minutes from each region) (DS1)*

ETM with fixed size and accent diversity, using a broad accent group (DS2)

In this experiment, the additional training material is taken from the different accents that belong to the same broad accent group as the test speaker's true accent. Thus, for example, if the speaker's true accent is eyk, then the extra training material will be chosen from all of the northern English accents, because this is the broad accent group that includes eyk. In this experiment 2.25 hours of data is used for each broad accent group. This training scheme is represented in Figure 9.2.

Results of this experiment will show whether it is better to use additional material from the test speaker's individual accent, or from the broad accent group to which the test speaker's accent belongs.

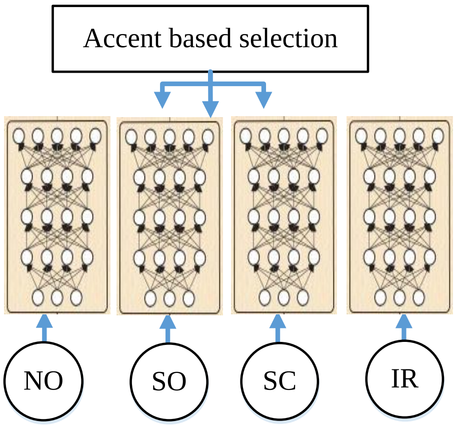

Fig. 9.2 *Accent-specific training using data from four groups of broad accents (2.25 hours from each group) (DS2)*

9.2.4 Adding accent-independent ETM to the training set

Accent independent ETM refers to the situation where the accent content of the ETM does not depend on the accent of the current test speaker. The purpose of the experiments is to determine whether it is better to use ETM from all accents from specific 'key' accents. We refer to this 'dimension' as 'accent diversity'. Maximum diversity is achieved when the ETM contains material from all 14 of the ABI-1 accents, and minimum diversity is when the ETM contains data from just one accent (which does not depend on the accent of the test speaker). Of course, as the diversity of the ETM increases, the amount of available training material also increases. Hence an additional dimension that needs to be explored is the size of the ETM. The details of the different experiments are as follows.

ETM with varying size and fixed accent diversity (DU1)

In this experiment the accent diversity remains fixed (14 accents) and only the amount of data in the ETM varies.

Specifically, we compare the results of training our system on (a) 2.25 hours of data spanning all 14 accents in the ABI-1 corpus, and (b) 8.96 hours of data spanning all 14 accents in the ABI-1 corpus. This training scheme is represented in Figure 9.3, and the results are reported in Section 9.4.

The aim of this experiment is to investigate whether it is better to train a generalized DNN using data from all 14 accents (maximum diversity, DU1) or an accent-specific system (minimum diversity, DS1, DS2). Then we analyse whether improvements in our results are due to the amount (DU1) or the accent diversity (DU2) of the extra training data added to the system.

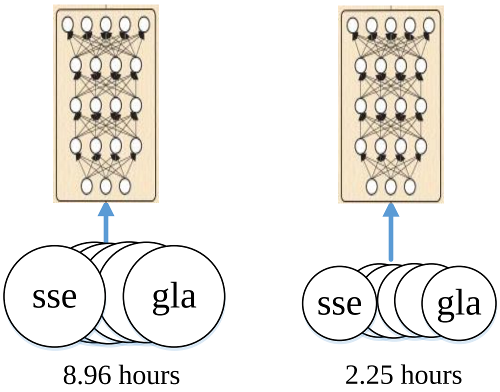

Fig. 9.3 *Varying set size, fixed training accent diversity (DU1)*

ETM with fixed size and varying accent diversity (DU2)

In this experiment, we analysed the effect of increasing accent diversity in the training set while keeping the amount unchanged (2.5 hours).

Specifically, in these experiments WSJT is supplemented by 2.25 hours of ETM data from the following sources.

(a) In this case only WSJD was used (minimum accent diversity).

(b) 2 accents from 1 broad accent group: Specifically in this case the accent group is northern English (NO), and the specific northern accents that were used are eyk and nwa.

(c) 4 accents from 3 broad accent groups: Specifically in this case the NO (including only eyk, nwa), SC (including only shl) and IR (including only roi) accent groups were used.

(d) 8 accents from 4 broad accents groups: Specifically in this case the NO (including only eyk, nwa), SC (including only gla, shl), IR (including only roi, uls) and SO (including only brm, ilo) accent groups were used.

(e) In this case the data from all 14 accents in the ABI-1 corpus is used (maximum accent diversity)

Table 9.1 summarises these experiments, and Figure 9.5 is a schematic representation of the different training schemes.

Table 9.1 *Increasing accent diversity*

Length	2.25 hours		
Code	ETM source	accents	Broad accents
(a)	WSJD	-	-
(b)	ABI-1	2	NO
(c)	ABI-1	4	NO, SC and IR
(d)	ABI-1	8	NO, SC, IR and SO
(e)	ABI-1	14	NO, SC, IR and SO

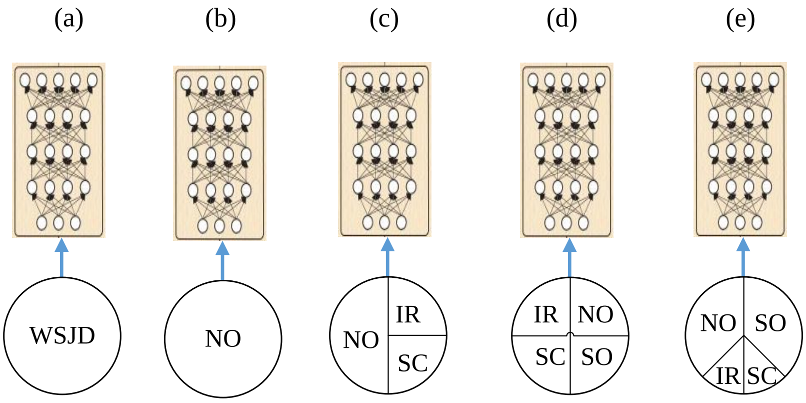

Fig. 9.4 *Varying training accent diversity, fixed set size (DU2)*

ETM from different broad accent groups, fixed size and accent diversity (DU3)

In this experiment the size of the ETM is fixed and the diversity of the ETM is also fixed, in the sense that the ETM covers only data from a single broad accent group (SO, NO, SC, IR).

Supplementing WSJT with data from any accent group exposes the training data to additional diversity and it changes the recognition performance. These experiments compare the performances across all accents on systems trained on the WSJT plus (a) 2.25 hours of speech from the WSJD, and 2.25 hours of data from accent groups (b) SC, (c) IR, (d) NO and (e) SO (as shown in Figure 9.5).

The purpose of the experiment is to determine whether the choice of the broad accent group from which the ETM is taken affects ASR performance. Results of this experiment are reported in Section 9.4.

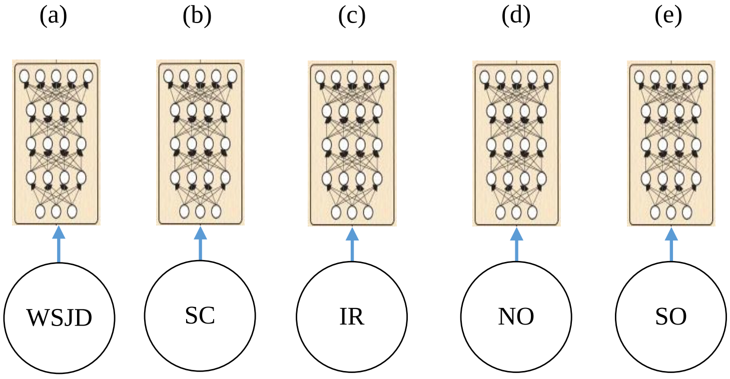

Fig. 9.5 *Varying training broad accent group, fixed set size (DU3)*

9.2.5 Increasing EPM size and accent diversity (DUP)

Accent independent EPM refers to the situation where the accent content of the EPM does not depend on the current test speaker.

One of the claims made in favour of HMM-DNN systems is that they can capture useful properties of the structure of speech in the training set during the pre-training phase. The aim of the experiment is to analyse the effect of increasing size and accent diversity in the pre-training data on the accuracy of the system. In this experiment we explore the effect of incorporating data with different degrees of accent diversity in the pre-training phase.

DNNs were pre-trained using:

(a) In this case the WSJT is used. The accent diversity is minimum.

(b) In this case 8.96 hours of data from ABI-1 is used. The accent diversity is 14.

(c) In this case 17.79 hours of data from ABI-2 is used. The accent diversity is 14.

(d) In this case 26.7 hours of data from ABI-1&2 is used. The accent diversity is 27.

In all the experiments in this section, WSJT is used for fine-tuning the system, and only the content used for the pre-training varies.

9.3 Unsupervised AID and visualisation of the accent space using i-vector AID system

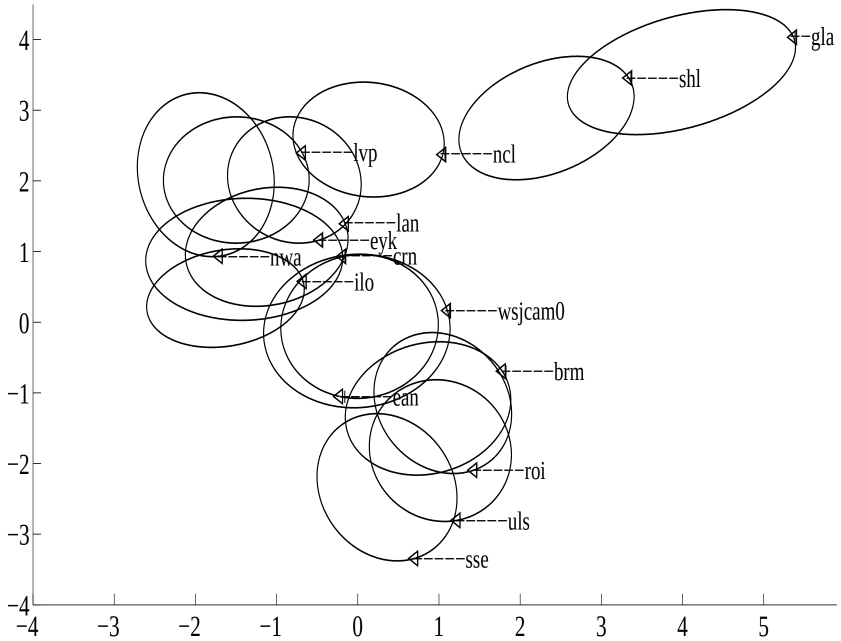

Fig. 9.6 *Visualization of the i-vector feature space for WSJCAM0 training set and ABI-1*

Our i-vector AID system is fully described in Section 6.2.2 and it can show how different accept groups are located in the accent space.

Figure 9.6 shows a representation of the AID i-vector space for the ABI-1 corpus. A cluster representing the WSJCAM0 has been added. LDA and PCA are used to map the 800 dimensional i-vector space onto 2 dimensions. For each 'accent region', 0.7-standard-deviation contours from the mean value represent utterances of that accent in the 'accent space'.

Our accent space consists of 3 main clusters, corresponding to northern English, Irish and Scottish accent groups. Southern English accents are scattered in the centre of the accent space. WSJCAM0 was recorded in Cambridge which is part of East Anglia (ean). However, the co-location of the WSJCAM0 and ean clusters in 2 dimensional accent space may be coincidental.

In order to facilitate analysis of the results presented in this section, the i-vector visualisation space provides a general overview of the regional accent distributions in the accent space.

9.4 Results and discussion

The results of the experiments presented in Section 9.2 are described in this section. Each experiment will be described separately in the following sections. For completeness, Table 9.2 provides a summary for the experiments reported in this chapter.

Table 9.2 *Summary of DNN-HMM results using various multi-accented data sets*

Type	Experiment	Code	WER (%)	AWR (%)	Regional accent	Broad accent	ETM length (hour)
Baseline	Baseline DNN	D0	6.85	0	-	-	0
Accent dependent	ETM: accent level	DS1	5.08	25.84	1	1	0.67
	ETM: broad accent group	DS2	4.93	28.03	[2:6]	1	2.25
Accent independent	ETM: size increased	DU1	4.91	28.32	14	4	2.25
	ETM: max size / all accents		4.39	35.91	14	4	8.96
	ETM: accent diversity increased	DU2	5.11	25.40	2	1	2.25
			5.01	26.86	4	3	2.25
			4.91	28.32	8	4	2.25
			4.91	28.32	14	4	2.25
	ETM: broad accent group changed	DU3	4.68	31.68	2	SC	2.25
			4.91	28.32	2	IR	2.25
			5.64	17.66	4	SO	2.25
			5.32	22.33	6	NO	2.25
	EPM: size and accent diversity increased	DUP	6.83	0.29	14	4	8.96
			6.76	1.31	14	4	17.79
			6.72	1.90	27	5	26.75

Results for the baseline DNN-HMM, and speaker and accent adapted GMM-HMM

In Section 8.4 we showed that in a GMM-HMM ASR system the maximum %WER reduction compared to the baseline (B0) is achieved by applying accent-specific acoustic model selection followed by speaker adaptation (BU).

In this section, we compare the baseline DNN-HMM accuracy (D0) with performance of the accent and speaker adapted GMM-HMM system (BU), and the baseline GMM-HMM system (B0).

Table 9.3 compares GMM-HMM and DNN-HMM performance, for the baseline system trained on WSJT. A relative gain of 46.85% is achieved in recognising the ABI-1 corpus by applying a baseline DNN model rather than GMM, which is 3.18% more than what we achieved after accent plus speaker adaptation in the GMM-HMM system. This shows that DNN based ASR systems are better than GMM based ASR systems in dealing with multi-accented data.

Table 9.3 *Comparison between baseline DNN-HMM, baseline GMM-HMM, and accent plus speaker adapted GMM-HMM results*

Code	B0	BU	D0
accent diversity	-	1	-
length of accent-specific adaptation data	0	40 min	0
%AWR	-	43.67	46.85
%WER	12.89	7.36	6.85

Looking at the results in Figure 9.7, although the error rates of the GMM-HMM for different accents are much lower than those reported in Chapter 8 (due to differences in training and decoding parameters, and sampling frequency we chose in the HTK and Kaldi toolkits [164]), there is still a clear effect of accent evident from the results and the relative gain achieved by applying accent followed by speaker adaptation compared to the baseline B0 remains in the similar range.

Looking at B0, the %WER for the most challenging accent (gla, 13.34%) is nearly 5 times bigger than that for standard southern English (sse, 2.84%). The GMM-HMM baseline results show clear effect of accent on the system accuracy.

The superior recognition results on the southern English and Irish accents with respect to the Scottish accents in all three systems (B0, D0, BU) matches our i-vector visualization in Figure 9.6. The WER for southern and Irish accents who seem closer to WSJCAM0 in the i-vector accent space is consistently higher than the Scottish and majority of northern accents.

Despite the fact that the DNN baseline D0 was never trained on ABI-1 accents, it follows a similar pattern as BU which was speaker and accent adapted. The DNN system works slightly more accurate than an accent- and speaker-adapted GMM system, except for recognition of the Scottish and a few northern English accents.

This result suggests that the DNN-HMM based system must have learnt how to address the accent issues during the training stage from WSJCAM0. This motivated us to run our i-vector based AID on the WSJCAM0 and investigate whether, despite our assumptions, WSJCAM0 contains other British regional accents (Section 9.5.1).

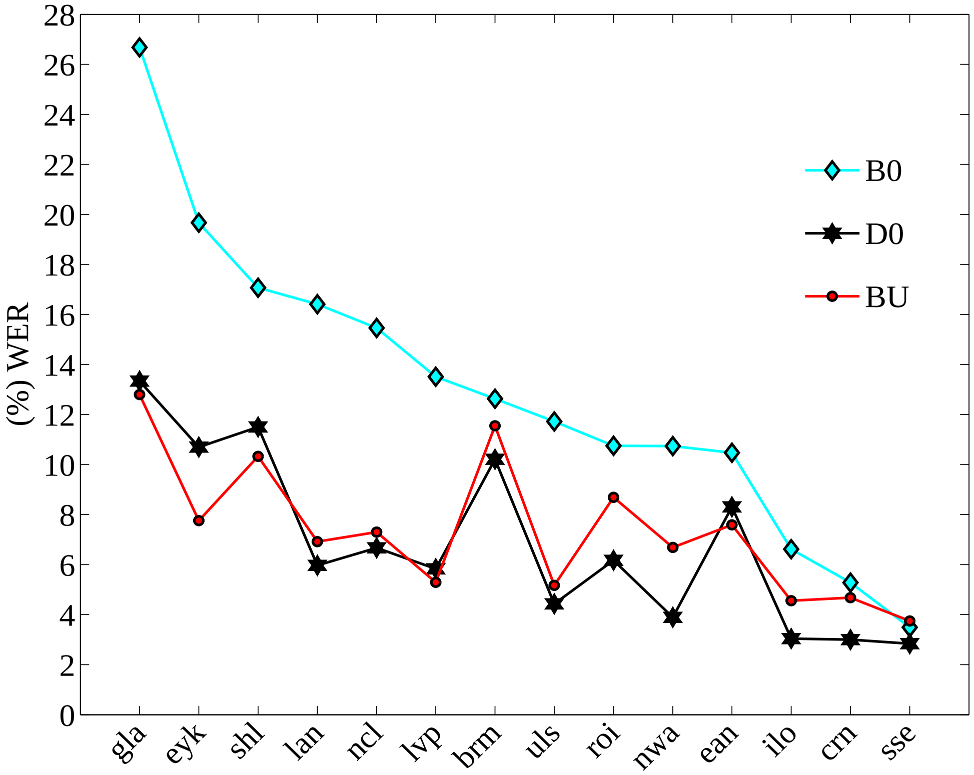

Fig. 9.7 *ASR results for baseline DNN-HMM, baseline GMM-HMM, and accent followed by speaker adapted GMM-HMM*

In all three systems (BU, B0, and D0), the poorest baseline performance belongs to the Scottish accents, namely gla and shl. This can be explained by their relatively large distance from the WSJCAM0 cluster in the i-vector accent space (Figure 9.7). As can be seen, using the DNN-HMM baseline, D0, rather than the GMM-HMM baseline, B0, has reduced the error rate for the most difficult accent (gla) without the need to accent or speaker adaptation like in system (BU).

Results for accent-dependent ETM (DS1, DS2) versus accent diverse ETM (DU1)

As described earlier accent-dependent experiments have the advantage that they add the ETM from an accent that matches the test speaker's accent. This helps the DNN to learn specifically the accent-specific characteristics of the test speaker in the fine-tuning phase. We investigate the effect of adding a small amount (40 minutes) of accent-dependent data that matches the speaker's accent in DS1 setup, or 2.25 hours of data that matches the speaker's broad accent group in DS2 setup. Obviously, there might not be enough accented data available that matches the test speaker's accent. Hence, the gain achieved by adding accent-specific data to the training set is limited due to lack of available accent-specific data.

On other hand, one might think that adding a larger amount of data from a diverse set of accents might be more feasible and the DNN's discriminative training is powerful enough to learn different accent properties even when an the training set contains wide range of accents.

To investigate this, we added small amount of accented data from 14 different accents to the training set, and tested the system on speakers with one of those 14 accents (DU1).

Compared to adding 40 minutes of accent-specific data based on the speaker's accent, 13.58% relative WER reduction is achieved by adding 13.5 times more ETM from different accents.

Table 9.4 *ASR performance after training an accent-specific versus a multi-accented network*

Code	D0	DS1	DS2	DU1
Accent diversity	-	1	[2:6]	14
ETM length	-	40min	2.25hrs	8.96hrs
%AWR	-	25.84	28.03	35.91
%WER	6.85	5.08	4.93	4.39

Table 9.4 compares the baseline result (D0) trained on WSJT with the results for WSJT supplemented with the ETM of length 40 minutes of accent-specific data (DS1), the ETM of length 2.25 hours from broad accent groups (DS2), and the ETM of length 8.96 hours from all 14 accents of ABI-1 (DU1) as described in Section 9.2.

In all three cases, inclusion of accented speech in the training data considerably improves the performance relative to the baseline D0.

The lowest WER of 4.39% is for accent-independent multi-accent training DU1 where the size and diversity of accented speech in the ETM is much higher than in accent-specific methods. This result suggests that addition of highly diverse data (in terms of accent) to the DNNs is not an issue and they can successfully learn different accent-specific properties during their discriminative fine-tuning to be able to address accent variations in the test data. In fact unsupervised addition of large amount of data (8.96 hours) with very diverse accent (covers all ABI-1 accents) has achieved 35.91% average WER reduction respect to the baseline which is the highest compared to the other accent-dependent and -independent experiments proposed in this chapter. However, it is not clear if this is due to greater accent diversity, or larger size of the training set. This is examined in the next two following sections.

Results for accent-independent ETM with varying size and fixed accent diversity (DU1)

This accent-independent experiment (described in Section 9.2.4) investigates the result of adding more accent diverse data while keeping the accent diversity unchanged (DU1). Fixing the accent diversity but increasing the size of the training set leads to improved performance. This is because as more training material becomes available for fine-tuning, the DNN parameters can be estimated more precisely.

Columns 2 and 3 of Table 9.5 show the improvement in performance obtained by adding additional accent-diverse ABI-1 training data. The average word error rate will reduce by 10.59% when we provide four times more accent diverse data from ABI-1 to the baseline training set.

Table 9.5 *ASR performance after supplementing the baseline with larger amount of accent diverse data*

Code	D0	DU1	
Accent diversity	-	All ABI-1 regions (14)	
Broad accent diversity	-	(NO,SC IR,SO)	
ETM length	-	2.25 hrs	8.96 hrs
%AWR	-	28.03	35.91
%WER	6.85	4.91	4.39

In this section we have investigated the relative gain achieved by increasing the amount of added data to the training set while keeping the diversity unchanged. In the next section we will investigate the relative improvement attained by keeping the added training data constant and increasing only the accent diversity.

Results for accent-independent ETM with fixed size and varying accent diversity (DU2)

This accent-independent experiment investigates the effect of increasing the accent diversity of ETM from minimum accent diversity in D0+WSJD (supplementing WSJT with WSJD) to maximum accent diversity in DU2 (supplementing WSJT with all 14 ABI-1 accents), while keeping the amount of added data constant (2.25 hours).

By adding the WSJCAM0 development set (WSJD) as the EMT, we can see a small reduction in WER compared to adding same amount of data from diverse set of accents. The latter (DU2 with 8 regional accents) has 20.3% more average word error rate reduction compared to the former (D0 + WSJD) with much limited accent diversity. Columns corresponding to DU2 in Table 9.6 show the result of increase in accent diversity from 2 to 14 accents. It is evident from the results that increasing the accent diversity from 2 to 8 reduces the WER by 3.91% but when the diversity increases from 8 to 14, the accuracy remains almost the same.

Looking at the average WER reduction (AWR) for target and off target accents shows for a fixed size of ETM when we add more data from a certain broad accent group the average WER reduction for those accents which match the ETM accent will increase dramatically (up to 42.5%). However, for the off target accents the average WER reduction is not considerable (up to 24.4%). This is interesting as it shows despite the fact that DNNs are good at dealing with multi-accent data, still addition of even small amount accented data (less than 2.25 hours) will have a major affect on those accents who belong to that accent group.

As we increase the accent diversity from 2 to 14 accents, the average WER on the target accents reduces, and the average WER on the off target accents increases. Again this proves that a more diverse ETM will provide more improvement in accuracy in general.

In this experiment, when the training set includes the material from the ABI-1 corpora, we present the AWR for those accents that were represented in the training set ('target'), and for accents that are not related in the training set ('off target').

Table 9.6 *ASR performance after adding 2.25 hours of data to the baseline and increasing the accent diversity*

Code	D0	D0+ WSJD	DU2			
Accent diversity	-	-	2	4	8	14
Broad accent diversity	-	WSJD	NO	(NO, SC,IR)	(NO,SC IR,SO)	(NO,SC IR,SO)
ETM length	2.25 hrs					
%AWR targets	-	-	42.5	16.8	18.2	21.9
%AWR off targets	-	-	14.2	22	29.0	-
%AWR all	-	8.02	25.40	26.86	28.32	28.32
%WER	6.85	6.30	5.11	5.01	4.91	4.91

Results for accent-independent ETM from various broad accent groups, fixed size and accent diversity (DU3)

This accent-independent experiment investigates the effect of adding ETM from various broad accent groups DU3 (supplementing WSJT with a single broad accent group from ABI-1), while keeping the amount of added data constant (2.25 hours).

In Table 9.7 the lowest WER is obtained by adding the Scottish (SC) data which is by far the most difficult accent. We have seen in Figure 9.6 that the Scottish accents appear to lie at the extremes of the 'accent space' and we speculate that this may be why adding SC

Table 9.7 *ASR performance after supplementing the baseline with 2.25 hours of data from four broad accent groups*

Code	D0	D0 + WSJD	DU3			
Accent diversity	-	-	2	2	4	6
Broad accent diversity	-	WSJD	SC	IR	SO	NO
EPM length	-	2.25 hrs				
%AWR targets	-	-	28.7	23.2	-4.3	19.3
%AWR off targets	-	-	24.4	21.6	8.7	12.4
%AWR All	-	8.02	31.68	28.32	17.66	22.33
%WER	6.85	6.30	4.68	4.91	5.64	5.32

supplement provides the highest gain (31.68% average WER reduction with respect to D0). After Scottish accents, SC, with highest target and off target average WER reduction, adding Irish (IR) accents provided a major reduction in WER of 28.32% on average. Interestingly adding SO didn't improve the target AWR and adding NO caused a small improvement in target and off-target recognition accuracy.

Comparing the results reported for the northern accents (DU2), comprising eyk and nwa, with the results reported for northern accents (DU3), comprising all six northern regional accents, we notice that adding more diversity to the ETM did not improve the recognition results. In fact, the average WER reduction on both target and off target accents has dropped from 42.5 and 14.2 in the former system to 19.3 and 12.4 respectively in the latter system. During the transformation from the former to the latter system the amount of data from the eyk accent has reduced to a third, in cost of a higher diversity in the latter system. Since the eyk accent was one of the difficult accents for our system, reduction of eyk in the latter system compared to the first one cause a considerable reduction in AWR on target accents.

Comparison between DU3 and DU2 systems for the cases where the 2.25 hour of ETM only comes from two regional accents shows some interesting points. Addition of two of the most difficult accents in Scottish accents SC (shl, gla) compared to the two northern accents NO (eyk, nwa) or two Irish accents (roi, uls) has the highest off target AWR reduction, while the highest target AWR reduction belongs to NO. This shows that adding even small amount (1.12 hours) of a difficult NO accent (eyk) can considerably reduce the WER, but for SC this not the case and maybe more data is required to further reduce their target WER. So the amount of target AWR relates to the nature of each accent. If the accents are considerably different from the training set like gla, shl and eyk, more data is required for them in ETM in order to reduce the target AWR.

Adding a data from a difficult broad accent group such as SC to the training set has a very positive effect on reducing even the off-target accents which can be concluded from the fact that the lowest average WER belongs to the experiment where the ETM only consists of SC compared to NO, and IR. Similarly adding 2.25 hours of SO which is the easiest accent provided very small AWR over all accents and achieved the lowest target and off target AWR.

Result of increase in size and accent diversity of the EPM (DUP)

Pre-training enables the DNN to learn aspects of the structure of the training data, which is advantageous for the fine-tuning. In this experiment we aim to measure the improvements achieved by using a more accent-diverse data for the pre-training.

Table 9.8 *ASR performance after adding more accent diverse data to the pre-traing set*

Code	D0’	D0	DUP		
EPM	-	WSJT	ABI-1	ABI-2	ABI-1 & ABI-2
Accent diversity	-	-	14	14	27
Broad accent diversity	-	-	(NO,SC IR,SO)	(NO,SC WA,SO)	(NO,SC,WA IR,SO)
EPM length	No pre-training	15.50	8.96	17.79	26.75
%AWR	-	-	0.29	1.31	1.90
%WER	6.95	6.85	6.83	6.76	6.72

Table 9.8 shows the result of supplementing the pre-training data with 8.96 hours of ABI-1 (14 accents, DUP), with 17.79 hours of data from the ABI-2 corpus (14 accents, DUP) and with 26.75 hours of data from ABI-1&2 (27 accents, DUP).

In an initial experiment (D0’) we did not pre-train the baseline and this resulted in 6.95% WER which is not far from the baseline result which is pre-trained on 15.50 hours of WSJCAM0 data D0. The last column of Table show that adding 26.75 hours of EPM from all 27 accents of ABI-1&2 will lead to only 1.89% relative WER reduction compared to the baseline D0. From this we conclude that adding even a large amount of highly diverse material at the pre-training stage will not result in a considerable change in overall WER.

9.5 Understanding the success of the baseline DNN system

An underlying assumption in the above experiments is that the WSJCAM0 corpus is ‘accent neutral’. In other words, it consists of general southern accented data. However, the relative success reported in Section 9.4 for the baseline DNN system trained on the WSJCAM0 questions this assumption.

Using a DNN-based baseline (D0) has reduced the average WER by 6.92% compared to the speaker and accent adapted GMM-HMM system (BU). In addition to that Figure 9.7

shows that the WER resulted from applying the baseline DNN (using Kaldi) and the accent- and speaker-adapted GMM-HMM (using Kaldi) are quite similar, despite the fact that the baseline DNN system D0 was never trained on any accented data from the ABI-1. These results motivated us to investigate whether WSJCAM0 contains other accents rather than southern accent, that has helped the DNN system to be able to successfully recognise the unseen accented data.

In this section we will investigate the accent properties of WSJCAM0 using an i-vector based AID system. We use our proposed i-vector based AID system (described in Section 6.2.2) with accuracy of 76.76% on the ABI-1 corpus to recognize accent variations in WSJCAM0 which was recorded in Cambridge.

9.5.1 Accent properties of WSJCAM0

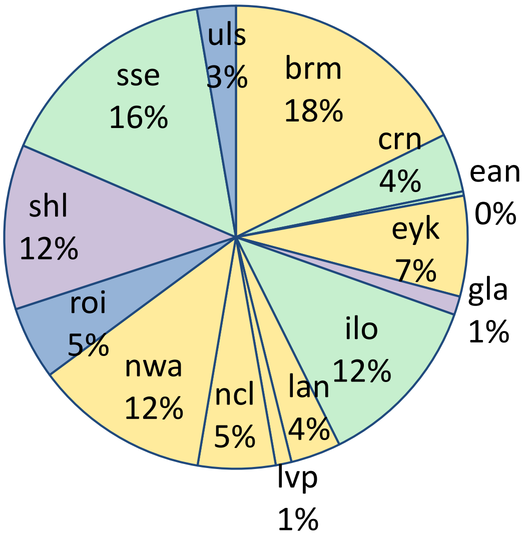

Fig. 9.8 *Result of applying the i-vector AID to the WSJCAM0 training set*

Our i-vector AID system is fully described in Section 6.2.2. We applied this i-vector AID system to the WSJCAM0 corpus to analyse its accent properties. Figure 9.8 suggests that 32%, 46%, 12% and 7% of the subjects who made the recordings of the WSJCAM0 corpus (training set only), exhibit traces of southern English, northern English, Scottish and Irish accents respectively.

This wide range of accent coverage in WSJCAM0 explains why our DNN-HMM baseline system in Section 9.4 performs so well. Also, the lack of Scottish and Irish accents resulted in achieving higher gain when more data is added from IR and SC accents rather than SO and NO in Section 9.4. Although WSJCAM0 was recorded in Cambridge which is part of East Anglia (ean), our AID system does not recognize them as native speakers of East Anglian English (ean) as the participants are from different parts of the country.

9.6 Summary and conclusions

The baseline results reported in Section 9.4 showed the strength of DNNs in dealing with multi-accented data. Our baseline GMM-HMM system even after two stages of accent followed by speaker adaptation, could not match the DNN-HMM results. Looking at the accent distribution of the WSJCAM0 using an i-vector based AID showed that, despite our previous hypothesis, WSJCAM0 is a multi-accented corpus (Figure 9.8). Hence, the baseline DNN-HMM system has managed to address accent variability of the multi-accented test set, and achieve a relative gain of 3.18% and 46.85% compared to the accent followed by speaker adapted, and the baseline GMM-HMM based systems respectively.

According to our visualisation of the AID space, the most difficult accent has accent properties most different from the rest of the training set. For example two Scottish accents, gla and shl, are on the edge of the accent space and including them in our training data results in high WER reduction compared to other accents (Table 9.7). In addition to including the data from more difficult accents in our training set, from Tables 9.5 and 9.6 we conclude that both increasing the accent diversity and size of the supplemented data are important in achieving lower WER. However, there is a limit to the improvement achieved through increasing the accent diversity. Given 2.25 hour of additional training material, in our experiment no gain in performance was observed by increasing the accent diversity from 8 to 14.

Table 9.2 summarises the results of experiments introduced in this chapter. Our results showed that adding 40 minutes of accent-specific ETM to the training set based on the test speakers accent region reduced the WER by 25.84%. Also, adding 2.25 hours of ETM from a broad accent region based on the speakers accent reduced the WER by 28.03. This result shows that, without the need for a separate adaptation stage, the DNN-based acoustic models can learn to address the accent issues from a small amount of accent-specific data added to the training set, based on the test speakers accent.

However, the best approach to deal with multi-accent data and achieve smallest average %WER over all accents is to study accent properties of the training data, and include as much data from a diverse set of accents in the training set (Section 9.4), but if this is not possible then data from more difficult accents should be included (Section 9.4), as these have the effect of stretching the training set and result in improvements across all accents.

We tested the effect of increasing the the size and accent diversity of the extra training material added to the training set. Our results showed that increasing the accent diversity by four times (from 2 to 8 regional accents) and keeping the ETM size constant (2.25 hours) has resulted in only 3.91% relative WER reduction, and adding four times more ETM (from 2.25 to 8.96 hours) to the training set while keeping the accent diversity fixed will result in 10.59% relative WER reduction.

We tested whether all the accent groups are equally useful as additional training data for accent robust ASR. Specifically we compared the recognition accuracies that resulted from adding 2.25 hours of ETM using five different resources. The results showed that the most gain is achieved through adding 2.25 hours of Scottish ETM, which results in 31.68% relative WER reduction compared to the baseline DNN-HMM system.

In the pre-training stage, increasing the accent diversity and size of the pre-training set can provide further reduction in WER through a better initialization. However, our results showed that the benefit achieved through pre-training on a large amount of accent diverse data is very small.

Chapter 10

Conclusions and future work

10.1 Overview

Four main approaches are proposed in the literature for addressing the problems caused by regional accents in ASR systems. Section 2.2 categorise them into namely front-end techniques, acoustic modelling techniques, pronunciation modelling techniques, and their combinations.

In this work we showed that accent mismatch between the training and the test data will result in significant accuracy reduction in the ASR performance. Our experiments showed that the ASR word error rate is up to seven times greater for accented speech compared with standard British English. The main objective of this research is to develop ASR techniques that are robust to accent variation. We applied different acoustic modelling techniques to compensate for the effects of regional accents on the ASR performance.

For conventional GMM-HMM based ASR systems, we showed that using a small amount of data from a test speaker to choose an accent dependent model using an accent identification system, or building a model using the data from N neighbouring speakers in AID space, will result in superior performance compared to that obtained with unsupervised or supervised speaker adaptation.

In addition we showed that using a DNN-HMM rather than a GMM-HMM based acoustic model would improve the recognition accuracy considerably. Even if we apply two stages of accent followed by speaker adaptation to the GMM-HMM baseline system, the GMM-HMM based system will not outperform the baseline DNN-HMM based system. For more contemporary DNN-HMM based ASR systems we investigated how adding different types of accented data to the training set can provide better recognition accuracy on accented speech.

To gain a better understanding of the effect of different regional accents on ASR performance, we proposed an accent space visualisation technique which represents the distributions of different regional accents in a two-dimensional accent feature space.

In this chapter we first review the important conclusions derived from the experiments described in this thesis and evaluate the main contributions of this thesis. Then we suggest ways in which this research might be extended in the future.

10.2 Practical contributions

In Section 1.4 we summarised the key contributions of this thesis. In this section we aim to elaborate on these key contributions by reviewing the important figures and results.

- The contributions related to Chapter 6 of this thesis are as following.
    - We reported the results for three different AID systems, namely ACCDIST-SVM based, phonotactic based, and i-vector based systems. We compared their accuracy and computational complexity with that of other AID techniques presented in the literature. We showed that fusing a single i-vector based and 15 phonotactic based AID systems rather than fusing 630 i-vector AID systems (proposed by Demarco et al. [17]) results in 4.5% relative AID error reduction and reduces the complexity of the system. This result suggests that the phonotactic AID and i-vector AID complement each other. Hence, their fusion can benefit from the accent specific information both at the phone level, obtained from the phonotactic system, and at the feature level, obtained from the i-vector system.
    - We also showed that using a multi-lingual phonotactic system comprising four language specific PRLMs (proposed by Hanani et al.[16]) rather than a multi-accent English phonotactic system comprising 15 accent specific PRLMs (includes one accent neutral PRLM) reduces the AID error rate by 7.5%. This is maybe due to the fact that in a multi-lingual phonotactic system there are a larger variety of phone sets to describe different acoustic events and the phonotactic system can use wider combination of phone sets from different languages to describe accent of an utterance. Our proposed AID systems in Chapter 6 were used for selection of accented acoustic models for ASR in Chapters 7 and 8.
    - We proposed a new approach for visualisation of the AID feature space and presented the result in chapter 6. This two-dimentional visualisation approach is applied to the supervectors from different AID systems. In this study the accent

space visualisation map represents distribution of different regional accents. We represented utterances that belong to each accent with one standard deviation contour. The distribution of these accent clusters provides general information about the similarities and differences between different regional accents from the perspective of each AID measure. For instance, regional accents that belong to the northern English broad accent group are located close to each other and further from Scottish, Irish and southern English accents in the accent visualisation map (Figures 6.3, 6.4, and 6.5). This is helpful in analysing the AID recognition accuracies and analysing AID confusion matrices. Visualisation maps can also assist analysis of the speech recognition results for utterances that belong to different regional accents.

- The contributions related to Chapters 7 and 8 of this thesis are as following.
  - Given a baseline GMM-HMM based ASR system trained on the WSJCAM0 corpus, experiments were carried out to recognise utterances from 285 speakers in the ABI-1 corpus with 14 different British English regional accents, we showed that accent is a significant issue for ASR systems. Given the same GMM-HMM based baseline, the ASR word error rate (WER) for difficult accents such as Glaswegian is up to seven times greater for accented speech compared with standard British English.
  - In all the GMM-HMM based acoustic model adaptation experiments reported in this work using the MLLR rather than the MAP adaptation technique to a higher accuracy. Hence, in this section we only report the MLLR based acoustic model adaptation results for adapting to the data from the test speaker (speaker adaptation) or to adapt to the data from other speakers with similar accent (accent-specific acoustic model selection). A summary of the recognition results after applying the different accent and speaker adaptation approaches proposed in Chapters 7 and 8 can be found in Tables 7.6 and 8.1.
  - Given a small amount of data from the test speaker, we proposed three approaches for accent-dependent modelling for reducing the accent issues in a GMM-HMM system, namely using the true accent model, choosing a model using an AID system, and building a model using data from neighbouring speakers in AID space. All these three techniques led to a much higher gain in ASR performance than obtained with unsupervised speaker adaptation. The gain achieved using five times more adaptation data from the test speaker for the unsupervised speaker

  adaptation could not match that of the accent adaptation experiment using our weakest AID system (Table 8.2).

  - We applied four different AID systems to accented model selection. These are systems based on the ACCDIST-SVM (95% AID accuracy), the i-vector approach proposed by DeMarco (81.05% AID accuracy), a phonotactic approach (80.65% AID accuracy) and a simplified i-vector based approach (76.76% AID accuracy). Using these systems for AID-based model selection resulted in relative reductions in WER of 43.16%, 41.39%, 41.22% and 40.59%, respectively. The similarity of the relative WER reductions for AID systems with varying performance indicates that the ASR performance obtained with AID-based model selection does not depend critically on the accuracy of the AID system. Thus, for real-time applications we recommend using a simple AID system that gives similar gain in ASR accuracy compared to a more accurate AID system that might be either text-dependent or requires a higher computational cost.
  - Some speakers might exhibit a mixture of multiple accents and therefore choosing an acoustic model based on the AID result might not match their multi-accent properties. We investigated this hypothesis by adapting the acoustic model to the data from the *N* closest speakers in the phonotactic based AID feature space. Comparing to the GMM-HMM baseline 39.45% relative WER reduction resulted from this experiment, which is not as good as the gain achieved by AID based acoustic model selection approach.

- The contributions related to Chapter 9 of this thesis are as following.
  - We investigated the effect of ABI-1 accented speech on the recognition accuracy of a baseline DNN-HMM based ASR system trained on WSJCAM0 with that of the accent-followed by speaker-adapted GMM-HMM based system. Our results showed that, using the former system rather than the latter will lead to 6.92% relative WER reduction. This result suggests that the accent mismatch between the training and test set is a less critical issue for the DNN-HMM based systems. We investigated how a baseline DNN-HMM based system has managed to successfully recognise ABI-1 utterances using only the training material from a supposedly accent neutral dataset (WSJCAM0). We applied an AID system to the WSJCAM0 dataset to understand its accent properties. The accent distribution of the WSJCAM0 training set according to our i-vector based AID system is shown in Figure 9.8. This result indicates that despite our hypothesis, WSJCAM0 is not an accent neutral database and it exhibits a range of different British English

regional accents. We believe that strength of the DNN based system in learning different accent properties from the WSJCAM0 corpus might have helped with recognising the accented speech from ABI-1 corpus.

- In a DNN-HMM based system using the true accent of the speaker to add 40 minutes of accent-specific Extra Training Material (ETM) to the supposedly accent neutral training set leads to 25.84% relative WER reduction compared to the baseline. This result shows that without the need for a separate adaptation stage the DNN based acoustic models can learn to address the accent issues from the small amount of accent-specific data added to the training set based on the test speakers accent.
- We examined the effect of supplementing the training with large amount of accent diverse data (8.96 hours from 14 accent regions). This experiment resulted in 35.91% average WER reduction compared to the baseline. This result is better than the result achieved by the system supplemented with 40 minutes accent-specific ETM selected based on the test speaker's accent. This made us wonder how much of this improvement in performance is due to the increase in accent diversity of the training set or an increase in the amount of data compared to the baseline system. Our results showed that increasing the accent diversity by four times (from 2 to 8 regional accents) and keeping the ETM size constant (2.25 hours) has resulted in only 3.91% relative WER reduction, while adding four times more ETM (from 2.25 to 8.96 hours) to the training set while keeping the accent diversity fixed will result in 10.59% relative WER reduction. This suggests that there is a limit to the improvement in accuracy achieved through increasing the accent diversity of the training set, and the majority of the improvement in the accuracy of a DNN-HMM based system is due to increase in the amount of data.
- We tested whether all the accent groups are equally useful as additional training data for accent robust ASR. Specifically we compared the recognition accuracies that resulted from adding 2.25 hours of ETM using five different resources, namely an accent diverse group, an Irish accent group, a Scottish accent group, a northern English accent group and a southern English accent group (Table 9.7). The results showed that adding 2.25 hours of Scottish ETM, and adding 2.25 hours of accent diverse data resulted in highest gain of all. They achieved 31.68% and 28.32% relative WER reduction compared to the baseline DNN based system respectively. Adding ETM from the most difficult broad accent group (the Scottish accent) reduced the relative WER by 28.7% on the test data from the same accent group (target), but also it has reduced the relative WER by

24.4% on other accent groups (non-target). However, adding a southern accented ETM has resulted in only 17.66% relative WER reduction compared to the DNN baseline. Adding a Scottish ETM and a southern English accented ETM resulted in the highest and the lowest gain in performance among all five different set ups. Analysing target and non-target %WER reduction achieved by adding ETM from broad accent groups, and analysing the relative average WER reduction on all the ABI-1 accents, indicates that not all accent groups play an equivalent role in improving the final recognition results, and incorporating data from certain accent groups is more beneficial.

– We investigated the effect of increasing both accent diversity and the amount of data at the pre-training stage (Table 9.8). The results show that adding even a large amount of highly accent diverse material at the pre-training stage will not result in a considerable change in overall performance. Compared to a DNN-HMM based recogniser that has been initialised randomly (not pre-trained), adding 42.25 hours of pre-training data from ABI-1&2 and WSJCAM0 comprising 27 regional accents will only led to 3.30% relative WER reduction.

10.3 Future work

Although the focus of this research is only at the acoustic model level, a complete solution to the problems caused by regional accents requires a combination of different approaches, such as adding accent related features at the feature level, or selecting a pronunciation dictionary based on the test speaker's accent. The work presented in this thesis can be developed in several ways.

- **Model adaptation in the DNN-HMM system**: Huang et al. [9] reported an improvement on accented speech recognition using a DNN-HMM acoustic model with an accent-specific top layer and shared bottom hidden layers. Given our current DNN-HMM system with multi-accent shared bottom hidden layers, we can add accent-dependent softmax layers. Adding an accent-specific layer on top of a multi-accent neural network acoustic model is one potential solution and will be addressed in our future work.

- **Adding accent discriminative acoustic features**: Recently Saon et al. [21] and Miao et al. [23] and Guta et al. [22] reported an improvement in ASR accuracy by incorporating speaker's i-vector at the feature level as an input to the DNN-HMM based system (Section 2.2). Given the speakers AID supervectors generated in this work, it is interesting to incorporate the speaker's phonotactic or i-vector based AID supervectors at the feature level as additional inputs to the DNN-HMM system. This way DNN acoustic models will be adapted to the speaker's accent.

- **Using accent dependent pronunciation dictionary**: Tjalve [29] has developed a set of British English accent-specific pronunciation dictionaries [165] during his PhD. He developed an Irish, a northern English, a southern English, an Scottish and a Welsh pronunciation dictionaries. In our future work we can investigate the improvement achieved by incorporating Tjvale's accent-specific pronunciation dictionaries in our DNN-HMM based systems.

- **Modelling the speech with newly proposed techniques**: Over the past four years, dramatic improvement occurred in the area of acoustic modelling for ASR. Three successful approaches, namely those based on DNN-HMMs [7, 166], Subspace Gaussian Mixture Model (SGMM)-HMMs [40], and Long Short-Term Memory Recurrent Neural Networks (LSTM RNNs) [167] were introduced. Soon, these methods started to gain more popularity than the GMM-HMM based approaches in modern ASR systems. It was demonstrated that using DNN, SGMM, and LSTM based approaches for acoustic modelling rather than GMMs provide a much higher speech recognition accuracy compared to that of the GMM based recognisers [5]. Recent studies showed that SGMM and DNN based systems provide similar results [168] and the LSTM RNN based systems provide the highest accuracy of all [167, 169, 170]. In our future studies, it is interesting to investigate the strength of SGMMs or LSTM RNNs based systems compared to the DNN based systems in modelling the accented speech.

10.4 Conclusion

This thesis proposed a framework for addressing and analysing accent issues faced in the GMM-HMM and DNN-HMM based ASR systems. Practical approaches have been proposed to understand and address the problems caused by regional accents.

In this chapter, we have summarised our main contributions, and showed that this work is a foundation on which future research can be built.

References

[1] C. Huang, T. Chen, and E. Chang, “Accent issues in large vocabulary continuous speech recognition,” *International Journal of Speech Technology*, vol. 7, no. 2-3, pp. 141–153, 2004.

[2] S. Young, G. Evermann, D. Kershaw, G. Moore, J. Odell, D. Ollason, D. Povey, V. Valtchev, and P. Woodland, *The HTK Book (ver. 3.2)*. Cambridge University Engineering Department, 2002.

[3] L. Rabiner, “A tutorial on hidden Markov models and selected applications in speech recognition,” *Proceedings of the IEEE*, vol. 77, no. 2, pp. 257–286, 1989.

[4] S. J. Young, J. J. Odell, and P. C. Woodland, “Tree-based state tying for high accuracy acoustic modelling,” in *Proceedings of the workshop on Human Language Technology*, pp. 307–312, Association for Computational Linguistics, 1994.

[5] D. Yu and L. Deng, *Automatic speech recognition: a deep learning approach*. Springer Publishing Company, Incorporated, 2014.

[6] G. E. Hinton, “A practical guide to training restricted Boltzmann machines,” in *Neural Networks: Tricks of the Trade - Second Edition*, pp. 599–619, 2012.

[7] G. Hinton, L. Deng, D. Yu, G. E. Dahl, A.-r. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. Sainath, *et al.*, “Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups,” *Signal Processing Magazine, IEEE*, vol. 29, no. 6, pp. 82–97, 2012.

[8] M. Benzeghiba, R. de Mori, O. Deroo, S. Dupont, T. Erbes, D. Jouvet, L. Fissore, P. Laface, A. Mertins, C. Ris, R. Rose, V. Tyagi, and C. Wellekens, “Automatic speech recognition and speech variability: A review,” *Speech Communication*, vol. 49, no. 10-11, pp. 763–786, 2007.

[9] Y. Huang, D. Yu, C. Liu, and Y. Gong, “Multi-accent deep neural network acoustic model with accent-specific top layer using the KLD-Regularized model adaptation,” in *Fifteenth Annual Conference of the International Speech Communication Association*, 2014.

[10] D. Van Compernolle, “Recognizing speech of goats, wolves, sheep and… non-natives,” *Speech Communication*, vol. 35, no. 1, pp. 71–79, 2001.

[11] C. Huang, T. Chen, S. Z. Li, E. Chang, and J.-L. Zhou, “Analysis of speaker variability.,” in *INTERSPEECH*, pp. 1377–1380, 2001.

[12] S. M. D'Arcy, M. J. Russell, S. R. Browning, and M. J. Tomlinson, "The Accents of the British Isles (ABI) corpus," *Proceedings Modélisations pour l'Identification des Langues*, pp. 115–119, 2004.

[13] T. Robinson, J. Fransen, D. Pye, J. Foote, and S. Renals, "WSJCAM0: A British English speech corpus for large vocabulary continuous speech recognition," vol. 1, (Detroit), pp. 81–84, 1995.

[14] N. Dehak, P. Kenny, R. Dehak, P. Dumouchel, and P. Ouellet, "Front-end factor analysis for speaker verification," *Audio, Speech, and Language Processing, IEEE Transactions on*, vol. 19, no. 4, pp. 788–798, 2011.

[15] O. Glembek, L. Burget, P. Matějka, M. Karafiát, and P. Kenny, "Simplification and optimization of i-vector extraction," in *Acoustics, Speech and Signal Processing (ICASSP), 2011 IEEE International Conference on*, pp. 4516–4519, IEEE, 2011.

[16] A. Hanani, M. J. Russell, and M. J. Carey, "Human and computer recognition of regional accents and ethnic groups from British English speech," *Computer Speech & Language*, vol. 27, no. 1, pp. 59–74, 2013.

[17] A. DeMarco and S. J. Cox, "Native accent classification via i-vectors and speaker compensation fusion," in *INTERSPEECH 2013, 14th Annual Conference of the International Speech Communication Association*, pp. 1472–1476, 2013.

[18] J. C. Wells, *Accents of English, Volume 2: The British Isles*, vol. 2. Cambridge University Press, 1982.

[19] Y. Zheng, R. Sproat, L. Gu, I. Shafran, H. Zhou, Y. Su, D. Jurafsky, R. Starr, and S.-Y. Yoon, "Accent detection and speech recognition for Shanghai-accented Mandarin," in *INTERSPEECH*, pp. 217–220, 2005.

[20] M. H. Bahari, R. Saeidi, D. Van Leeuwen, *et al.*, "Accent recognition using i-vector, gaussian mean supervector and gaussian posterior probability supervector for spontaneous telephone speech," in *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*, pp. 7344–7348, IEEE, 2013.

[21] G. Saon, H. Soltau, D. Nahamoo, and M. Picheny, "Speaker adaptation of neural network acoustic models using i-vectors," in *Automatic Speech Recognition and Understanding (ASRU), 2013 IEEE Workshop on*, pp. 55–59, IEEE, 2013.

[22] V. Gupta, P. Kenny, P. Ouellet, and T. Stafylakis, "I-vector-based speaker adaptation of deep neural networks for French broadcast audio transcription," in *Acoustics, Speech and Signal Processing (ICASSP), 2014 IEEE International Conference on*, pp. 6334–6338, IEEE, 2014.

[23] Y. Miao, H. Zhang, and F. Metze, "Towards speaker adaptive training of deep neural network acoustic models," 2014.

[24] L. Kat and P. Fung, "Fast accent identification and accented speech recognition," in *Acoustics, Speech, and Signal Processing, 1999. Proceedings., 1999 IEEE International Conference on*, vol. 1, pp. 221–224, IEEE, 1999.

[25] D. Vergyri, L. Lamel, and J.-L. Gauvain, “Automatic speech recognition of multiple accented English data.,” in *INTERSPEECH*, pp. 1652–1655, 2010.

[26] S. Goronzy, *Robust adaptation to non-native accents in automatic speech recognition*, vol. 2560 of *Lecture Notes in Computer Science*. Springer, 2002.

[27] S. Goronzy, S. Rapp, and R. Kompe, “Generating non-native pronunciation variants for lexicon adaptation,” *Speech Communication*, vol. 42, no. 1, pp. 109–123, 2004.

[28] J. J. Humphries and P. C. Woodland, “Using accent-specific pronunciation modelling for improved large vocabulary continuous speech recognition,” in *Fifth European Conference on Speech Communication and Technology, EUROSPEECH*, 1997.

[29] M. Tjalve and M. Huckvale, “Pronunciation variation modelling using accent features,” in *INTERSPEECH 2005 - Eurospeech, 9th European Conference on Speech Communication and Technology*, pp. 1341–1344, 2005.

[30] Z. Wang, T. Schultz, and A. Waibel, “Comparison of acoustic model adaptation techniques on non-native speech,” in *Acoustics, Speech, and Signal Processing, 2003. Proceedings.(ICASSP’03). 2003 IEEE International Conference on*, vol. 1, pp. I–540, IEEE, 2003.

[31] U. Nallasamy, F. Metze, and T. Schultz, “Enhanced polyphone decision tree adaptation for accented speech recognition,” in *INTERSPEECH 2012, 13th Annual Conference of the International Speech Communication Association*, pp. 1902–1905, 2012.

[32] U. Nallasamy, F. Metze, and T. Schultz, “Active learning for accent adaptation in automatic speech recognition,” in *Spoken Language Technology Workshop (SLT), 2012 IEEE*, pp. 360–365, IEEE, 2012.

[33] H. Kamper, F. J. M. Mukanya, and T. Niesler, “Multi-accent acoustic modelling of South African English,” *Speech Communication*, vol. 54, no. 6, pp. 801–813, 2012.

[34] D. Yu, F. Seide, and G. Li, “Conversational speech transcription using context-dependent deep neural networks,” in *Proceedings of the 29th International Conference on Machine Learning, ICML 2012*, 2012.

[35] G. E. Dahl, D. Yu, L. Deng, and A. Acero, “Context-dependent pre-trained deep neural networks for large-vocabulary speech recognition,” *IEEE Transactions on Audio, Speech & Language Processing*, vol. 20, no. 1, pp. 30–42, 2012.

[36] I. J. Goodfellow, Q. V. Le, A. M. Saxe, H. Lee, and A. Y. Ng, “Measuring invariances in deep networks,” in *Advances in Neural Information Processing Systems 22: 23rd Annual Conference on Neural Information Processing Systems 2009.*, pp. 646–654, 2009.

[37] Y. Huang, M. Slaney, M. L. Seltzer, and Y. Gong, “Towards better performance with heterogeneous training data in acoustic modeling using deep neural networks,” in *INTERSPEECH 2014, 15th Annual Conference of the International Speech Communication Association*, pp. 845–849, 2014.

[38] Y. R. Oh and H. K. Kim, “MLLR/MAP adaptation using pronunciation variation for non-native speech recognition,” in *Automatic Speech Recognition & Understanding, 2009. ASRU 2009. IEEE Workshop on*, pp. 216–221, IEEE, 2009.

[39] K. Kirchhoff and D. Vergyri, “Cross-dialectal acoustic data sharing for arabic speech recognition.,” in *ICASSP (1)*, pp. 765–768, 2004.

[40] D. Povey, L. Burget, M. Agarwal, P. Akyazi, F. Kai, A. Ghoshal, O. Glembek, N. Goel, M. Karafiát, A. Rastrow, *et al.*, “The subspace Gaussian mixture model—a structured model for speech recognition,” *Computer Speech & Language*, vol. 25, no. 2, pp. 404–439, 2011.

[41] P. Motlicek, P. N. Garner, N. Kim, and J. Cho, “Accent adaptation using subspace Gaussian mixture models,” in *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*, pp. 7170–7174, IEEE, 2013.

[42] X. Chen and J. Cheng, “Deep neural network acoustic modeling for native and non-native mandarin speech recognition,” in *Chinese Spoken Language Processing (ISCSLP), 2014 9th International Symposium on*, pp. 6–9, IEEE, 2014.

[43] M. Chen, Z. Yang, J. Liang, Y. Li, and W. Liu, “Improving deep neural networks based multi-accent mandarin speech recognition using i-vectors and accent-specific top layer,” in *Interspeech*, 2015.

[44] T. Hazen, “Automatic language identification using a segment-based approach,” 1993.

[45] R. Tucker, M. Carey, and E. Parris, “Automatic language identification using sub-word models,” in *Acoustics, Speech, and Signal Processing, 1994. ICASSP-94., 1994 IEEE International Conference on*, vol. 1, pp. I–301, IEEE, 1994.

[46] A. S. House and E. P. Neuburg, “Toward automatic identification of the language of an utterance. i. preliminary methodological considerations,” *The Journal of the Acoustical Society of America*, vol. 62, no. 3, pp. 708–713, 1977.

[47] M. Zissman, E. Singer, *et al.*, “Automatic language identification of telephone speech messages using phoneme recognition and n-gram modeling,” in *Acoustics, Speech, and Signal Processing, 1994. ICASSP-94., 1994 IEEE International Conference on*, vol. 1, pp. I–305, IEEE, 1994.

[48] Y. Yan and E. Barnard, “An approach to automatic language identification based on language-dependent phone recognition,” in *Acoustics, Speech, and Signal Processing, 1995. ICASSP-95., 1995 International Conference on*, vol. 5, pp. 3511–3514, IEEE, 1995.

[49] M. Zissman, T. P. Gleason, D. M. Rekart, B. L. Losiewicz, *et al.*, “Automatic dialect identification of extemporaneous conversational, latin american spanish speech,” in *Acoustics, Speech, and Signal Processing, 1996. ICASSP-96. Conference Proceedings., 1996 IEEE International Conference on*, vol. 2, pp. 777–780, IEEE, 1996.

[50] C. Cortes and V. Vapnik, “Support-vector networks,” *Machine learning*, vol. 20, no. 3, pp. 273–297, 1995.

[51] V. Vapnik, *The nature of statistical learning theory*. Springer Science & Business Media, 2000.

[52] N. Brummer and D. A. van Leeuwen, "On calibration of language recognition scores," in *Speaker and Language Recognition Workshop, 2006. IEEE Odyssey 2006: The*, pp. 1–8, IEEE, 2006.

[53] D. A. Van Leeuwen and N. Brummer, "Channel-dependent GMM and Multi-class Logistic Regression models for language Recognition," in *Speaker and Language Recognition Workshop, 2006. IEEE Odyssey 2006: The*, pp. 1–8, IEEE, 2006.

[54] F. S. Richardson, W. M. Campbell, and P. A. Torres-Carrasquillo, "Discriminative n-gram selection for dialect recognition.," in *INTERSPEECH*, pp. 192–195, 2009.

[55] F. Biadsy, J. Hirschberg, and N. Habash, "Spoken arabic dialect identification using phonotactic modeling," in *Proceedings of the EACL 2009 Workshop on Computational Approaches to Semitic Languages*, pp. 53–61, Association for Computational Linguistics, 2009.

[56] A. Hanani, M. J. Russell, and M. J. Carey, "Human and computer recognition of regional accents and ethnic groups from British English speech," *Computer Speech & Language*, vol. 27, no. 1, pp. 59–74, 2013.

[57] L.-F. Zhai, M.-h. Siu, X. Yang, and H. Gish, "Discriminatively trained language models using Vupport Vector Machines for language identification," in *Speaker and Language Recognition Workshop, 2006. IEEE Odyssey 2006: The*, pp. 1–6, IEEE, 2006.

[58] L.-F. Zhai, M.-H. Siu, X. Yang, and H. Gish, "Discriminatively trained language models using support vector machines for language identification," in *Speaker and Language Recognition Workshop, 2006. IEEE Odyssey 2006: The*, pp. 1–6, June 2006.

[59] M. Senoussaoui, P. Kenny, N. Dehak, and P. Dumouchel, "An i-vector extractor suitable for speaker recognition with both microphone and telephone speech.," in *Odyssey*, p. 6, 2010.

[60] N. Dehak, P. Kenny, R. Dehak, P. Dumouchel, and P. Ouellet, "Front-end factor analysis for speaker verification," *IEEE Transactions on Audio, Speech & Language Processing*, vol. 19, no. 4, pp. 788–798, 2011.

[61] M. Soufifar, M. Kockmann, L. Burget, O. Plchot, O. Glembek, and T. Svendsen, "Ivector approach to phonotactic language recognition.," in *INTERSPEECH*, pp. 2913–2916, 2011.

[62] N. Dehak, P. A. Torres-Carrasquillo, D. A. Reynolds, and R. Dehak, "Language recognition via i-vectors and dimensionality reduction.," in *INTERSPEECH*, pp. 857–860, 2011.

[63] A. DeMarco and S. J. Cox, "Iterative classification of regional British accents in i-vector space," in *2012 Symposium on Machine Learning in Speech and Language Processing, MLSLP 2012*, pp. 1–4, 2012.

[64] P. Kenny, G. Boulianne, P. Ouellet, and P. Dumouchel, “Joint factor analysis versus Eigenchannels in speaker recognition,” *IEEE Transactions on Audio, Speech & Language Processing*, vol. 15, no. 4, pp. 1435–1447, 2007.

[65] P. Kenny, “Joint factor analysis of speaker and session variability: Theory and algorithms,” *CRIM,(Report) CRIM-06/08-13*, 2005.

[66] P. Kenny, G. Boulianne, and P. Dumouchel, “Eigenvoice modeling with sparse training data,” *IEEE Transactions on Speech and Audio Processing*, vol. 13, no. 3, pp. 345–354, 2005.

[67] D. A. Reynolds, T. F. Quatieri, and R. B. Dunn, “Speaker verification using adapted Gaussian mixture models,” *Digital Signal Processing*, vol. 10, no. 1-3, pp. 19–41, 2000.

[68] A. P. Dempster, N. M. Laird, and D. B. Rubin, “Maximum likelihood from incomplete data via the EM algorithm,” *Journal of the royal statistical society. Series B (methodological)*, pp. 1–38, 1977.

[69] P. Kenny, P. Ouellet, N. Dehak, V. Gupta, and P. Dumouchel, “A study of interspeaker variability in speaker verification,” *IEEE Transactions on Audio, Speech & Language Processing*, vol. 16, no. 5, pp. 980–988, 2008.

[70] N. Dehak, *Discriminative and generative approaches for long-and short-term speaker characteristics modeling: application to speaker verification*. Ecole de Technologie Superieure (Canada), 2009.

[71] P. Kenny, G. Boulianne, and P. Dumouchel, “Eigenvoice modeling with sparse training data,” *Speech and Audio Processing, IEEE Transactions on*, vol. 13, no. 3, pp. 345–354, 2005.

[72] M. Huckvale, “ACCDIST: an accent similarity metric for accent recognition and diagnosis,” in *Speaker Classification II, Selected Projects*, pp. 258–275, 2007.

[73] R. A. Fisher, “The use of multiple measurements in taxonomic problems,” *Annals of Eugenics*, vol. 7, no. 2, pp. 179–188, 1936.

[74] I. Jolliffe, *Principal component analysis*. Wiley Online Library, 2002.

[75] C. M. Bishop *et al.*, *Pattern recognition and machine learning*, vol. 4. springer New York, 2006.

[76] R. Huang, Q. Liu, H. Lu, and S. Ma, “Solving the small sample size problem of LDA,” in *16th International Conference on Pattern Recognition, ICPR 2002*, pp. 29–32, 2002.

[77] S. Roweis, “Em algorithms for PCA and SPCA,” *Advances in neural information processing systems*, pp. 626–632, 1998.

[78] F. Jelinek, *Statistical methods for speech recognition*. MIT press, 1997.

[79] M. J. F. Gales and S. J. Young, “The application of hidden Markov models in speech recognition,” *Foundations and Trends in Signal Processing*, vol. 1, no. 3, pp. 195–304, 2007.

[80] S. B. Davis and P. Mermelstein, “Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences,” *Acoustics, Speech and Signal Processing, IEEE Transactions on*, vol. 28, no. 4, pp. 357–366, 1980.

[81] L. Deng and D. X. Sun, “A statistical approach to automatic speech recognition using the atomic speech units constructed from overlapping articulatory features,” *The Journal of the Acoustical Society of America*, vol. 95, no. 5, pp. 2702–2719, 1994.

[82] H. Hermansky, “Perceptual linear predictive (PLP) analysis of speech,” *the Journal of the Acoustical Society of America*, vol. 87, no. 4, pp. 1738–1752, 1990.

[83] L. Deng and N. Jaitly, “Deep discriminative and generative models for pattern recognition,” in *USENIX–Advanced Computing Systems Association*, 2015.

[84] T. N. Sainath, R. J. Weiss, A. Senior, K. W. Wilson, and O. Vinyals, “Learning the speech front-end with raw waveform CLDNNs,” in *Sixteenth Annual Conference of the International Speech Communication Association*, 2015.

[85] Z. Tüske, P. Golik, R. Schlüter, and H. Ney, “Acoustic modeling with deep neural networks using raw time signal for LVCSR,” in *INTERSPEECH 2014, 15th Annual Conference of the International Speech Communication Association, Singapore, September 14-18, 2014*, pp. 890–894, 2014.

[86] R. J. van Son and L. C. Pols, “An acoustic profile of consonant reduction,” in *Spoken Language, 1996. ICSLP 96. Proceedings., Fourth International Conference on*, vol. 3, pp. 1529–1532, IEEE, 1996.

[87] J. W. Cooley and J. W. Tukey, “An algorithm for the machine calculation of complex Fourier series,” *Mathematics of computation*, vol. 19, no. 90, pp. 297–301, 1965.

[88] S. S. Stevens and J. Volkmann, “The relation of pitch to frequency: A revised scale,” *The American Journal of Psychology*, pp. 329–353, 1940.

[89] Y. A. LeCun, L. Bottou, G. B. Orr, and K.-R. Müller, “Efficient backprop,” in *Neural networks: Tricks of the trade*, pp. 9–48, Springer, 2012.

[90] T. Hain, P. C. Woodland, T. R. Niesler, and E. W. Whittaker, “The 1998 HTK system for transcription of conversational telephone speech,” in *Acoustics, Speech, and Signal Processing, 1999. Proceedings., 1999 IEEE International Conference*, vol. 1, pp. 57–60, IEEE, 1999.

[91] T. N. Sainath, B. Kingsbury, A.-r. Mohamed, and B. Ramabhadran, “Learning filter banks within a deep neural network framework,” in *Automatic Speech Recognition and Understanding (ASRU), 2013 IEEE Workshop on*, pp. 297–302, IEEE, 2013.

[92] W. Holmes, *Speech synthesis and recognition.* CRC press, 2001.

[93] S. Furui, “Speaker-independent isolated word recognition using dynamic features of speech spectrum,” *Acoustics, Speech and Signal Processing, IEEE Transactions on*, vol. 34, no. 1, pp. 52–59, 1986.

[94] L. Deng, J. Li, J.-T. Huang, K. Yao, D. Yu, F. Seide, M. Seltzer, G. Zweig, X. He, J. Williams, *et al.*, “Recent advances in deep learning for speech research at Microsoft,” in *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*, pp. 8604–8608, IEEE, 2013.

[95] A.-r. Mohamed, G. Hinton, and G. Penn, “Understanding how deep belief networks perform acoustic modelling,” in *Acoustics, Speech and Signal Processing (ICASSP), 2012 IEEE International Conference on*, pp. 4273–4276, IEEE, 2012.

[96] S. Katz, “Estimation of probabilities from sparse data for the language model component of a speech recognizer,” *Acoustics, Speech and Signal Processing, IEEE Transactions on*, vol. 35, no. 3, pp. 400–401, 1987.

[97] H. Ney, U. Essen, and R. Kneser, “On structuring probabilistic dependences in stochastic language modelling,” *Computer Speech & Language*, vol. 8, no. 1, pp. 1–38, 1994.

[98] S. F. Chen and J. Goodman, “An empirical study of smoothing techniques for language modeling,” in *Proceedings of the 34th annual meeting on Association for Computational Linguistics*, pp. 310–318, Association for Computational Linguistics, 1996.

[99] L. E. Baum, J. A. Eagon, *et al.*, “An inequality with applications to statistical estimation for probabilistic functions of markov processes and to a model for ecology,” *Bull. Amer. Math. Soc*, vol. 73, no. 3, pp. 360–363, 1967.

[100] L. E. Baum, T. Petrie, G. Soules, and N. Weiss, “A maximization technique occurring in the statistical analysis of probabilistic functions of markov chains,” *The annals of mathematical statistics*, pp. 164–171, 1970.

[101] Z. Ghahramani, “An introduction to hidden Markov models and bayesian networks,” *IJPRAI*, vol. 15, no. 1, pp. 9–42, 2001.

[102] L. Liporace *et al.*, “Maximum likelihood estimation for multivariate observations of markov sources,” *Information Theory, IEEE Transactions on*, vol. 28, no. 5, pp. 729–734, 1982.

[103] H. A. Bourlard and N. Morgan, *Connectionist speech recognition: a hybrid approach*, vol. 247. Springer Science & Business Media, 2012.

[104] X. Huang, A. Acero, H.-W. Hon, and R. Foreword By-Reddy, *Spoken language processing: A guide to theory, algorithm, and system development*. Prentice Hall PTR, 2001.

[105] Z. Ghahramani, “An introduction to hidden Markov models and Bayesian networks,” *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 15, no. 01, pp. 9–42, 2001.

[106] A. Poritz, "Hidden Markov models: A guided tour," in *Acoustics, Speech, and Signal Processing, 1988. ICASSP-88., 1988 International Conference on*, pp. 7–13, IEEE, 1988.

[107] J. A. Bilmes *et al.*, "A gentle tutorial of the EM algorithm and its application to parameter estimation for Gaussian mixture and hidden Markov models," 1998.

[108] P. Bansal, A. Kant, S. Kumar, A. Sharda, and S. Gupta, "Improved hybrid model of HMM-GMM for speech recognition," 2008.

[109] L. E. Baum and T. Petrie, "Statistical inference for probabilistic functions of finite state markov chains," *The annals of mathematical statistics*, pp. 1554–1563, 1966.

[110] L. E. Baum and G. Sell, "Growth functions for transformations on manifolds," *Pac. J. Math*, vol. 27, no. 2, pp. 211–227, 1968.

[111] P. C. Woodland, J. J. Odell, V. Valtchev, and S. J. Young, "Large vocabulary continuous speech recognition using HTK," in *Proceedings of ICASSP '94: IEEE International Conference on Acoustics, Speech and Signal Processing, Adelaide, South Australia, Australia, April 19-22, 1994*, pp. 125–128, 1994.

[112] A. Kannan, M. Ostendorf, and J. R. Rohlicek, "Maximum likelihood clustering of Gaussians for speech recognition," *Speech and Audio Processing, IEEE Transactions on*, vol. 2, no. 3, pp. 453–455, 1994.

[113] E. Trentin and M. Gori, "A survey of hybrid ANN/HMM models for automatic speech recognition," *Neurocomputing*, vol. 37, no. 1-4, pp. 91–126, 2001.

[114] H. Bourlard and N. Morgan, "Continuous speech recognition by connectionist statistical methods," *IEEE Transactions on Neural Networks*, vol. 4, no. 6, pp. 893–909, 1993.

[115] N. Morgan and H. Bourlard, "Continuous speech recognition using multilayer perceptrons with hidden Markov models," in *Acoustics, Speech, and Signal Processing, 1990. ICASSP, 1990 International Conference on*, pp. 413–416, IEEE, 1990.

[116] S. Renals, N. Morgan, H. Bourlard, M. Cohen, and H. Franco, "Connectionist probability estimators in HMM speech recognition," *IEEE Transactions on Speech and Audio Processing*, vol. 2, no. 1, pp. 161–174, 1994.

[117] H. Franco, M. Cohen, N. Morgan, D. E. Rumelhart, and V. Abrash, "Context-dependent connectionist probability estimation in a hybrid hidden Markov model-neural net speech recognition system," *Computer Speech & Language*, vol. 8, no. 3, pp. 211–222, 1994.

[118] J. Hennebert, C. Ris, H. Bourlard, S. Renals, and N. Morgan, "Estimation of global posteriors and forward-backward training of hybrid HMM/ANN systems," in *Fifth European Conference on Speech Communication and Technology, EUROSPEECH*, 1997.

[119] Y. Yan, M. A. Fanty, and R. A. Cole, “Speech recognition using neural networks with forward-backward probability generated targets,” in *1997 IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP ’97*, pp. 3241–3244, 1997.

[120] A. J. Robinson, G. D. Cook, D. P. W. Ellis, E. Fosler-Lussier, S. Renals, and D. A. G. Williams, “Connectionist speech recognition of Broadcast News,” *Speech Communication*, vol. 37, no. 1-2, pp. 27–45, 2002.

[121] G. E. Hinton, “Training products of experts by minimizing contrastive divergence,” *Neural computation*, vol. 14, no. 8, pp. 1771–1800, 2002.

[122] M. A. Carreira-Perpinan and G. E. Hinton, “On contrastive divergence learning,” in *Proceedings of the tenth international workshop on artificial intelligence and statistics*, pp. 33–40, 2005.

[123] G. E. Hinton, S. Osindero, and Y. W. Teh, “A fast learning algorithm for deep belief nets,”

[124] C. Allauzen, M. Riley, J. Schalkwyk, W. Skut, and M. Mohri, “Openfst: A general and efficient weighted finite-state transducer library,” in *Implementation and Application of Automata, 12th International Conference, CIAA 2007*, pp. 11–23, 2007.

[125] M. Mohri, F. Pereira, and M. Riley, “Weighted finite-state transducers in speech recognition,” *Computer Speech & Language*, vol. 16, no. 1, pp. 69–88, 2002.

[126] N. Morgan, H. Bourlard, *et al.*, “Neural networks for statistical recognition of continuous speech,” *Proceedings of the IEEE*, vol. 83, no. 5, pp. 742–772, 1995.

[127] V. V. Digalakis, D. Rtischev, and L. G. Neumeyer, “Speaker adaptation using constrained estimation of Gaussian mixtures,” *Speech and Audio Processing, IEEE Transactions on*, vol. 3, no. 5, pp. 357–366, 1995.

[128] M. J. F. Gales, “Maximum likelihood linear transformations for HMM-based speech recognition,” *Computer Speech & Language*, vol. 12, no. 2, pp. 75–98, 1998.

[129] T. Anastasakos, J. McDonough, R. Schwartz, and J. Makhoul, “A compact model for speaker-adaptive training,” in *Spoken Language, 1996. ICSLP 96. Proceedings., Fourth International Conference on*, vol. 2, pp. 1137–1140, IEEE, 1996.

[130] P. C. Woodland, “Speaker adaptation for continuous density HMMs: A review,” in *ISCA Tutorial and Research Workshop on Adaptation Methods for Speech Recognition*, 2001.

[131] C. J. Leggetter and P. C. Woodland, “Maximum likelihood linear regression for speaker adaptation of continuous density hidden Markov models,” *Computer Speech & Language*, vol. 9, no. 2, pp. 171–185, 1995.

[132] “the htk book. revised for htk version 3.2.”

[133] C. Leggetter and P. Woodland, “Flexible speaker adaptation using maximum likelihood linear regression,” in *Proc. ARPA Spoken Language Technology Workshop*, vol. 9, pp. 110–115, 1995.

[134] M. J. Gales, *The generation and use of regression class trees for MLLR adaptation*. University of Cambridge, Department of Engineering, 1996.

[135] J. Gauvain and C. Lee, “Bayesian learning of Gaussian mixture densities for hidden Markov models,” in *Speech and Natural Language, Proceedings of a Workshop, USA*, 1991.

[136] D. Yu, L. Deng, and G. Dahl, “Roles of pre-training and fine-tuning in context-dependent DBN-HMMs for real-world speech recognition,” in *Proc. NIPS Workshop on Deep Learning and Unsupervised Feature Learning*, 2010.

[137] M. L. Minsky and S. Papert, “a.(1969). perceptrons.”

[138] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, “Learning internal representations by error propagation,” tech. rep., DTIC Document, 1986.

[139] K. Hornik, M. Stinchcombe, and H. White, “Multilayer feedforward networks are universal approximators,” *Neural networks*, vol. 2, no. 5, pp. 359–366, 1989.

[140] G. E. Hinton, “Connectionist learning procedures,” *Artificial intelligence*, vol. 40, no. 1, pp. 185–234, 1989.

[141] L. Bottou, “Online learning and stochastic approximations,” *On-line learning in neural networks*, vol. 17, no. 9, p. 25, 1998.

[142] R. Salakhutdinov and G. E. Hinton, “Deep Boltzmann machines,” in *International Conference on Artificial Intelligence and Statistics*, pp. 448–455, 2009.

[143] G. E. Hinton and R. R. Salakhutdinov, “Reducing the dimensionality of data with neural networks,” *Science*, vol. 313, no. 5786, pp. 504–507, 2006.

[144] D. Erhan, Y. Bengio, A. Courville, P.-A. Manzagol, P. Vincent, and S. Bengio, “Why does unsupervised pre-training help deep learning?,” *The Journal of Machine Learning Research*, vol. 11, pp. 625–660, 2010.

[145] D. T. Toledano, A. Ortega, A. Teixeira, J. Gonzalez-Rodriguez, L. Hernandez-Gomez, R. San-Segundo, and D. Ramos, *Advances in Speech and Language Technologies for Iberian Languages: IberSPEECH 2012 Conference Proceedings*, vol. 328. Springer, 2012.

[146] D. H. Ackley, G. E. Hinton, and T. J. Sejnowski, “A learning algorithm for boltzmann machines,” *Cognitive Science*, vol. 9, no. 1, pp. 147–169, 1985.

[147] G. E. Hinton, S. Osindero, and Y. W. Teh, “A fast learning algorithm for deep belief nets,” *Neural Computation*, vol. 18, no. 7, pp. 1527–1554, 2006.

[148] A. Mohamed, T. N. Sainath, G. E. Dahl, B. Ramabhadran, G. E. Hinton, and M. A. Picheny, “Deep belief networks using discriminative features for phone recognition,” in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP*, pp. 5060–5063, 2011.

[149] T. N. Sainath, B. Kingsbury, B. Ramabhadran, P. Fousek, P. Novák, and A. Mohamed, "Making deep belief networks effective for large vocabulary continuous speech recognition," in *2011 IEEE Workshop on Automatic Speech Recognition & Understanding, ASRU*, pp. 30–35, 2011.

[150] R. C. S. L. L. Giles, "Overfitting in neural nets: Backpropagation, conjugate gradient, and early stopping," in *Advances in Neural Information Processing Systems 13: Proceedings of the 2000 Conference*, vol. 13, p. 402, MIT Press, 2001.

[151] J. Fransen, D. Pye, T. Robinson, P. Woodland, and S. Young, *WSJCAM0 corpus and recording description*. 1994.

[152] S. Safavi, M. Russell, and P. Jančovič, "Identification of age-group from children's speech by computers and humans," in *Interspeech*, 2014.

[153] S. O. Sadjadi, M. Slaney, and L. Heck, "MSR identity toolbox v1. 0: A MATLAB toolbox for speaker recognition research," *Speech and Language Processing Technical Committee Newsletter*, 2013.

[154] C. Vair, D. Colibro, F. Castaldo, E. Dalmasso, and P. Laface, "Channel factors compensation in model and feature domain for speaker recognition," in *Speaker and Language Recognition Workshop, 2006. IEEE Odyssey 2006: The*, pp. 1–6, IEEE, 2006.

[155] X. Zeng, J. Yang, and D. Xu, "Approaches to language identification using Gaussian mixture model and linear discriminant analysis," in *Intelligent Information Technology Application Workshops, 2008. IITAW'08. International Symposium on*, pp. 1109–1112, IEEE, 2008.

[156] S. Safavi, A. Hanani, M. Russell, P. Jancovic, and M. J. Carey, "Contrasting the effects of different frequency bands on speaker and accent identification," *Signal Processing Letters, IEEE*, vol. 19, no. 12, pp. 829–832, 2012.

[157] D. Cai, X. He, and J. Han, "Semi-supervised discriminant analysis," in *Proceedings of the International Conference on Computer Vision (ICCV'07)*, 2007.

[158] J. Goldberger, S. Roweis, G. Hinton, and R. Salakhutdinov, "Neighbourhood components analysis," in *Advances in Neural Information Processing Systems 17*, pp. 513–520, MIT Press, 2004.

[159] J. H. Friedman, "Regularized discriminant analysis," *Journal of the American Statistical Association*, vol. 84, no. 405, pp. 165–175, 1989.

[160] A. Robinson, "The British English Example Pronunciation (BEEP) dictionary," *Retrieved from World Wide Web: ftp://svrftp. eng. cam. ac. uk/pub/comp. speech/dictionaries/beep. tar. gz*, 1996.

[161] R. Weide, "The cmu pronunciation dictionary, release 0.6," 1998.

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

[163] G. Saon and H. Soltau, "A comparison of two optimization techniques for sequence discriminative training of deep neural networks," in *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2014*, pp. 5567–5571, 2014.

[164] C. Gaida, P. Lange, R. Petrick, P. Proba, A. Malatawy, and D. Suendermann-Oeft, "Comparing open-source speech recognition toolkits," tech. rep., Technical report, Project OASIS, 2014.

[165] S. Fitt and S. Isard, "Synthesis of regional english using a keyword lexicon.," 1999.

[166] A. Mohamed, G. E. Dahl, and G. E. Hinton, "Acoustic modeling using deep belief networks," *IEEE Transactions on Audio, Speech & Language Processing*, vol. 20, no. 1, pp. 14–22, 2012.

[167] A. Graves, A.-r. Mohamed, and G. Hinton, "Speech recognition with deep recurrent neural networks," in *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*, pp. 6645–6649, IEEE, 2013.

[168] R. Sriranjani, S. Umesh, *et al.*, "Investigation of different acoustic modeling techniques for low resource indian language data," in *Communications (NCC), 2015 Twenty First National Conference on*, pp. 1–5, IEEE, 2015.

[169] H. Sak, A. Senior, and F. Beaufays, "Long short-term memory based recurrent neural network architectures for large vocabulary speech recognition," *arXiv preprint arXiv:1402.1128*, 2014.

[170] A. Graves, N. Jaitly, and A.-r. Mohamed, "Hybrid speech recognition with deep bidirectional LSTM," in *Automatic Speech Recognition and Understanding (ASRU), 2013 IEEE Workshop on*, pp. 273–278, IEEE, 2013.

[171] G. H. Golub and C. F. Van Loan, "Matrix computations. 1996," *Johns Hopkins University, Press, Baltimore, MD, USA*, pp. 374–426, 1996.

[172] T. Li, S. Zhu, and M. Ogihara, "Using discriminant analysis for multi-class classification: an experimental investigation," *Knowledge and information systems*, vol. 10, no. 4, pp. 453–472, 2006.

[173] B. E. Boser, I. M. Guyon, and V. N. Vapnik, "A training algorithm for optimal margin classifiers," in *Proceedings of the fifth annual workshop on Computational learning theory*, pp. 144–152, ACM, 1992.

[174] L. Bottou and C.-J. Lin, "Support vector machine solvers," *Large scale kernel machines*, pp. 301–320, 2007.

[175] J. Mercer, "Functions of positive and negative type, and their connection with the theory of integral equations," *Philosophical transactions of the royal society of London. Series A, containing papers of a mathematical or physical character*, pp. 415–446, 1909.

[176] F. Melgani and L. Bruzzone, "Classification of hyperspectral remote sensing images with support vector machines," *Geoscience and Remote Sensing, IEEE Transactions on*, vol. 42, no. 8, pp. 1778–1790, 2004.

[177] C. Hsu and C. Lin, “A comparison of methods for multiclass support vector machines,” *IEEE Transactions on Neural Networks*, vol. 13, no. 2, pp. 415–425, 2002.

[178] J. A. Gualtieri and R. F. Cromp, “Support vector machines for hyperspectral remote sensing classification,” in *The 27th AIPR Workshop: Advances in Computer-Assisted Recognition*, pp. 221–232, International Society for Optics and Photonics, 1999.

[179] J. C. Platt, N. Cristianini, and J. Shawe-Taylor, “Large margin dags for multiclass classification,” in *Advances in Neural Information Processing Systems 12, [NIPS Conference]*, pp. 547–553, 1999.

[180] T. P. Minka, “Algorithms for maximum-likelihood logistic regression,” 2003.

[181] L. R. Rabiner and B.-H. Juang, “An introduction to hidden Markov models,” *ASSP Magazine, IEEE*, vol. 3, no. 1, pp. 4–16, 1986.

[182] R. Kuhn, J. Junqua, P. Nguyen, and N. Niedzielski, “Rapid speaker adaptation in eigenvoice space,” *IEEE Transactions on Speech and Audio Processing*, vol. 8, no. 6, pp. 695–707, 2000.

[183] M. Ferras, C. C. Leung, C. Barras, and J.-L. Gauvain, “Constrained MLLR for speaker recognition,” in *Acoustics, Speech and Signal Processing, 2007. ICASSP 2007. IEEE International Conference on*, vol. 4, pp. IV–53, IEEE, 2007.

[184] Y. Li, H. Erdogan, Y. Gao, and E. Marcheret, “Incremental on-line feature space MLLR adaptation for telephony speech recognition.,” in *INTERSPEECH*, 2002.

[185] D. Povey and K. Yao, “A basis method for robust estimation of constrained MLLR,” in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP 2011*, pp. 4460–4463, 2011.

[186] S. Matsoukas, R. Schwartz, H. Jin, and L. Nguyen, “Practical implementations of speaker-adaptive training,” in *DARPA Speech Recognition Workshop*, 1997.

[187] D. Povey, H. J. Kuo, and H. Soltau, “Fast speaker adaptive training for speech recognition,” in *INTERSPEECH 2008, 9th Annual Conference of the International Speech Communication Association*, pp. 1245–1248, 2008.

Appendix A

A.1 Dimension reduction techniques

In this section two techniques for dimensionality reduction that are widely used in the field of pattern recognition are introduced. These algorithms will be later used in our accent space visualisation technique introduced in Section 2.5.

A.1.1 Principal Components Analysis (PCA)

PCA is a dimension reduction approach which is often used before applying a supervised learning or classification algorithm. It applies an orthogonal projection of the data onto a lower dimensional space, such that the variance of the projected data is maximized [74].

Given a data set with N observations each of dimension D, the first stage is to normalise these observations with respect to their mean and variance. Given a data matrix O with the normalised observation feature vectors, o_i, along its columns, our goal is to project the data to space with dimension p where $p < D$.

Eigenvalue decomposition is applied to the covariance matrix S of the data set O. This technique decomposes a matrix S into a product of an orthogonal matrix of eigenvectors V and a diagonal matrix of non-negative eigenvalues Λ as shown in Equation A.1. In this chapter, the transpose operation is denoted by symbol '$\,'\,$'.

$$S = V\Lambda V' \tag{A.1}$$

In PCA, the eigenvectors corresponding to the p largest eigenvalues are selected, and form an p-dimensional space called the principal space [171].

For the dimension reduction applications, the data matrix O is then projected into this principal space with a much lower dimensions [75].

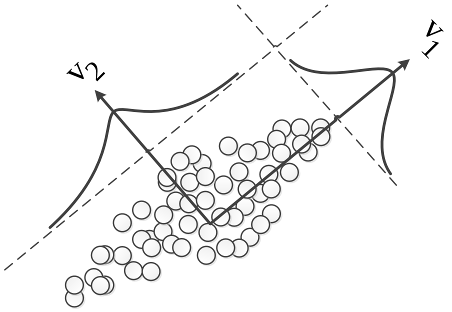

Fig. A.1 *PCA projection into dimensions that maximise the variance of the projected data*

Figure A.1 shows an example of dimension reduction using PCA, where the data is projected into $p = 2$ dimensions, namely v_1 and v_2 with the highest variance. In this section we explain how EM can be used in the PCA process.

One property of PCA is that it only finds the dimensions of maximum variance of the data, and is not concerned with separating any classes in the data. In the next section we explain how EM algorithm can be used to extract the principal components in the PCA.

A.1.2 EM algorithm for PCA (EM-PCA)

In the previous section we explained how we applied eigenvalue decomposition to the data covariance matrix to compute the p principal components in the PCA.

However, explicit calculation of data covariance matrix is a computationally expensive procedure. An alternative is to use the EM iterative approach to find a subspace constructed from the leading p eigenvectors without the need for direct calculation of the sample covariance matrix [77].

Given a data matrix, O, with the original observation feature vectors, o_i, along its columns and a matrix of unknown projections Z, matrix E can be defined such that its columns correspond to the p principal components. Lets assume that the corresponding columns of Z and O are set to z_i and o_i respectively.

In the EM algorithm the values of the unknown projections Z has to be found. Hence, in the expectation step, we assume that the principal subspace orientation E is known, and we estimate the values of unknown projections Z by projecting the observation data matrix O to the current subspace E.

$$Z = (E'E)^{-1}E'O \tag{A.2}$$

In the maximization step, we consider values of unknown projections Z are known, and try to maximize the expected joint likelihood of the estimated projection Z and the observation O through finding the subspace orientation (updated estimate of E) that minimizes the absolute value of the reconstruction error for all the observations O. In other words, for each observation feature vector o_i, a unique pair of (o_i, z_i) is found to minimize the value of the reconstruction error defined by $||Ez_i - o_i||$, under the assumption that z_i lies in the current principal subspace E and o_i is located in the original observation subspace defined by O.

$$E = OZ'(ZZ')^{-1} \tag{A.3}$$

A.1.3 Linear Discriminant Analysis (LDA)

Just like PCA, LDA tries to reduce the dimensionality of the data. Unlike PCA which is only concerned with maximising the variance, the goal of LDA is to project the data to a subspace that gives maximum separation between the means of the projected classes and minimise the variance within each projected class. LDA assumes that the data corresponding to the individual classes has a Gaussian distribution, and takes advantage of the class labels to project the data to a dimension that is most useful for a classifier. Given a set of observation feature vectors from different classes, LDA finds new orthogonal axes that can maximize the between-class covariance and minimize the intra-class covariance. These axes are therefore expected to be better in discriminating between different classes.

The intuition behind the LDA algorithm is similar to PCA, but instead of applying eigenvalue decomposition to the data covariance matrix, it applies it to matrix comprising of the ratio of the between-class covariance, S_b, and the within-class covariance S_w matrix shown by Equation A.4.

$$\mathbb{S} = S_b S_w^{-1} \tag{A.4}$$

In this way it gets a decomposition just like in PCA of the form $\mathbb{S} = V \Lambda V'$. Here, the eigenvalue represent, the ration of the between-class to within-class variance along the direction of the corresponding eigenvector. Thus, if the goal is to project the data into a p-dimensional space. only the p eigenvectors corresponding to the p largest eigenvalues are selected.

LDA, was first proposed by Fisher [73]. The aim of a two-class LDA is to find a projection vector that projects the data from a high-dimensional feature space to a one-dimensional space that best discriminate among the classes. An example of a two-class LDA is shown by Figure A.2, where the data is projected to the dimension that maximizes the separation between the projected class means and minimizes the variance within each projected class. This Figure also shows that the principal LDA dimension is not the same as the principal PCA dimension.

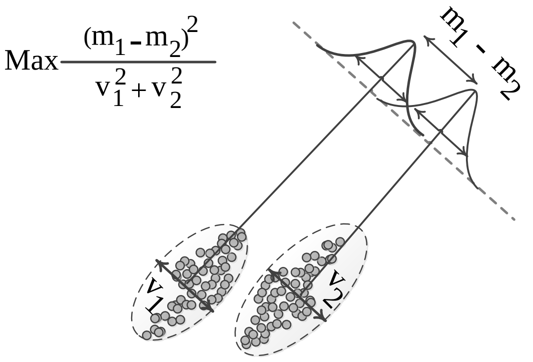

Fig. A.2 *An example of a two-class LDA*

Given the Fisher criterion, $J(\psi)$, along the space direction, ψ, the LDA optimization problem aims to find the vector ψ that maximises the ratio given by Equation A.5. Values of S_b and S_w are given by Equations A.6 and A.7 respectively. The total number of classes is represented by R, and the number of samples for each class r is denoted by n_r, the mean value of all samples is shown by $\bar{w}$, and the mean of feature vectors for each class r is given by $\bar{w_r}$.

$$J(\psi) = \frac{\psi' S_b \psi}{\psi' S_w \psi} \tag{A.5}$$

$$S_b = n_r \sum_{r=1}^{R} (w_r - \bar{w})(w_r - \bar{w})' \tag{A.6}$$

$$S_w = \sum_{r=1}^{R} \frac{1}{n_r} \sum_{i=1}^{n_r} (w_i^r - \bar{w_r})(w_i^r - \bar{w_r})' \tag{A.7}$$

Equation A.5 shows the statement of the criterion that LDA aims to maximize and can be solved through eigenvalue decomposition. The transformation, ψ, can be obtained by solving the generalized eigenvalue problem given by Equation A.8. The rows of the transformation matrix are equal to the $R-1$ eigenvectors corresponding the $R-1$ largest eigenvalues. The upper limit of the rank of S_w and S_b are respectively $(n_r - R)$ and $(n_r - 1)$ [75, 172].

$$S_b \psi = \lambda S_w \psi \tag{A.8}$$

Appendix B

B.1 Classification techniques

Each AID system involves in transforming a speaker's utterance into a supervector containing accent related patterns. Classification techniques are applied in the supervector space, to identify the speaker's accent.

In the AID experiments reported in this work, Support Vector Machine (SVM) was applied for classification of these accent-dependent supervectors. Logistic Linear Regression (LLR) is applied to fuse the SVM scores from multiple AID systems to achieve a better classification result. In this section both SVM and the LLR techniques are introduced.

B.1.1 Support Sector Machine (SVM)

Support Vector Machines (SVMs) were introduced by Vapnick and colleges in [50, 51, 173] as a discriminative data classification algorithm. SVM is primarily a binary classification method. Given a labelled training data with N input vectors, lets assume each data vector o_i of dimension D corresponds to a target class label $y_i \in \{1, -1\}$.

The goal of SVM is to find a hyperplane $w.o_i + b = 0$ that linearly separates the data samples into two classes and maximizes the distance between the hyperplane and the support vectors. The separating hyperplane (or decision boundary) is defined in terms of its normal vector w (perpendicular to the hyperplane) and bias value b.

An example of a two-dimensional classification problem is shown in Figure B.1, where the solid line represents the optimal separating hyperplane of form $w.o_i + b = 0$. The support vectors are the closest data points to this separating hyperplane (located on the dotted lines). The positive class (target class) is defined by support vectors that are located on the hyperplane $w.o_i + b = 1$ and the negative class (background class) is defined by support vectors that are located on the hyperplane $w.o_i + b = -1$. Given the target class labels y_i for each data sample o_i, the full SVM constraint can be summarised as shown in Equation B.1.

$$y_i(w.o_i + b) \geq 1 \tag{B.1}$$

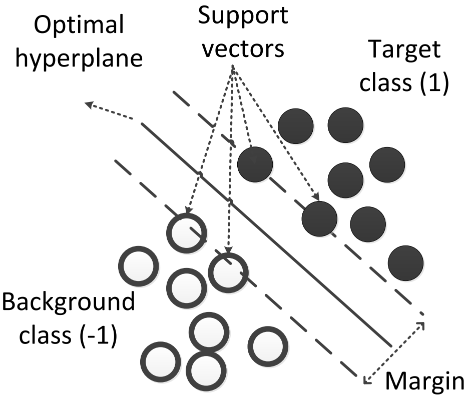

Fig. B.1 *An example of a two dimensional linearly separable problem addressed by SVM*

Different values of bias (e.g. b_1 and b_2) correspond to parallel hyperplanes, with distance $\mathbb{D} = |b_1 - b_2|/||w||$ between them. Thus, the distance between the hyperplanes defined by the support vectors of the positive and negative classes will be equal to $\mathbb{D} = 2/||w||$. The perpendicular distance between the optimal separating hyperplane and the support vectors is defined as the SVM margin. Maximising the margin (gap) between these two parallel hyperplanes is equivalent to minimizing $||w||$ (or $\frac{1}{2}||w||^2$) subject to the constraint denoted by Equation B.1. The new test vector o, can be classified into two classes according to the sign of the classification $f(o)$ defined by Equation B.2. For positive values of $f(o)$ data belongs to the target class, and for negative values of $f(o)$ data belongs to the background class.

$$f(o) = w.o + b \tag{B.2}$$

In SVM classification summation of distances from the support vectors to the decision hyperplane has to be maximised. Thus, the primal formulation for the linear SVM optimization problem can be summarised as shown by Equation B.3. This is a convex quadratic programming problem with D variables subject to linear constraints, and it has a unique minimum.

$$\begin{aligned} &\textit{Minimimize the objective function } \frac{1}{2}||w||^2 \\ &\textit{Subject to constraints } y_i(w.o_i+b)-1 \geq 0 \textit{ for } i=1,...,N \end{aligned} \tag{B.3}$$

To solve this optimization problem, a Lagrangian multiplier α_i is introduced for each constraint. Thus the primal form of the SVM can be represented as shown by Equation B.4.

$$L_P \equiv \frac{1}{2}||w||^2 - \sum_{i=1}^{N} \alpha_i(y_i(w.o_i+b)-1) \textit{ where } \alpha_i \geq 0 \tag{B.4}$$

One way to find w and b that maximize the prime Equation B.4, can be to differentiate L_P with respect to w and b separately and set the derivative to zero as shown by Equations B.5 and B.6.

$$\frac{\partial L_P}{\partial w} = 0 \implies w = \sum_{i=1}^{N} \alpha_i y_i o_i \tag{B.5}$$

$$\frac{\partial L_P}{\partial b} = 0 \implies \sum_{i=1}^{N} \alpha_i y_i = 0 \tag{B.6}$$

It is possible to rewrite the SVM's primal form (L_P) in form of a dual formulation (L_D) using the Lagrangian multiplier α_i and substituting the value of w (Equation B.7). The dual formulation is a convex quadratic programming problem with N variables, where N is the number of data samples. The solution of the dual form is now depend on dot products of input vectors which can be written in form of a Linear Kernel function $k(o_i,o_j)$ as shown by Equation B.8.

$$\begin{aligned} &\textit{Maximize the objective function } L_D \equiv \sum_{i=1}^{N} \alpha_i - \frac{1}{2}\sum_{i,j=1}^{N} \alpha_i\alpha_j y_i y_j o_i.o_j \\ &\textit{Subject to constraints } \alpha_i \geq 0 \textit{ and } \sum_{i=1}^{N} \alpha_i y_i = 0 \end{aligned} \tag{B.7}$$

$$o_i.o_j = k(o_i,o_j) \tag{B.8}$$

After calculation of the Lagrange multipliers in the dual form, the linear SVM classifier for a new input vector o can be represented by Equation B.9.

$$f(o) = \sum_{i=1}^{N} \alpha_i y_i k(o, o_i) + b \tag{B.9}$$

Up to this point, we described how the SVM classification algorithm can be applied to a linearly separable problem. In many cases the data set is not linearly separable. To deal with this linear inseparability, SVMs use a kernel function for mapping the original observations into a higher dimensional space $o \to \phi(o)$, using a suitable non-linear transformation function $\phi(.)$. In this high-dimensional space it is hoped that data points are linearly separable, and a linear classifier can be used to address a linearly inseparable problem in the original low-dimensional space [75, 174].

Using a 'kernel trick', it is possible to discard the transformation $\phi(.)$ of data points to the higher dimensional subspace, and only compute their pairwise inner dot products in the high dimensional subspace using a kernel function. To confirm the convergence of the quadratic programming problem for optimisation of the dual formulation defined in Equation B.7, kernel functions must satisfy Mercer conditions [75, 175].

Using the kernel concept, the SVM classification function in Equation B.9 can be written as defined by Equation B.10. As mentioned earlier, for test vector o sign of the function $f(o)$ defines whether the data vector belongs to the target or background class.

$$f(o) = \sum_{i=1}^{N} \alpha_i y_i k(\phi(o), \phi(o_i)) + b \tag{B.10}$$

Depending on the nature of data set other types of kernel functions are used to address the classification problem. In one of the classification problems we applied an SVM classification using a correlation distance kernel. The correlation kernel is shown in Equation B.11, where $\bar{O}$ is the sample mean of a $D \times N$ dimensional data matrix O and the value of parameter γ is obtained experimentally (in our experiment $\gamma = 1$). The correlation kernel is constructed based on the correlation matrix, $corr(o_i, o_j)$, shown in Equation B.12.

$$k(o_i, o_j) = e^{-\gamma(1 - corr(o_i, o_j))} \tag{B.11}$$

$$corr(o_i, o_j) = \frac{(o_i - \bar{O})(o_j - \bar{O})}{\sqrt{\sum_{i=1}^{N}(o_i - \bar{O})'(o_i - \bar{O})}\sqrt{\sum_{j=1}^{N}(o_j - \bar{O})'(o_j - \bar{O})}} \tag{B.12}$$

SVMs were initially designed for binary classification tasks. However, it is possible to extend their application to the multi-class problem. This can be achieved using the 'one against all' or 'one against one' approach.

In the 'one against all' classification approach, for a C-class classification problem, C binary SVM classifiers are constructed. Each SVM classifier is trained to distinguish one class from the remaining $C-1$ classes. This is done by assuming class label $y_i = 1$ for data points of one class (target class), and class label $y_i = -1$ for the data points that belong to the remaining classes (background class). During the testing, the new data point belongs to the class that gives the largest positive value of the decision function B.10 [50, 176, 177].

Another method to address the multi-class problem is to apply the 'one against one' approach, during which a comparison is made between all possible pair of classes. Given a test data point, each pair-wise classification gives one vote to the winning target class, and the class with most votes is considered as the target class for the test data point and the remaining classes will identified as imposters [178]. Studies showed that the 'one against all' approach can achieve a comparable results to the 'one against one' classification approach, while reducing the computational cost [177, 179]. Thus, the 'one against all' is chosen in this work.

B.1.2 Logistic Linear Regression (LLR)

A popular approach to classification is to evaluate several different classifiers on a task and then to combine their outputs.

In some cases it is observed that classifiers are complementary. In other words, when one classifier classifies a sample incorrectly the other classifier might classify it correctly. Thus, it can be useful to combine the outputs of the several complementary classification systems. This is referred to as system fusion. A number of different approaches to fusion have been proposed, of which the most popular is based on Logistic Linear Regression (LLR) [52, 53]. Basically, LLR estimates the fusion weights from multiple classifiers and through calibrating the outputs from several systems it aims to improve the discriminative power of the fused system.

Given a training set $o_1, o_2, ..., o_N$, suppose that each input vector o_n corresponds to a target class value $y_n = c$ where the target class values are denoted by $c \in \{1, 2, ..., C\}$. Let s^i_{nc} be the score generated for the n-th vector from the i-th subsystem for the c-th target class. The goal of score fusion is to combine the scores derived from L subsystems.

Using linear regression the score fusion technique is defined by Equation B.13. Here, the linear combination of scores from different subsystems is denoted by $\bar{s}_{nc}$, the weight value of

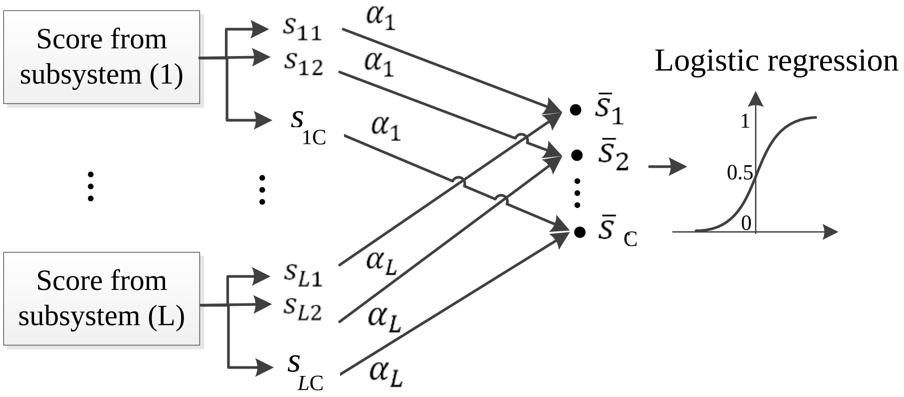

Fig. B.2 *Fusion the scores from L subsystems with C classes using LLR approach*

the i-th subsystem is represented by the regression coefficient α^i, and the bias value is shown by β_c.

$$\bar{s}_{nc} = \beta_c + \sum_{i=1}^{L} \alpha^i s^i_{nc} \tag{B.13}$$

Equation B.13 can be represented in matrix form (Equation B.14), using α and s_{nc} as shown in Equation B.15 and Equation B.16 respectively.

$$\bar{s}_{nc} = \beta_c + \alpha^{'} s_{nc} \tag{B.14}$$

$$\alpha = [\alpha^1, \alpha^2, ..., \alpha^L] \tag{B.15}$$

$$s_{nc} = [s^1_{nc}, s^2_{nc}, ..., s^L_{nc}] \tag{B.16}$$

However, a more effective score fusion method is achieved by using the LLR technique which is defined by Equation B.14 [180].

$$p(y_n = c|o_n) = \frac{exp(\bar{s}_{nc})}{\sum_{\tilde{c}=1}^{C} exp(\bar{s}_{n\tilde{c}})} \tag{B.17}$$

The goal is to estimate the weight of each subsystem α^i, and class dependent bias value β_c by maximising the likelihood function defined by Equation B.18. The bias value is estimated such that a global decision threshold can be applied to all classes. The conjugate gradient descent optimisation technique is used to solve this problem [180].

$$Q(\alpha^i, \beta_c) = \sum_{n=1}^{N} \sum_{c=1}^{C} \tau_{nc} log(\frac{exp(\bar{s}_{nc})}{\sum_{\tilde{c}=1}^{C} exp(\bar{s}_{n\tilde{c}})}) \tag{B.18}$$

The training label $\tau_{nc} \in \{0, 1\}$ indicates whether data point o_n belongs to target class c or alternatively $y_n = c$ as shown by Equation B.19.

$$\tau_{nc} = \begin{cases} 1 & if \;\; y_n = c \\ 0 & otherwise \end{cases} \tag{B.19}$$

Figure B.2 shows how the scores from L subsystems are fused from L parallel subsystems each with scores from C classes. The final score for each class is denoted by $\bar{s}_{nc}$ which is found after the weighted summation of corresponding scores from all subsystems [52, 53]

Appendix C

C.1 Markov Chains

In this section we start by introducing Markov chains and then we use this background to present the HMMs. Finally, we describe three problems that should be addressed by HMMs before they can be used in an ASR system.

Markov chains were introduced by Andrei Markov in 1906. Markov chains model a discrete random process with an assumption that at each time instance the probability of the random variable depends only on the value of the Markov chain at the previous time. This assumption uses small amount of memory to model dynamic data sequences. A Markov chain can be represented by a finite set of states $\{s_1,...,s_N\}$, such that each of these discrete values corresponds to a state in the Markov chain $q_t \in \{s_1,...,s_N\}$, where N is the total number of states. In an observable Markov model, each state corresponds to an observable event. In other words, the observable event sequence has a one-to-one correspondence to the state sequence $Q = \{q_1,q_2,...,q_T\}$ at each given time t. The process starts from one state and transitions successively to other states. A probability function is assigned to every transition. Given a sequence of random variables, the transition probability a_{ij} between states s_i and s_j in a Markov chain is equal to the probability of transitioning from the state i at time $t-1$ to the state j at time t as shown by Equation C.1.

$$a_{ij} = p(q_t = s_j | q_{t-1} = s_i), \;\; i,j = 1,2,...,N \tag{C.1}$$

The transition probability matrix A is defined by Equation C.2 and satisfies the condition denoted by Equation C.3.

$$A = [a_{ij}], \;\; where \;\; a_{ij} \geq 0 \tag{C.2}$$

$$\sum_{j=1}^{N} a_{ij} = 1, \;\; i = 1,2,...,N \tag{C.3}$$

A simple two state Markov chain is illustrated in Figure C.1.

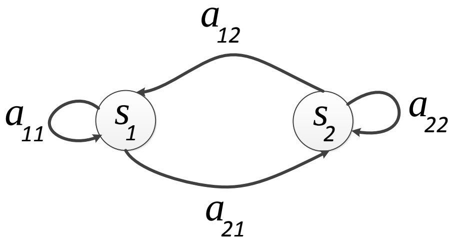

Fig. C.1 *A simple 2-state Markov chain*

Given the state transition probabilities, the state occupation probability is defined by Equation C.4 and computed recursively through Equation C.5.

$$p_j(t) = p[q_t = s_j] \tag{C.4}$$

$$p_i(t+1) = \sum_{j=1}^{N} a_{ij} p_j(t) \tag{C.5}$$

The probability of an arbitrary observation sequence of a Markov chain is denoted by Equation C.6 and it is computed by multiplying the initial state occupation probability π_{q_1} at time $t = 1$ by transition probabilities $a_{q_t q_{t+1}}$ computed by traversing throughout the state sequence Q.

$$p_Q = \pi_{q_1} \prod_{t=1}^{T-1} a_{q_t q_{t+1}} \tag{C.6}$$

Further details on Markov chains and their parameters can be found in [3, 181].

Appendix D

D.1 Acoustic model adaptation techniques for GMM-HMMs

This section describes three alternative acoustic model adaptation techniques, namely model adaptation using eigenvoice approach, using Constrained MLLR (CMLLR), and using Speaker Adaptive Training (SAT).

D.1.1 Model adaptation using eigenvoice

The eigenvoice approach constraints a new speaker model to be a linear combination of a set of reference speaker models and can perform well using small amount of adaptation data.

Initially a set of R speaker-dependent (SD) HMMs are produced for speakers in the training set using an adaptation technique (e.g. MAP). For each speaker the mean vectors of output Gaussians are concatenated to form a supervector. A dimension reduction approach (e.g. PCA) is then applied to these supervectors and eigenvectors corresponding to the top K largest eigenvalues are selected. Given a small amount of adaptation data from the test speaker, the dimension reduction ensures that small number of parameters need to be estimated from this data to represent the new speaker. These selected eigenvectors are called 'eigenvoices', and each speaker in the eigenvoice space is represented by an eigenvector e_j where $j = \{0, 1, ..., R\}$, and the mean reference supervector is represented by eigenvector $e(0)$. A mechanism is needed to locate a speaker adapted HMM in the eigenspace. In order to locate a new speaker in the eigenspace, his or her Gaussian mixture mean supervector is represented by Equation D.1.

The weight vector $w(j)$ with $j = \{0, 1, ..., R\}$ is estimated by the ML eigenvoice decomposition (MLED) for the observation $O = \{o_1 ... o_t\}$ from the new speaker, where o_t is normalised observation vector at time t. This can be done as shown in Equation D.2 by iteratively optimizing the quadratic function, Q, to maximise the likelihood of observation, with respect to current model λ, by setting $\partial Q/\partial w(j) = 0$ where $\hat{\lambda}$ is the estimated model. This iterative process continues until the weight value $w(j)$ converges. For a given speaker, in each iteration a new model λ is obtained and for state s in a set of HMM's, and Gaussian m in a mixture Gaussian output distribution, the value of the occupation probability $\gamma_m^{(s)}(t) = p(m,s|\lambda,o_t)$ is updated. Here, eigenvoice j is denoted by $e(j) = [e_1^1(j),...,e_m^{(s)}(j),...]^T$, the inverse covariance and the adapted mean are shown by $C_m^{(s)-1}$, and $\mu_m^{(s)}$ respectively.

$$s = e(0) + \sum_{j=1}^{R} w(j)e(j) \tag{D.1}$$

$$Q(\lambda,\hat{\lambda}) = -\frac{1}{2}p(O|\lambda) \times \sum_s \sum_m \sum_t \gamma_m^{(s)}(t) f(o_t,s,m) \tag{D.2}$$

Where, $f(o_t,s,m)$ and $\mu_m^{(s)}$ is estimated as shown in Equations D.3 and D.4 respectively.

$$f(o_t,s,m) = -nlog(2\pi) - log|C_m^{(s)}| + [(\mu_m - o_t)^T C_m^{(s)-1}(\mu_m - o_t)] \tag{D.3}$$

$$\mu_m^{(s)} = \sum_{j=1}^{R} w(j)e_m^{(s)}(j) \tag{D.4}$$

In eigenvoice approach, similar to MLLR, parameters of both observed and unobserved Gaussians are updated during the adaptation process. However, emphasis on prior knowledge in eigenvoice approach is more significant than MLLR, and therefore it has fewer degrees of freedom and requires much less adaptation data from the new speaker. The former is constrained on the prior knowledge about the reference speakers' patterns of variation, and the latter is contained on the definition of regression class and the initial speaker-independent model on which the transformations are applied [182].

D.1.2 Model adaptation using CMLLR

Unlike MLLR, in Constrained MLLR (CMLLR) the model means and variances are estimated using the same transform. After applying the same constrained transformation, A_c, to the covariance matrix, Σ_{ik}, and mean vector, μ_{ik}, their updated values is denoted by Equations D.5 and D.6 respectively. This is equivalent to transforming the observation vectors o_t in a way that the vector at time t can be represented as shown in Equation D.7 [130].

$$\hat{\mu}_{ik} = A_c\mu_{ik} + b_c \tag{D.5}$$

$$\hat{\Sigma}_{ik} = A_c\Sigma_{ik}A_c^T \tag{D.6}$$

$$\hat{o_t} = A_c^{-1} o_t + A_c^{-1} b_c \tag{D.7}$$

Given the sufficient statistics, the maximum likelihood solution to this formula involves an iterative optimization and gives a similar performance to MLLR adaptation when same form of transformation matrix is used [127].

When a single transformation is used the model-space transform, CMLLR, is equivalent to the feature-space MLLR transform (fMLLR) [183]. The aim of the fMLLR is to discover a linear transform of an acoustic space, such that probability of test data given the speaker independent model is maximised. Unlike the CMLLR, the fMLLR transform is applied directly to the acoustic feature vectors without any extra computation; more details can be found in [184, 185].

D.1.3 Model adaptation using SAT

Given a training set containing utterances from R speakers, initially MLLR is applied such that the speaker-independent Gaussian mean vector μ is mapped to an estimate of the speaker-dependent model for each speaker $\hat{\mu_r}$ in the training set as shown in Equation D.8. Here A_r is a full matrix and b_r is a bias vector that compromise the speaker specific transformation W_r.

$$\hat{\mu_r} = A_r \mu + b_r \tag{D.8}$$

From a set of observation sequence for each speaker where x_r is the observation sequence from speaker r, the optimum set of HMM parameters, $\hat{\lambda}$, and the set of speaker transformations $\hat{W} = (\hat{W_1}, ..., \hat{W_R})$ are jointly estimated to maximize the likelihood of the training data as shown in Equation D.9.

$$(\hat{\lambda}, \hat{W}) = argmax_{\lambda,W} \prod_{r=1}^{R} p(x_r; W_r, \lambda) \tag{D.9}$$

Then the Gaussian means, variances, and mixture weights of these models are updated using the EM algorithm. As a result SAT reduces the error rate through estimating models with higher training set likelihoods and smaller new variances. However, SAT is not practical when the size of the training set is large, due to the per-speaker MLLR computation stage which has a high computational cost and large storage size[129]. In [186, 187], different variations of the SAT approach are described.

Appendix E

E.0.4 Shifted Delta Cepstral (SDC) features

The MFCC features are concatenated to the Shifted Delta Cepstral (SDC) features for the i-vector AID, In this section we describe the SDC features.

The SDC feature vectors $SDC(t)$ are constructed by concatenating delta Cepstral coefficients $\Delta c(t,i)$ calculated across multiple speech frames.

The SDC vectors are specified by four parameters in the following form '$N-d-P-k$', where the number of cepstral coefficients calculated at each frame is denoted by N, the time step for calculating the delta is defined by d. To construct the SDC feature vector the delta coefficients of k blocks are stacked, with time shift of P between successive blocks.

The SDC coefficients for a cepstral frame i at time t, are computed as denoted by Equation E.1.

$$\Delta c(t,i) = c(t+iP+d) - c(t+iP-d) \qquad where \;\; i = 0,1,2,...,k-1 \tag{E.1}$$

Equation E.2 describes how the SDC feature vector at time t is formed by stacking the delta cepstra coefficients.

$$SDC(t) = \begin{bmatrix} \Delta c(t,0) \\ \Delta c(t,1) \\ \vdots \\ \Delta c(t,k-1) \end{bmatrix} \tag{E.2}$$

Appendix F

F.0.5 ABI-1 phone set

Here is the phone set used in this work.

ah, sp, ax, ey, b, l, k, s, eh, p, t, ih, ng, m, n, d, aw, ae, r, oh, sh, uw, jh, v, z, ch, f, g, iy, ea, er, ao, ow, w, oy, uh, dh, ay, ia, th, zh, hh, ua

F.0.6 List of ABI-1 out of vocabulary words

In this work BEEP dictionary was used to creat a British English recogniser using WSJCAM0 and ABI-1 corpus. Here are the list of ABI-1 words which did not exist in the BEEP dictionary.

ALOW: ax l ow, B: b iy, C: s iy, CO-OPERATIONS: k ow oh p ax r ey sh n z, D: d iy, DOLBY: d oh l b iy, F: eh f, FIREBOOTERS: f ay ax b uw t ax z, FIREBOOTERS: f ay ax r b uw t ax z, FOOTBOATERS: f uh t b ow t ax z, FOOTBOOTERS: f uh t b uw t ax z, FREEBOATERS: f r iy b ow t ax z, FREEFOOTERS: f r iy f uh t ax z, FRUCTOSE: f r ah k t ow s, G: jh iy, H: ey ch, HEARD: hh er d, HEERED: hh ia d, HODE: hh ow d, HOID: hh oy d, HOW'D: hh aw d, HOWD: hh aw d, HOWED: hh ow d, HUDD: hh ah d, HURED: hh y ua d, ITEMISE: ay t ax m ay z, ITEMISED: ay t ax m ay z d, ITEMISES: ay t ax m ay m ay z ih z, K: k ey, L: eh l, M: eh m, MACK: m ae k, MINGLE-HANDED: m ih ng g l hh ae n d ih d, N: eh n, OARMEN: ao m ax n, OARMEN: ao r m ax n, OARSEMEN: ao z m ax n, OK: ow k ey, P: p iy, PORTUGESE: p ao ch uh g iy z, R: aa, R: aa r, RE-ROUTE: r iy r uw t, REC: r eh k, REC: r ih k, RECIRC: r iy s er k, RECO: r ih k ow, ROMEO: r ow m ow, S: eh s, SAIL-STEERING: s ey l s t ia r ih ng, SELF-HANDED: s eh l f hh ae n d ih d, SELF-STEERING: s eh l f s t ia r ih ng, SIL: sil, SINGLE-HEADED: s ih ng g l hh ae n d ih d, SISTER-SHIP: s ih s t ax sh ih p, SISTER-SHIP: s ih s t ax r sh ih p, SISTER-SHIPS: s ih s t ax sh ih p s, SISTER-SHIPS: s ih s t ax r sh ih p s, SIX-HANDED: : s ih k s hh ae n d ih d, T: t iy, TWELCE: t w eh l s, U: y uw, V: v iy, VILLIANS: v ih l ia n z, WAYPOINT: w ey p oy n t, WAYPORT: w ey p ao t, Y: w ay