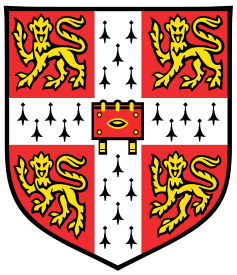
ALTA Project - Spoken Language Assessment and Learning

Improve Adaptation Performance of ASR to Non-Native Speakers



Junjie Pan

Department of Engineering
University of Cambridge
M.Phil in Machine Learning, Speech and Language Technology

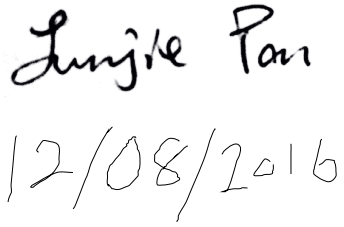
This dissertation is submitted for the degree of
Master of Philosophy

Hughes Hall August 2016

Declaration

I, Junjie Pan of Hughes Hall, being a candidate for the M.Phil in Machine Learning, Speech and Language Technology, hereby declare that this report and the work described in it are my own work, unaided except as may be specified below, and that the report does not contain material that has already been used to any substantial extent for a comparable purpose.

Word count: 11,166



Junjie Pan
August 2016

Acknowledgements

I would like to thank Prof. Mark Gales for his constant guidance and support throughout the year which has enabled me to finish this project on schedule. I also want to express my gratitude to Prof. Mark Gales for presenting me with the opportunity to work on the ALTA project and for providing valuable feedback and suggestions which helped shape our research goals.

A special appreciation to Dr. Kate Knill and Dr. Yu Wang for inspiring discussions and invaluable advice at various points during the past few months.

Finally, I am thankful for the support offered by Chunyang Wu, Chao Zhang, Andrey Malinin who provided me practical suggestions for troubleshooting in this project.

Abstract

This dissertation describes the work done in the ALTA MPhil project of automatic assessment of spoken English on improving the adaptation performance for non-native speakers. The performance of current *deep neural network*(DNN) based *automatic speech recognition*(ASR) systems are highly influenced by the mismatch between training and test data introduced by inter-speaker variabilities, especially for non-native speakers.

Accordingly, this project concentrates on the state-of-art DNN-based unsupervised speaker adaptation in acoustic modelling at speaker and language level respectively. Their performance to speakers with different *first languages*(L1s) is investigated and compared to *speaker adaptive training*(SAT) using CMLLR. For activation function adaptation, its performance is sensitive to the supervision WERs, and it does not work for datasets containing the same L1 as training data at both levels. Moreover, this adaptation shows the ability to achieve considerable WER reduction as the same degree as SAT under some conditions. At speaker level, the datasets should contain different L1s from training data, and have relatively low supervision WERs(approximately 45%). At language level, except for the requirements at speaker level, multi-L1 datasets are also needed. In terms of the comparison between the two levels, language-level adaptation show generally worse WERs than speaker-level in most cases, and is more susceptible to supervision WERs, but the advantage of language-level adaptation is that it is easier to obtain more adaptation data. Furthermore, by comparing crowd-sourced supervision with Hybrid-SI supervision, it is found that for single-L1 datasets, language-level adaptation does not work for low-level speakers(A1 and A2) regardless of the quality of the supervision. In addition, experiments on feature augmentation using i-vectors at speaker level indicate that this adaptation strongly relies on the i-vector extracting model, and does not work if the L1 in test data is unseen in training data. For the same L1 case, this approach shows slightly lower WERs by 0.05% and 0.08% for 10 and 30 dimensional i-vectors when compared with Hybrid-SI under CE training criterion.

Table of contents

List of figures

List of tables

Chapter 1

Introduction

Speech plays an irreplaceable role in delivering messages, exchanging information, and communication in humans' daily life. In that case, a number of researches have been carried out in using machines to provide a convenient ,cheap but high-quality way to process speech. Among a variety of speech issues, automatic speech recognition (ASR) [3], which aims to convert unknown speech waveforms into the corresponding orthographic text, is an extremely challenging but practical task.

In the past, *Guassian mixture model*(GMM) based *hidden Markov models*(HMMs) have been widely used in acoustic modelling [2]. Recently, with the rapid development of *deep neural networl*(DNN) theory and more powerful computing hardware, DNN-HMM systems have achieved substantial improvement in speech recognition compared to conventional GMM-HMM systems [13]. However, both GMM-HMM and DNN-HMM systems suffer from the performance degradation resulted from mismatches introduced by the speaker variability between training and test data on the acoustic conditions. For example, in English speech recognition, speakers may come from different countries and regions over the world, so that their English pronunciation is likely to be influenced by their *first languages*(L1s) and living environments. Moreover, the coherence and cohesion in speech data will also be effected by the speaker's English levels. These acoustic conditions are normally various in training and test data, and cause mismatches. Accordingly, ASR systems will show inconsistent performance due to these inter-speaker variabilities. Adaptation techniques is then proposed to reduce such problems. For GMM-HMM model adaptation, *maximum a posterior*(MAP) [10] and *maximum likelihood linear regression*(MLLR) [16] have been proved to be the two most effective methods [30]. However, these techniques cannot be directly used in discriminative DNNs. In that case, previous work has successfully used constraint MLLR(CMLLR) to implement *speaker adaptive training*(SAT) for DNN-HMM models, and achieved dramatical improvement of performance [19] [9]. However, SAT

using CMLLR is quite expensive as it is in a supervised fashion. Considering that, several state-of-art DNN-based unsupervised adaptation approaches on acoustic modelling will be investigated in this project.

1.1 Automatic Speech Recogniser (ASR)

ASR [3] acts as an essential role in current automated grader framework as shown in figure 1.1, which converts audio data into text data. The first step of ASR is to extract a number of acoustic features containing sufficient information from the speech as observations. The second step is to generate dictionaries, language model and acoustic model, which are used to infer the most possible word sequences (hypothesis). The dictionary, also referred as the lexicon, is used to map actual words in the vocabulary to some sub-units such as sub-words or phonemes. The language model is a probability distribution over sequences of words, containing the semantic and syntactic information of the utterance. The acoustic model is a mapping between acoustic features and sub-word units. In statistic speech recognition

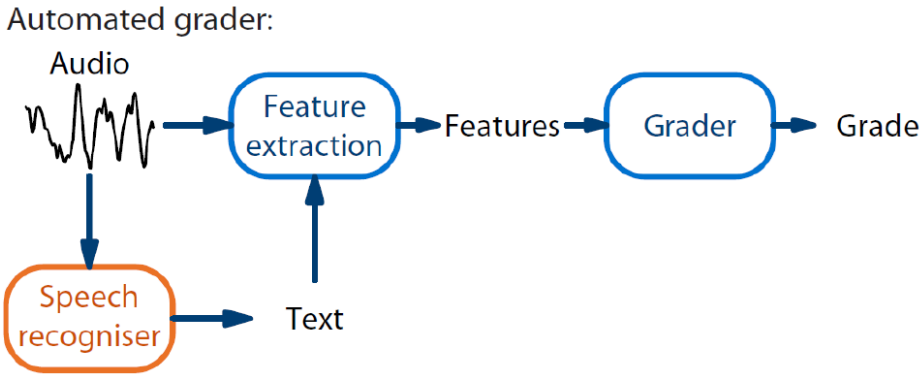


Fig. 1.1 Architecture of current automated grader framework

approach, Bayesian decision rule is applied as the decision criterion to determine the most possible hypothesis H given a sequence of observations $O = \{o_1, \cdots, o_N\}$

$$\begin{aligned} H &= \arg\max_H P(H|O) \\ &= \arg\max_H \left(\frac{p(O|H)P(H)}{p(O)} \right) \\ &\Rightarrow \arg\max_H p(O|H)P(H) \end{aligned} \tag{1.1}$$

where the prior of the hypothesis $P(H)$ comes from the language model, and the likelihood of observations $p(O|H)$ is derived from the acoustic model.

In ALTA project, the general framework of the ASR is a joint decoding system as shown in figure 1.2. It consists of two parts: the HMM-GMM system (Tandem), and the HMM-DNN system (Hybrid). The acoustic feature are perceptual linear predictive (PLP) features combined with Bottleneck (BN) features generated from a BN neural network. This project focus on the Hybrid part of the framework.

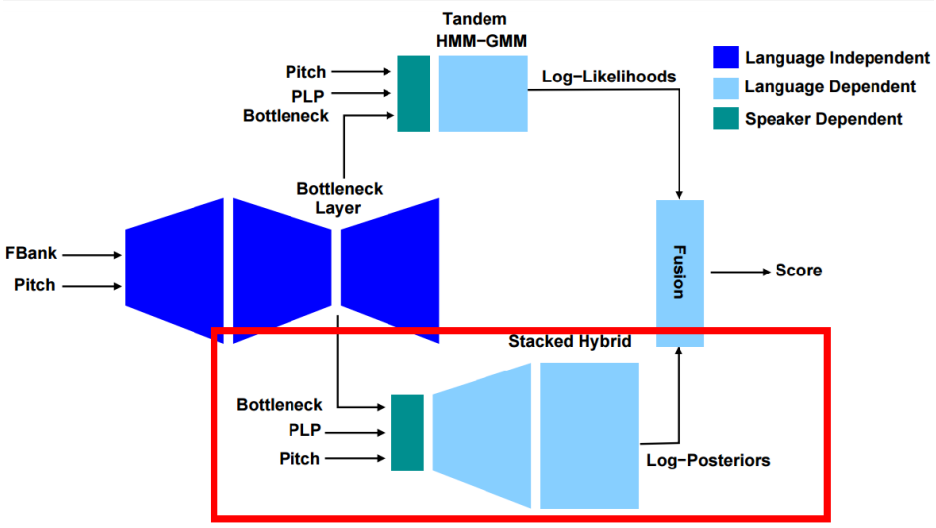


Fig. 1.2 Architecture of current joint decoding ASR system

1.2 Speaker Variability

The state-of-art ASR system are DNN-HMM based model and trained on the observations derived from the audio, which learns the information in the extracted features. Theoretically, well-trained DNN-HMMs should be independent from the acoustic conditions if the features only contain the inherent speech variability information. However, additional acoustic condition variabilities, such as inter-speaker differences (accents and language proficiency) and intra-speaker differences (speaking styles and emotion effects), are likely to distilled during feature extraction process in reality. Therefore, HMMs-based ASR are always trained on features consist of both speech variability information and acoustic condition variability information. In this case, the speaker variability is likely to cause mismatch issues between training and test data, and degrade the ASR performance.

Chapter 2

Acoustic Modelling

In this chapter, a brief description of different parts of the DNN-HMM speech recognition system is given.

2.1 Feature Extractions

Raw speech data are continuous waveforms, which are not suitable for speech recognition tasks. Therefore various schemes are implemented to convert continuous speech waveforms into discrete parametric vectors, which are referred as features. These features can be regarded as another form of representations that extract compact informations from the original speech waveforms.

- **Perceptual Linear Prediction (PLP) Features**

  The PLP feature [12] extraction process is

  1. *Spectrum Analysis*
     The power spectrum, the square of the magnitude, is obtained from the raw speech signals by sampling, windowing, and discrete Fourier-Transform. It is then processed by the Bark-frequency scaling.
  2. *Post Processing*
     Mel-frequency filter bank is applied to the power spectrum to extract PLP features, where the coefficients of Mel filter bank are scaled by the equal-loudness curve and the cubic root.
  3. *Linear Prediction (LP) and PLP Coefficients Calculation*
     Given the resultant spectrum of the post processing, an auto-correlation sequence

in time domain can be generated. It is then used to calculate the LP coefficients. Consequently, the PLP coefficients (also known as the Cepstral coefficients) are computed on the base of the LP coefficients.

- **Bottleneck (BN) Features**

  The BN features are extracted from a DNN architecture depicted in Figure 2.1, where the dimension of the BN layer is typically fewer than 100 [32]. The hidden layers in front of the BN layer are initialised as de-noising auto-encoder using layer-wise pre-training in unsupervised fashion [18]. After pre-training a stack of auto-encoders, a BN layer is connected to the output of the auto-encoder, and initialised with random weights. An extra hidden layer and a classification layer after the BN layer are trained on the same fashion. After that, the whole network is trained with supervision to estimate the context-dependent targets, where the cross-entropy (CE) criterion is applied. When finishing the training process, BN features are generated from the output of the BN layer.

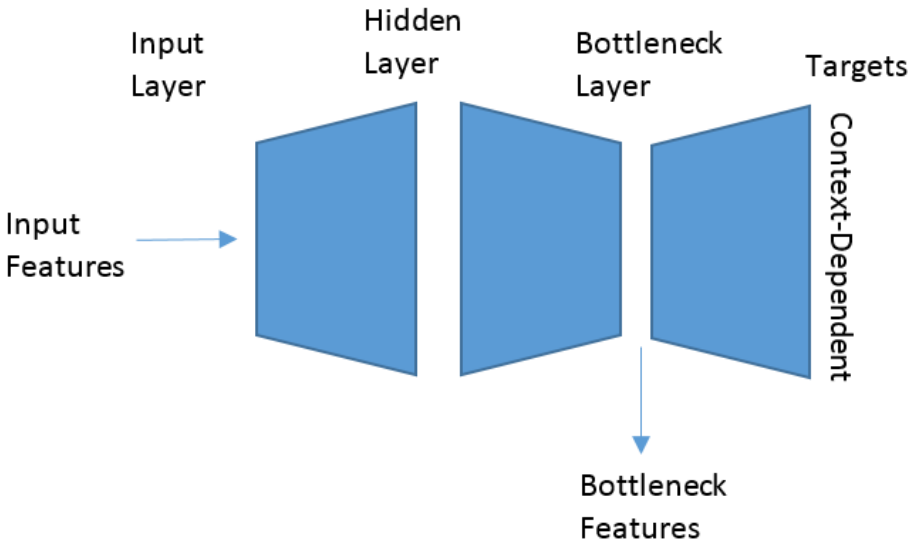


Fig. 2.1 The BN deep neural network architecture

2.2 Hidden Markov Models (HMMs)

As described in Chapter 1, HMMs are widely used in acoustic models, and proved to be the most successful technique in speech recognition [7]. This section will discuss the basic concept and some training algorithms for HMMs.

2.2.1 HMMs

An HMM is a finite state machine, where three key assumptions are given:

1. Markov Assumption.
   The transition probability between states are constant, and only dependent on current state.

2. State Conditional Independence Assumption.
   The probability of observed data are independent from each other when the state that generated it is given.

3. Accurate Representation Assumption
   The observations (features) are assumed to be accurate representations of the speech signal over the length of the frame.

In speech recognition, HMMs are implemented to handle a variable number of feature vectors from each speech unit. For N-state HMMs, there are $N-2$ emitting states, one non-emitting entry sate and one non-emitting exit state. The transition probability between any pair of states i and j is a_{ij}, and the transition happens at every input frame. For every emitting state, it generates the observations with probability $b_j(o)$. An example of the topology of HMMs is illustrated in Figure 2.2.

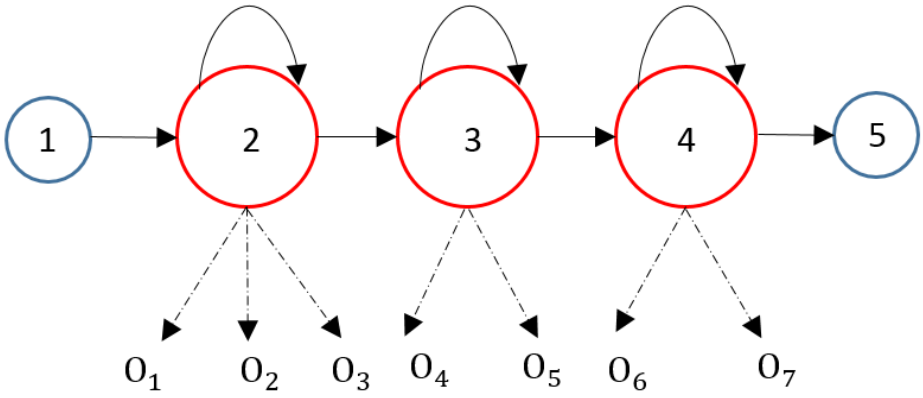


Fig. 2.2 The topology of a left-to-right HMMs with three emitting states

Let $\mathbf{O} = \{\mathbf{o}_1, \mathbf{o}_2, ..., \mathbf{o}_T\}$ be the observations (speech features) to a corresponding acoustic units such as a word or a phone. Each $\mathbf{o}_t$ $(0 <= t <= T)$ is a D-dimension feature vector. These observed speech features are generated from the emitting state 2,3,4 in Figure 2.2. Let $\mathbf{S} = \{\mathbf{s}(1), \mathbf{s}(2), ..., \mathbf{s}(T)\}$ be a sequence of states, and the joint probability of observations

and state-sequence is

$$p(\mathbf{O},\mathbf{S}|\lambda) = a_{s(0),s(1)}\prod_{t=1}^{T} b_{s(t)}(o_t)a_{s(t),s(t+1)} \tag{2.1}$$

where $\lambda = \{N,\{a_{ij}\},\{b_j(\cdot)\}\}$ represents a HMM, $\mathbf{s}(0)$ is always the entry state (state 1 in Figure 2.2), and $\mathbf{s}(T+1$ is always the exit state (state 5 in Figure 2.2). To be specific, a HMM contains the following parameters $\mathbf{M} = \{\pi,\{a_{ij}\},\{b_j\}\}$

- **Initial State Distribution** π

$$\pi_i = P(\mathbf{s}(1) = i), where 1 <= i <= N \tag{2.2}$$

  where π is constrained by

$$\sum_{i=1}^{N}\pi_i = 1, \pi_i \geq 0 \tag{2.3}$$

  where N is the number of states, and $\pi_1 = 1$.

- **State Transition Probability**$\{a_{ij}\}$
  a_{ij} represents the transition probability from state i to state j:

$$a_{ij} = P(\mathbf{s}(t+1) = j|\mathbf{s}(t) = i) \tag{2.4}$$

  where a_{ij} satisfies

$$\sum_{j=1}^{N} a_{ij} = 1, a_{ij} \geq 0 \tag{2.5}$$

- **State Output Probability** $\{b_j\}$

  The probability of an observation o_t at time t generated by the emitting state j is

$$b_j(\mathbf{o}_t) = p(\mathbf{o}_t|\mathbf{s}(t) = j) \tag{2.6}$$

When applying HMMs to speech recognition, it is necessary to consider issues about HMM parameters training and optimisation. These aspects will be discussed in the following sections.

2.2.2 Maximum Likelihood (ML) Training

ML training is a popular approach in learning HMM parameters. The aim of ML training is to find a set of parameters that maximize the likelihood of the observation sequence. In speech recognition, a transcription or hypothesis $\mathscr{H}$ is also needed as supervision. The mathematical expression is:

$$\hat{\mathbf{M}} = \arg\max_{\mathbf{M}} p(\mathbf{O}|\mathscr{H},\mathbf{M}) = \arg\max_{\mathbf{M}} \prod_{i=1}^{T} p(\mathbf{o}_i|\mathscr{H},\mathbf{M}) \tag{2.7}$$

Equation (2.7) is equivalent to maximize the log-likelihood function of $\mathbf{M}$ given $\mathbf{O}$:

$$L(\mathbf{M}) = logp(\mathbf{O}|\mathscr{H},\mathbf{M}) = \sum_{i=1}^{T} logp(\mathbf{o}_i|\mathscr{H},\mathbf{M}) \tag{2.8}$$

To maximize $L(\mathbf{M})$, we just need to differentiate it with respect to $\mathbf{M}$ and equate the result to 0. However, the hidden variables in HMMs make it impractical to optimize $\mathbf{M}$ in such a direct way. Luckily, there is a number of approximation methods that can be adopted to solve this problem, and Expectation-Maximization (EM) algorithm is one of them [4].

2.2.3 Expectation-Maximization (EM) algorithm

EM algorithm is a widely-used approach in statistical optimisation tasks. The basic concept of using EM in ML is to find a lower boundary of the log-likelihood function and do iterations to update model parameters to increase the boundary, so that the log-likelihood is also increased. Jensen's inequality is applied to find the lower boundary[1]. In order to make use of Jensen's inequality, $q(\mathbf{x})$, a distribution of the hidden state sequence is introduced to the log-likelihood function and yields

$$\begin{aligned} L(\mathbf{M}) &= logp(\mathbf{O}|\mathscr{H},\mathbf{M}) = log\sum_{\mathbf{x}} q(\mathbf{x})\frac{p(\mathbf{O},\mathbf{x}|\mathscr{H},\mathbf{M})}{q(\mathbf{x})} \\ &\geq \mathbb{E}\{logp(\mathbf{O},\mathbf{x}|\mathscr{H},\mathbf{M})|q(\mathbf{x})\} + H(q(\mathbf{x})) \end{aligned} \tag{2.9}$$

[1] Jensen's inequality: $f(\sum_{m=1}^{M}\lambda_m x_m) \geq \sum_{m=1}^{M}\lambda_m f(s_m)$ where $f(\cdot)$ is any concave function, and $\sum_{m=1}^{M}\lambda_m = 1, \lambda_m \geq 0, m = 1,\cdots,M$.

Where equality is obtained when $q(\mathbf{x}) = P(\mathbf{x}|\mathbf{O},\mathscr{H},\mathbf{M})$. $\mathbb{E}(f(\mathbf{x}))|_{g(\mathbf{x})}$ is the expectation of $f(\mathbf{x})$ with respect to $g(\mathbf{x})$.

$$\mathbb{E}(f(\mathbf{x})|g(\mathbf{x})) = \begin{cases} \sum_{\mathbf{x}} f(\mathbf{x})g(\mathbf{x}) & \text{if } g(\mathbf{x}) \text{ is discrete} \\ \int_{\mathbf{x}} f(\mathbf{x})g(\mathbf{x})d\mathbf{x} & \text{if } g(\mathbf{x}) \text{ is continuous} \end{cases} \tag{2.10}$$

$H(g(\mathbf{x}))$ is the entropy of $g(\mathbf{x})$

$$H(g(\mathbf{x})) = \begin{cases} -\sum_{\mathbf{x}} g(\mathbf{x})logg(\mathbf{x}) & \text{if } g(\mathbf{x}) \text{ is discrete} \\ \int_{\mathbf{x}} g(\mathbf{x})logg(\mathbf{x})d\mathbf{x} & \text{if } g(\mathbf{x}) \text{ is continuous} \end{cases} \tag{2.11}$$

Our goal is to reach the lower boundary, so the posterior $P(\mathbf{x}|\mathbf{O},\mathscr{H},\mathbf{M})$ is needed. However, it is impractical to compute it directly as one of its condition M is exact the set of parameters to be optimised. In this case, the model parameters and the posterior need to be updated iteratively. This updating process consist of two stages:

1. **Expectation**

   Given the current estimation of model parameters $M^{(k)}$, the lower boundary (2.9) can be obtained as:

$$\begin{aligned} L(M^{(k+1)}) &= logp(O|H,M^{(k+1)}) \\ &\geq \mathbb{E}\{p(O,x|H,M^{(k+1)})|P(x|O,H,M^{(k)})\} + H\left(P(x|O,H,M^{(k)})\right) \end{aligned} \tag{2.12}$$

   where $M^{(k+1)}$ is the parameters to be estimated. To find the new estimation $M^{(k+1)}$, an auxiliary function is obtained:

$$Q(M^{(k+1)},M^{(k)}) = \mathbb{E}\{logp(O,x|H,M^{(k+1)})|P(x|O,H,M^{(k)})\} \tag{2.13}$$

2. **Maximization**

   This auxiliary function obtained in expectation step is only a function of the new estimation $M^{(k+1)}$. Now it is possible to maximise the value of the the auxiliary function with respect to $M^{(k+1)}$. After getting the new model estimate, the expectation step can be repeated.

This iterative process ensures the value of the auxiliary function to increase, which guarantees that the log-likelihood in the next iteration will not decrease. The proof of that is:

$$\begin{aligned} & Q(M^{(k+1)}, M^{(k)}) \geq Q(M^{(k)}, M^{(k)}) \\ \Rightarrow & Q(M^{(k+1)}, M^{(k)}) + H\left(P(x|O,H,M^{(k)})\right) \geq Q(M^{(k)}, M^{(k)}) + H\left(P(x|O,H,M^{(k)})\right) \\ \Rightarrow & logp(O|H,M^{(k+1)}) \geq logp(O|H,M^{(k)}) \end{aligned} \tag{2.14}$$

It should be aware that EM algorithm is a local-optimum approach, and it will be influenced by the initialisation condition.

2.3 DNN-HMM Hybrid System

In a DNN-HMM hybrid system, the DNN produces a posterior for HMM states. The representation of each layer in DNN can be expressed as:

$$\mathbf{h}_l^m = f(\mathbf{W}_l \mathbf{h}_{l-1}^m + \mathbf{b}_l) \tag{2.15}$$

where m denotes the speaker m, $\mathbf{h}_l^m$ and $\mathbf{h}_{l-1}^m$ are the abstract representation at current layer l and previous layer $l-1$, f represents the activation function at layer l, $\mathbf{W}_l$ is the weight matrix at layer l, and $\mathbf{b}_l$ is the bias at layer l.

Consequently, an L-hidden-layer DNN model learns a non-linear function $u(\mathbf{x};\mathbf{M})$ from a set of training samples $\{(\mathbf{x}_t, \mathbf{s}_t)\}_{t=1}^T$, where $\mathbf{x}_t$ are training data and $\mathbf{s}_t$ are tied states at time t.

$$u(\mathbf{x};\mathbf{M}) = f_o(\mathbf{W}_o^\top f_L(\mathbf{W}_L^\top f_{L-1} \cdots f_l(\mathbf{W}_l^\top (\cdots f_1(\mathbf{W}_1 \mathbf{x})))) \tag{2.16}$$

where f_l is the activation function at l-th hidden layer, f_o is the output activation function, $\mathbf{W}_o$ is the weight matrix at the output layer. These hidden layer activation functions can take different forms, such as a sigmoid $f_l(c) = \frac{1}{1+exp(-c)}$, rectifying linear units(ReLU) $f_l(c) = max(0,c)$, or maxout units $f_l(c) = max(c_i), i = 1 \cdots M$. The output activation function f is a *softmax* to produce posterior distribution:

$$P_j(s|\mathbf{x};\mathbf{M}) = \frac{exp(\mathbf{W}_{oj}^\top f_L)}{\sum_{j'} exp(\mathbf{W}_{oj'}^\top f_L)} \tag{2.17}$$

where $\mathbf{M} = \{\mathbf{W}_1, \mathbf{W}_2, \cdots, \mathbf{W}_L, \mathbf{W}_o\}$ are model parameters, W_{oi} denotes the weight matrix at node j in the output layer.

The output of node j in each layer can be expressed as:

$$\mathbf{y}_j = P_j(\mathbf{s}|\mathbf{o}_t) = f_j(\mathbf{a}_j) \tag{2.18}$$

where $\mathbf{y}_j$ denotes the output of the current layer. $\mathbf{a}_j = \mathbf{W}_{lj}\mathbf{x}_l + \mathbf{b}_l$. Where $\mathbf{w}_{lj}$ denotes the weights for inputs to node j in layer l. For input layer, $\mathbf{x}_l = \mathbf{x}_{in}(t)$; otherwise, $\mathbf{x}_l = \mathbf{y}_{l-1}$, where $\mathbf{y}_{l-1}$ is the output from the previous layer.

In speech recognition tasks, DNNs are trained to optimise an objective function with error back-propagation approach. Normally, at frame level, cross-entropy (CE) loss function is adopted as the objective using stochastic gradient descent (SGD) optimisation method.

2.3.1 Cross-Entropy (CE) Training

The *cross-entropy* (CE) criterion is common used for DNN-based acoustic model training at frame level. It measures the distance between the estimated probability distribution and the target probability distribution given a set of training labels. The mathematical expression for CE is:

$$F_{CE} = -\sum_{t=1}^{T}\sum_{k=1}^{K}\hat{\mathbf{y}}_k(t)log\frac{\mathbf{y}_k(t)}{\hat{\mathbf{y}}_k(t)} \tag{2.19}$$

where k denotes the node k at a DNN layer, $\hat{\mathbf{y}}_k(t)$ denotes the target label of node k at time t, $\mathbf{y}_k(t)$ is the output of node k.

Stochastic gradient descent (SGD) approach is then used to minimise the CE loss function. The derivative of F_{CE} with respect to the activation function is:

$$\frac{\partial F_{CE}}{\partial f_k(\mathbf{a}_j)} = \mathbf{y}_k(t) - \hat{\mathbf{y}}_k(t) \tag{2.20}$$

It acts as the error propagation from the output layer k. In the training process of Hybrid *speaker-independent* (SI) DNN model, $\hat{y}_k(t)$ is obtained from the reference transcription. In the DNN-based unsupervised adaptation process, $\hat{y}_k(t)$ comes from the hypothesis from the output of the previous decoding pass.

2.3.2 Sequence Training

The DNN-based acoustic model trained from CE criterion is normally further refined with some sequence-level criteria. This section will introduce two most common sequence training criteria - *maximum mutual information* (MMI) and *minimum phone error* (MPE).

Maximum Mutual Information (MMI)

Maximum Mutual Information (MMI) is based on the mutual information, the posterior $P(H|O)$, between the correct word sequence (transcription) and the observations [1] [14]. It can be expressed as:

$$F_{MMI}(\mathbf{M}) = \frac{p^{\kappa}(\mathbf{O}|\mathbf{H}_{ref},\mathbf{M})P(\mathbf{H}_{ref})}{\sum_H p^{\kappa}(\mathbf{O}|\mathbf{H},\mathbf{M})P(\mathbf{H})} \tag{2.21}$$

where κ is a scaling factor, $\mathbf{H}_{ref}$ denotes the reference transcription, and $\mathbf{H}$ represents all the possible hypotheses. The denominator hypotheses are normally stored in decoded speech lattices. MMI criterion is used to minimise the expected sentence error.

Minimum Phone Error (MPE)

Another type of criteria is designed to minimise the expected error corresponding to the distortions between the hypothesis and the reference. From the Bayesian perspective, this kind of criteria can be expresses as:

$$F_{MBR}(\mathbf{M}) = \sum_H P(\mathbf{H}|\mathbf{O},\mathbf{M}) loss(\mathbf{H},\mathbf{H}_{ref}) \tag{2.22}$$

where $\mathbf{M}$ is a set of model parameters, $loss(\cdot)$ is the loss function of the hypothesis given the reference (transcription). This criterion is known as minimum Bayesian risk (MBR) [11]. The model parameters can be estimated by

$$\hat{\mathbf{M}} = \arg\min F_{MBR}(\mathbf{M}) \tag{2.23}$$

Due to different loss functions, there a variety of discriminative training criteria. MPE criterion [27] is one of them, which adopts a loss function with respect to the word error rate (WER). Its loss function is:

$$loss(\mathbf{H},\mathbf{H}_{ref}) = \#(Corrections) - \#(Insertions) = A(\mathbf{H},\mathbf{H}_{ref}) \tag{2.24}$$

It measures the phone accuracy of the hypothesis given the correct transcription. The MPE can be expressed as:

$$F_{MMI}(\mathbf{M}) = \sum_H \frac{p^{\kappa}(\mathbf{O}|\mathbf{H},\mathbf{M})P(\mathbf{H}_{ref})}{\sum_{H\prime} p^{\kappa}(\mathbf{O}|\mathbf{H}\prime,\mathbf{M})P(\mathbf{H}\prime)} A(\mathbf{H},\mathbf{H}_{ref}) \tag{2.25}$$

It can be noted that the form of MPE is similar to MMI, while the $\mathbf{H}\prime$ and $\mathbf{H}$ here both denote the possible hypotheses.

Chapter 3

DNN-based Speaker Adaptation

Hybrid-SI DNN-HMM systems are built with the training methods mentioned in Section 2, assuming that the acoustic conditions are the same in training and test datasets. Obviously, it is not a suitable assumption in reality, as there will be mismatches occurring due to acoustic condition variabilities. As introduced in Section 1.2, one significant aspect of acoustic variabilities is the inter-speaker variability, such as different *first languages* (L1s) and English proficiencies. In order to solve the mismatch problem, speaker adaptation approaches are implemented. The basic concept of speaker adaptation is to minimise the mismatch between training and test data by normalising the features or tuning the model parameters according to a set of adaptation data. As described in Section 1.1, the framework of our current ASR is a joint decoding system consisting of a GMM-HMM tandem system and a DNN-HMM hybrid system. As mentioned in Section 1, a large number of adaptation techniques for the GMM-HMM tandem system has been investigated in previous work, but it is not the case to the DNN-HMM hybrid system. In this section, several state-of-art DNN-based speaker adaptation techniques will be discussed in this section.

3.1 Speaker Adaptive Training (SAT)

3.1.1 Constraint Maximum Likelihood Linear Regression (CMLLR)

Constraint Maximum Likelihood Linear Regression (CMLLR) [8] [5] now is widely used for improving the generalisation of DNNs, making acoustic models more independent from training speakers and perform better to unseen speakers in test data. The architecture of CMLLR is shown in Fig. 3.1.

The basic idea behind CMLLR is to apply a transform to the observations (the input features) before DNN training. The set of transforms derived from CMLLR can reduce the

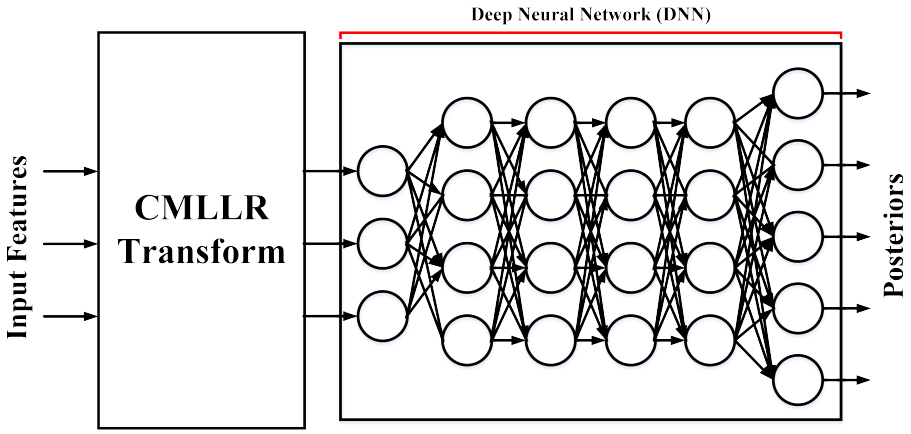


Fig. 3.1 Architecture of the SAT-DNN model using CMLLR.

mismatch between an initial model set and the adaptation data. Its general form is:

$$\hat{\mu} = \mathbf{A}'\mu - \mathbf{b}' \tag{3.1}$$

$$\hat{\Sigma} = \mathbf{A}'\Sigma\mathbf{A}'^{\top} \tag{3.2}$$

where μ and Σ denote the mean and variance of the observations respectively, $\mathbf{A}$ represents a weight matrix and $\mathbf{b}$ represents a bias vector.

The parameters of CMLLR can be estimated using *maximum likelihhod* (ML) criterion. The auxiliary function is:

$$Q(\mathbf{M},\hat{\mathbf{M}}) = c - \frac{1}{2}\sum_{n=1}^{N}\sum_{t=1}^{T}\gamma_n(t)\left[c_n + log(|\hat{\Sigma}_n|) + (\mathbf{o}(t)-\hat{\mu}_n)^{\top}\hat{\Sigma}_n^{-1}(\mathbf{o}(t)-\hat{\mu}_n)\right] \tag{3.3}$$

where c denotes a constant determined by the transition probabilities, c_n denotes the normalisation constant for component n, $\gamma_n(t) = p(x_n(t)|\mathbf{M},\mathbf{O})$ indicates the posterior of component n at time t, $\mathbf{O} = o(1),\cdots,o(T)$ represents the adaptation data. By substituting equation (3.1) (3.2) into equation (3.3), the auxiliary function for CMLLR can be re-written as:

$$Q(\mathbf{M},\hat{\mathbf{M}}) = c - \frac{1}{2}\sum_{n=1}^{N}\sum_{t=1}^{T}\gamma_n(t)\left[c_n + log(|\hat{\Sigma}_n|) - log(|\mathbf{A}|^2) + (\hat{\mathbf{o}}(t)-\hat{\mu}_n)^{\top}\hat{\Sigma}_n^{-1}(\hat{\mathbf{o}}(t)-\hat{\mu}_n)\right] \tag{3.4}$$

where

$$\hat{\mathbf{o}(t)} = (\mathbf{A}')^{-1}\mathbf{o}(t) + (\mathbf{A}')^{-1}\mathbf{b}' = \mathbf{A}\mathbf{o}(t) + \mathbf{b} = \mathbf{W}\mathbf{o}_{ext}(t) \tag{3.5}$$

where $\mathbf{W} = [\begin{smallmatrix}\mathbf{b}\\ \mathbf{A}\end{smallmatrix}]$ is the augmented transformation matrix, $\mathbf{o}_{ext}(t) = [\begin{smallmatrix}1\\ \mathbf{o}(t)\end{smallmatrix}]$ is the augmented observation.

The log-likelihood of CMLLR can be calculated by including Jacobian in equation (3.6)

$$\mathscr{L}(\mathbf{o}_{ext}|\mu,\Sigma,\mathbf{A},\mathbf{b}) = |\mathbf{A}|\mathscr{N}(\mathbf{A}\mathbf{o}ext + \mathbf{b};\mu,\Sigma) \tag{3.6}$$

3.2 Activation Function Adaptation

Instead of applying linear transformation to the input acoustic features using CMLLR, some other approaches perform the adaptation by tuning the parameters of the activation function in the DNN model. Two of this kind of adaptation methods, *Learning Hidden Unit Contributions* (LHUC) and *Parameterised Sigmoid Activation Functions* (p-Sigmoid),will be introduced in this section.

3.2.1 Learn Hidden Unit Contributions (LHUC)

LHUC [26] [24] is an effective model-based neural network speaker adaptation technique. The major concept of this method is to train a set of additional speaker-dependent(SD) amplitude parameters for each hidden unit, which are tied from each speaker.

For a SI-DNN model with parameter $\mathbf{M} = \{\mathbf{W}_1, \cdots, \mathbf{W}_L, \mathbf{W}_o\}$, adaptation data $\{\mathbf{x}_t^{(m)}, \mathbf{s}_t^{(m)}\}$ are used to train a set of SD parameters $\mathbf{M}^{(m)} = \{\mathbf{r}_1^{(m)}, \cdots, \mathbf{r}_L^{(m)}\}$ for speaker m, where $\mathbf{r}_l^{(m)} \in \mathbb{R}^{M_l}$ is the vector of SD parameters for the l-th hidden layer. The SD hidden layer output with LHUC can be written as:

$$\mathbf{h}_l^{(m)} = a(\mathbf{r}_l^{(m)}) \circ f_l(\mathbf{W}_l^\top \mathbf{h}_{l-1}) \tag{3.7}$$

where $\circ$ is an element-wise multiplication, $f^l(\cdot)$ is the activation function at the lth hidden layer, and $a(\cdot)$ is an element-wise function to constrain the amplitude of $\mathbf{r}_l^{(m)}$ to $[0,2]$.

$$a(c) = \frac{2}{1+exp(-c)} \tag{3.8}$$

In this case, the output of layer l is scaled by the SD term $a(\mathbf{r}_l^{(m)})$ for speaker m. $\mathbf{r}_l^{(m)}$ is always initialised to 0.0 to make the SD parameter $a(\mathbf{r}_l^{(m)}) = 1.0$, so that the initial SD model is equivalent to the SI model.

The SD parameters $\mathbf{M}^{(m)}$ can be generated by using CE criterion. Normally, frame-state alignments are provided as supervision, $\hat{\mathbf{y}}_k$ follows the Bernoulli distribution and equation (2.19) becomes the negative log posterior probability for speaker m over $K^{(m)}$ adaptation data points can be expressed as:

$$\mathscr{F}_{CE}(\mathbf{M}^{(m)}) = -\sum_{k}^{K^{(m)}} logP(\mathbf{s}_k|\mathbf{x}_k^{(m)};\mathbf{M}^{(m)}) \tag{3.9}$$

where the posterior $P(\mathbf{s}_k|\mathbf{x}_k^{(m)};\mathbf{M}^{(m)})$ is similar to equation (2.17)

$$P(s_k|\mathbf{x}_k^{(m)};\mathbf{M}^{(m)}) = \frac{exp(\mathbf{W}_{s_k}^{\top}a(\mathbf{r}_l^{(m)})\circ f_L)}{\sum_{s'} exp(\mathbf{W}_{s'}^{\top}a(\mathbf{r}_l^{(m)})\circ f_L)} \tag{3.10}$$

By applying the chain rule, the gradient of $\mathscr{F}_{CE}$ with respect to $\mathbf{r}_l^{(m)}$ in layer $l \in \{1,\cdots,L-1\}$ is:

$$\frac{\partial \mathscr{F}_{CE}}{\partial \mathbf{r}_l^{(m)}} = \frac{\partial \mathscr{F}_{CE}}{\partial f_L}\frac{\partial f_L}{\partial f_{L-1}}\cdots\frac{\partial f_{l+2}}{\partial f_l+1}\frac{\partial f_{l+1}}{\partial a(\mathbf{r}_l^{(m)})}\frac{\partial a(\mathbf{r}_l^{(m)})}{\partial \mathbf{r}_l^{(m)}} \tag{3.11}$$

where $\frac{\partial \mathscr{F}_{CE}}{\partial f_L}$ can be calculated by

$$\begin{aligned}\frac{\partial \mathscr{F}_{CE}}{\partial f_L} &= -\sum_k \frac{\partial}{\partial a(\mathbf{r}_L^{(m)})}\left(exp(\mathbf{W}_s^{\top}a(\mathbf{r}_l^{(m)})\circ f_L) - log\sum_{s'} exp(\mathbf{W}_{s'}^{\top}a(\mathbf{r}_l^{(m)})\circ f_L\right) \\ &= -\sum_k \left(\mathbf{W}_{s_k}^{\top} - \sum_{s'} P(s'|x_k^{(m)};\mathbf{M}^{(m)})\mathbf{W}_{s'}^{\top}\right) a(\mathbf{r}_L^{(m)})\end{aligned} \tag{3.12}$$

and the rest of equation (3.11) are obtained by

$$\frac{\partial f_{l+1}}{\partial f_l} = \frac{\partial f_{l+1}}{\partial \mathbf{q}_{l+1}}\frac{\partial \mathbf{q}_{l+1}}{\partial f_l} = \frac{\partial f_{l+1}}{\partial \mathbf{q}_{l+1}}\mathbf{W}_{l+1}^{\top}a(\mathbf{r}_l^{(m)}) \tag{3.13}$$

$$\frac{\partial f_{l+1}}{\partial a(\mathbf{r}_l^{(m)})} = \frac{\partial f_{l+1}}{\partial \mathbf{q}_{l+1}}\frac{\partial \mathbf{q}_{l+1}}{\partial a(\mathbf{r}_l^{(m)})} = \frac{\partial f_{l+1}}{\partial \mathbf{q}_{l+1}}\mathbf{W}_{l+1}^{\top}f_l \tag{3.14}$$

$$\frac{\partial f_{l+1}}{\mathbf{r}_l^{(m)}} = \frac{2exp(-\mathbf{r}_l^{(m)})}{(1+exp(-\mathbf{r}_l^{(m)}))^2} \tag{3.15}$$

where $\mathbf{q}_{l+1} = \mathbf{W}_{l+1}^{\top}a(\mathbf{r}_l^{(m)})\circ f_l$, and for sigmoid activation function, $\partial f_{l+1}/\partial \mathbf{q}_{l+1} = f_{l+1}(1-f_{l+1})$.

For the last hidden layer L, the gradient of $\mathscr{F}_{CE}$ with respect to $\mathbf{r}_l^{(m)}$ can be derived by

$$\frac{\partial \mathscr{F}_{CE}}{\partial \mathbf{r}_l^{(m)}} = \frac{\partial \mathscr{F}_{CE}}{a(\mathbf{r}_l^{(m)})} \frac{a(\mathbf{r}_l^{(m)})}{\partial \mathbf{r}_l^{(m)}} \tag{3.16}$$

where $\partial \mathscr{F}_{CE} / a(\mathbf{r}_l^{(m)})$ can be obtained from equation (3.7) (3.9) (3.10):

$$\frac{\partial \mathscr{F}_{CE}}{\partial a(\mathbf{r}_l^{(m)})} = -\sum_k \left(\mathbf{W}_{s_k}^\top - \sum_{s'} P(s'|x_k^{(m)}; \mathbf{M}^{(m)}) \mathbf{W}_{s'}^\top \right) f_L \tag{3.17}$$

Due to the nature of LHUC, it shows several advantages. First of all, it keeps the number of adaptation parameters no more than number of hidden units, which makes it computationally affordable. Besides, it can be applied to any feed-forward neural network as there is no assumption about the layer structure. Moreover, it does not depend on SAT, so the training and adaptation processes becomes simple and Hybrid-SI model can be directly used.

3.2.2 Parameterised Sigmoid Activation Functions (*p*-Sigmoid)

Speaker adaptation using p-Sigmoid activation function [31] is a similar approach to LHUC. Its basic idea is also introducing a set of additional SD parameters $\mathbf{M}^{(m)} = \{\alpha_j^{(m)}\}$ for speaker m at target SI hidden unit j to the standard sigmoid function. The hidden layer output with p-Sigmoid activation function can be written as:

$$\begin{aligned} \mathbf{h}_l^{(m)} &= \alpha_j^{(m)} \cdot \frac{1}{1 + exp(\mathbf{W}_{l-1}^\top \mathbf{h}_{l-1})} \\ &= \alpha_j^{(m)} \cdot f(\mathbf{W}_{l-1}^\top \mathbf{h}_{l-1}) \end{aligned} \tag{3.18}$$

where f denotes the sigmoid activation function, and the SD parameter $\alpha_j^{(m)}$ is normally set to 1.0 to make the initial SD model equivalent to a SI model. The SD parameter can be trained using CE criterion similar to LHUC.

The major difference between p-sigmoid and LHUC method is that a sigmoid function constraint scaling factor in LHUC is replaced by a linear scaling factor $\alpha_j^{(m)}$. This difference means that the SD parameters of p-Sigmoid are able to be jointly learned with other DNN model parameters more easily, making p-Sigmoid method more flexible than LHUC.

3.3 I-vector based Speaker-aware Training (SaT)

Speaker adaptation can also be performed by providing the DNN model with extra information about speakers before training for speaker normalisation. It takes advantage of the ability of a DNN to exploit supplementary information to modify its parameters. This kind of adaptation is known as *speaker-aware training* (SaT) [21]. In SaT, i-vector technique is widely used to find a speaker subspace and extract the speaker-specific information. In this section, the extraction process of i-vectors as well as two i-vector based adaptation approaches will be introduced.

3.3.1 I-vector Extraction and Training

I-vector is considered as a compact low-dimensional representation of acoustic characteristics from individual speakers. The extraction of i-vector is achieved in an unsupervised fashion, which means it requires no transcriptions or hypothesis [28]. Consequently, the SAT-DNN using i-vector technique is robust to errors in hypothesis. The extraction process of i-vector is describes as below [17]:

1. **Universal Background Model (UBM) generation**
   Universal background model (UBM) [20] is a speaker independent GMM model consisting of N Gaussian components, which is defined by a mean supervector $\mu_0^{(n)}$, diagnoal component covariance matrices $\Sigma^{(n)}$ and mixture coefficients $\omega^{(m)}$.

2. **I-vector Estimation**
   On top of the UBM obtained in the first step, a SD model $\mathscr{M}$ is generated, which is trained at speaker-level using all the data belong to speaker m. I-vectors are then initialised at speaker level from the SD model $\mathscr{M}$, and spanning a so-called *speaker eigenspace*. Each point in this speaker eigenspace indicates one speaker. The speaker-adapted mean supervector $\mu_m^{(n)}$ of component $n \in N$ for speaker m is then given by:

   $$\mu_m^{(n)} = \mu_0^{(n)} + \mathbf{W}^{(n)}\mathbf{v}^{(m)} \tag{3.19}$$

   where $\mathbf{v}^{(m)}$ is the i-vector for speaker m with D_{iv} dimensions that indicates the eigenspace is spanned by D_{iv} bases with the highest variability in the mean supervector. $\mathbf{W}^{(n)}$ is a $D \times D_{iv}$ weight matrix for component n. D is the size of the mean supervetor.

   In practical, after the speaker-adapted mean supervector $\mu_m^{(n)}$ is extracted from a SD model $\mathscr{M}$, dimension reduction technique like *principal component analysis* (PCA) is applied to obtain the initial D_{iv}-dimension i-vectors. I-vectors and parameters of

the SD model $\mathscr{M}$ are estimated using *maximum likelihood* (ML) criterion. Similar to equation (3.3), an auxiliary function can be built:

$$Q(\mathscr{M}, \mathbf{v}^{(m)}; \hat{\mathscr{M}}, \hat{\mathbf{v}}^{(m)}) = -\frac{1}{2} \sum_{m,t,n} \gamma_t^{(n)}(m)(\mathbf{x}_t - \mu_m^{(n)})^\top \Sigma^{(n)-1}(\mathbf{x}_t - \mu_m^{(n)}) \tag{3.20}$$

where $\gamma_t^{(n)}(m)$ is the posterior probability of component n for speaker m at time t, and it depends on the model parameters $\hat{\mathscr{M}}$ and the i-vector $\hat{\mathbf{v}}^{(m)}$. Details of ML training has be introduced in Section 2.2.2.

3.3.2 Feature Augmentation

SaT can be simply achieved by training DNN models using acoustic features concatenated with i-vectors [22] [17] [15]. The framework of this method is shown in Fig 3.2, and it can be expressed by:

$$\mathbf{a} = \sum \mathbf{W}_{mix} \mathbf{x}_{mix}^{(m)} + \mathbf{b} \tag{3.21}$$

$$= \sum \begin{bmatrix} \mathbf{W}_o \\ \mathbf{U}^{(m)} \end{bmatrix} \begin{bmatrix} \mathbf{x}_o & \mathbf{v}^{(m)} \end{bmatrix} + \mathbf{b} \tag{3.22}$$

where $\mathbf{W}_o$ is the weight matrix for the original acoustic feature vector $\mathbf{x}_o$, $\mathbf{U}^{(m)}$ is the weight matrix for the i-vector $\mathbf{v}^{(m)}$ of speaker m, and $\mathbf{b}$ is the bias. Accordingly, the input of each hidden layer can be regarded as being added a SD bias, so the representation of layer l can be written as:

$$\mathbf{h}_l^{(m)} = f(\mathbf{W}_l \mathbf{h}_{l-1}^{(m)} + \mathbf{b}_l^{(m)}) \tag{3.23}$$

$$= f(\mathbf{W}_l \mathbf{h}_{l-1}^{(m)} + \mathbf{U}^{(m)} \mathbf{v}^{(m)} + \mathbf{b}_l)$$

where $\mathbf{W}_l$ is the weight matrix for the input at layer l, $\mathbf{b}_l^{(m)}$ denotes the SD bias and f is the activation function. The output of node j in each layer is:

$$\mathbf{y}_j = f_j(\mathbf{a}_{lj}) \tag{3.24}$$

where f_j is the activation function, $\mathbf{y}_j$ denotes the output at node j, and $\mathbf{a}_{lj}$ is the input of layer l at node j. CE criterion can be used to estimate the SD matrix $\mathbf{U}^{(m)}$:

$$F_{CE} = -\sum_k \hat{\mathbf{y}}_k log(\mathbf{y}_k) \tag{3.25}$$

where k denotes the node k in each layer. SGD optimisation will be applied to minimize the CE and find the estimation of the matrix $\mathbf{U}^{(m)}$:

$$\begin{aligned}\frac{\partial F_{CE}}{\partial \mathbf{U}^{(m)}} &= -\sum_k \frac{\hat{\mathbf{y}}_k}{\mathbf{y}_k}\frac{\partial \mathbf{y}_k}{\partial \mathbf{U}^{(m)}} \\ \frac{\partial \mathbf{y}_k}{\partial \mathbf{U}^{(m)}} &= \frac{\mathbf{v}^{(m)} \cdot exp(-\mathbf{a}_{lk})}{(1+exp(-\mathbf{a}_{lk}))^2}\end{aligned} \tag{3.26}$$

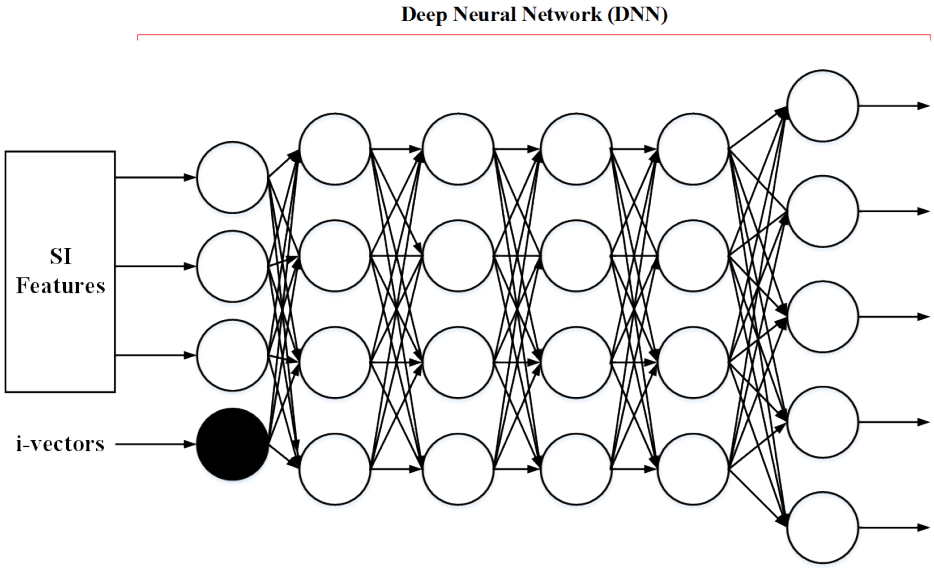


Fig. 3.2 Topology of i-vector based feature augmentation.

3.3.3 Factorised Feature Transforms

I-vectors can also be implemented in factorised feature transforms [21] [6], where a SD linear transformation is applied based on i-vectors in addition to the SD bias. Its topology is shown in Figure 3.3 and the general form 3.21 can be re-written as:

$$\begin{aligned}\mathbf{a} &= \sum \mathbf{W}_{mix}\mathbf{x}_{mix}^{(m)} + \mathbf{b} \\ &= \sum \begin{bmatrix} \mathbf{W}_o + \mathbf{W}^{(m)}D^{(m)} \\ \mathbf{U}^{(m)} \end{bmatrix} \begin{bmatrix} \mathbf{x}_o & \mathbf{v}^{(m)} \end{bmatrix} + \mathbf{b}\end{aligned} \tag{3.27}$$

where $\mathbf{W}^{(m)}$ is the factor matrix, and $D^{(m)} = diag(\mathbf{v}^{(m)})$. Consequently, the representation of layer l is:

$$\mathbf{h}_l = f(\mathbf{W}_l\mathbf{h}_{l-1} + \mathbf{W}^{(m)}D^{(m)}\mathbf{h}_{l-1} + \mathbf{U}^{(m)}\mathbf{v}^{(m)} + \mathbf{b}_l) \tag{3.28}$$

In this case, the SD parameters $\mathbf{M} = \{\mathbf{W}^{(m)}, \mathbf{U}^{(m)}\}$, where $\mathbf{W}^{(m)}$ and $\mathbf{U}^{(m)}$ can be estimated iteratively using CE criterion.

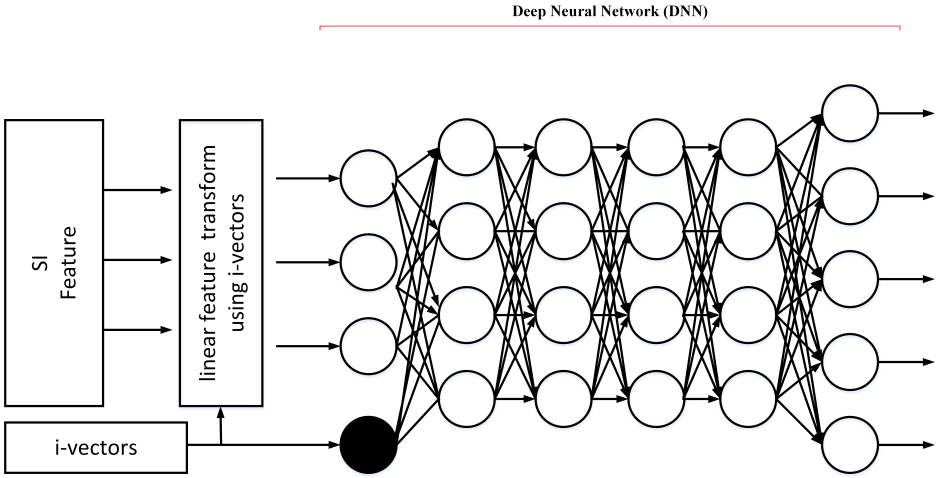


Fig. 3.3 Topology of i-vector based factorised feature transform.

3.4 Summary

Speaker adaptive training (SAT) using CMLLR has been proved to show significant improvement to the DNN-HMM based ASR systems. However, CMLLR is applied in a supervised form, making it extremely expensive. In contrast, activation function adaptation approaches are in unsupervised fashion using decoding hypotheses, which actually scales the output of each layer by a set of SD parameters $\mathbf{M}$. Consequently, such adaptation only refines the SD parameters and does not change the structure and parameters of the neural network, which strengthens its robustness against over-fitting. I-vector based SaT approaches are also unsupervised according to the nature of i-vectors. The feature augmentation way is intuitive and direct, which concatenates the acoustic feature with the i-vector to build a speaker-adapted feature and feeds it into the DNN training. As for factorised feature transform, it can be regarded as a further transform on top of the feature augmentation by adding a SD transform to the input using i-vectors. The basic concept behind factorised feature transform is similar to activation function adaptation, where factorised feature transform applies a SD linear transformation to the input of each layer while activation function adaptation introduces a SD scaling factor to the output of the each layer. The difference is that the factorised feature transform needs to rebuild a SD DNN model by adding speaker-specific information during training. This SD DNN model can then used to decode any test datasets directly. However,

this adaptation approach highly depends on the i-vector extracting model, the generality of which directly influences the performance of the SD DNN model.

Chapter 4

Experimental Setup

In this project, several DNN-based unsupervised speaker adaptation approaches will be implemented to a Hybrid-SI model and their performance is compared with Hybrid-SAT using CMLLR. The following sections was divided into three parts. In Section 4.1, the characteristics of the data and features, and details about the Hybrid-SI model used in this project are introduced. In Section 4.2 and Section 4.3, *learning rates*(LRs) choosing procedure and their corresponding learning curve for activation function adaptation at speaker-level and language level, and i-vector based feature augmentation are demonstrated to prove the correctness of their implementation.

4.1 Infrastructure

In this section, information about datasets, acoustic features and Hybrid-SI model is introduced.

4.1.1 Datasets

The data used in this project comes from *Business Language Testing Service*(BULATS) with non-native speakers with a number of *first languages* (L1s) from different countries. BULATS consists of A-E 5 sections, where the content of each section is shown in Table 4.1. Among these sections, what we are interested in are section C-E, as speakers talk in a free style in these sections, which can indicate their English skill levels more obviously. The proficiency of English is divided into 6 levels by CEFR shown in Table 4.2.

In the following experiments, the training dataset **BLXXXtrain** that contains 108.23 hours speech recordings of 1075 Indian Gujarati speakers from BULATS, and four adaptation datasets **BLXXXtrn04**, **BLXXXeval1**, **BLXXXeval2**, and **BLXXXeval3** consisting of

Section	Content	Example
A	Introducotry Questions	what's your name
B	Read Aloud	read specific sentences
C	Topic Discussion	discuss a successful business case
D	Interpret and Discuss Chart/Slide	talk about the information in a given chart/slide
E	Answer Topic Questions	5 questions about operating a small company

Table 4.1 Five Sections of BULATS

Level	Discription	Group Level	Group Name
A1	Beginner	A	Basic User
A2	Elementary		
B1	Intermediate	B	Independent User
B2	Upper Intermiate		
C1	Advanced	C	Proficient User
C2	Mastery		

Table 4.2 Foreign language proficiency levels in CEFR framework

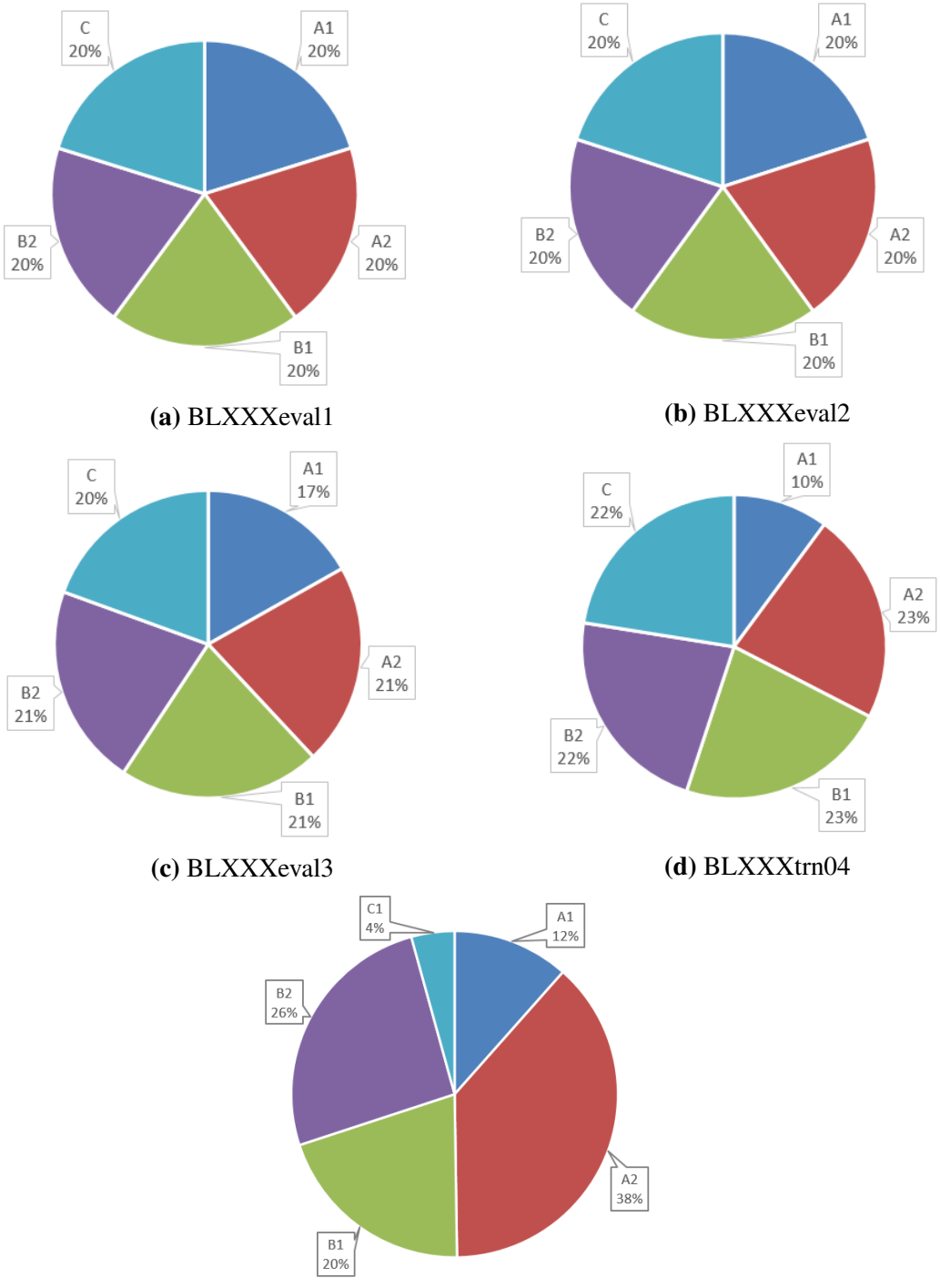
different L1s are adopted. The statistics of these datasets are shown in Table 4.3, and the distribution of the speakers' English level is illustrated in Figure 4.1.

4.1.2 Acoustic Features

The configuration of input features are 39-dimensional PLP features after *Heteroscedastic Linear Discriminant analysis* (HLDA) combined with 39-dimensional *bottleneck* (BN) features generated from AMI dataset, where AMI dataset is a large corpus consisting of high-quality recordings from native and proficient non-native English speakers. The generation process of input features is indicated in Figure 4.2.

Dataset	Speakers	L1s	Duration(hrs)
BLXXXtrn00+01+02+03	1075	Gujarati (100%)	108.23
BLXXXtrn04	442	Spanish(100%)	26.53
BLXXXeval1	223	Gujarati(100%)	24.11
BLXXXeval2	220	Spanish(100%)	24.68
BLXXXeval3	226	Dutch(14.6%), Polish(17.7%), French(16.4%) Thai(16.4%), Arabic(17.7%), Vietnamese(17.3%)	25.44

Table 4.3 Statistics of test datasets:BLXXXtrn04, BLXXXeval1, BLXXXeval2, and BLXXXeval3.



(a) BLXXXeval1

(b) BLXXXeval2

(c) BLXXXeval3

(d) BLXXXtrn04

(e) BLXXXtrn00+01+02+03

Fig. 4.1 Distribution of speakers' English levels in five different datasets.

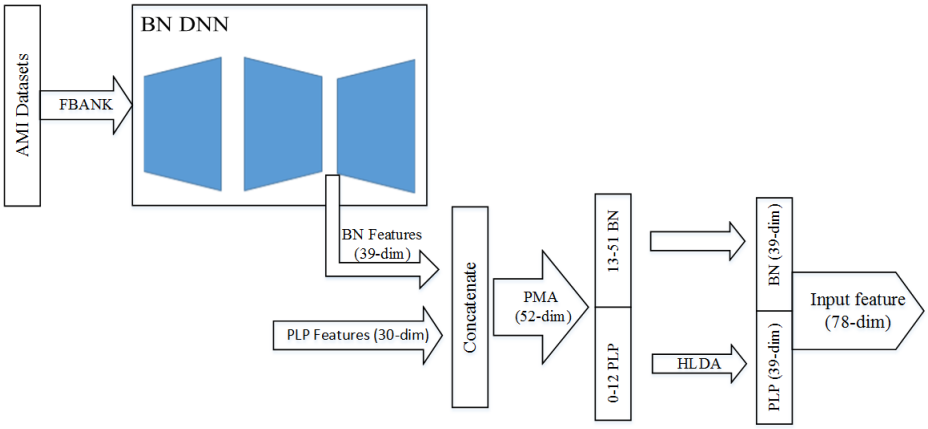


Fig. 4.2 Framework of input feature generation.

4.1.3 Hybrid-SI Model

The Hybrid-SI model used in this project is a triphone state DNN-HMMs, which consists of 5 hidden layers with sigmoid activation function and softmax output function. Each hidden layer contains 1000 nodes, and the output layer contains 6019 nodes. At the input layer, as mentioned in the previous section, the input acoustic feature is a 78-dimensional PLP+BN feature vector in one frame. Each frame is concatenated with 4 frames on its left and 4 frames on its right, so there are $78 \times 9 = 702$ nodes at the input layer in total. This Hybrid-SI model is trained on BLXXXtrain dataset using CE criterion training first. Then sequence level training using MPE criterion is applied on top of the DNN model from CE training, in order to obtain better performance.

In our project, the CE training of DNNs consists of two steps, discriminative pre-training and fine-tuning, and *stochastic gradient descent* (SGD) optimisation is applied. Specifically, the pre-training is a layer-by-layer procedure with *LIST* [29] updating schedule, where the initial DNN model starts with one hidden layer. After finishing the current training, a new hidden layer with random initialisation is added. This process executes iteratively until obtaining the target number of layers. Particularly, in the last pre-training epoch, the last inserted hidden layer is only initialised randomly and its parameters will be updated in the first epoch of fine-tuning step. The fine-tuning step consists of 16 - 20 epochs with *NEWBOB*[29] updating schedule, and manages to refine the DNN parameters to achieve better performance. As for sequence training using MPE criterion, phone-marked numerator and denominator lattices need to be generated first. Then *LIST* schedule is applied in MPE training to execute multiple iterations.

4.2 Speaker-level Adaptation

Activation function adaptation like LHUC and *p*-Sigmoid, and SaT using i-vector augmentation were implemented at speaker-level. The following sections introduces more details about the adaptation process.

4.2.1 Activation Function Adaptation

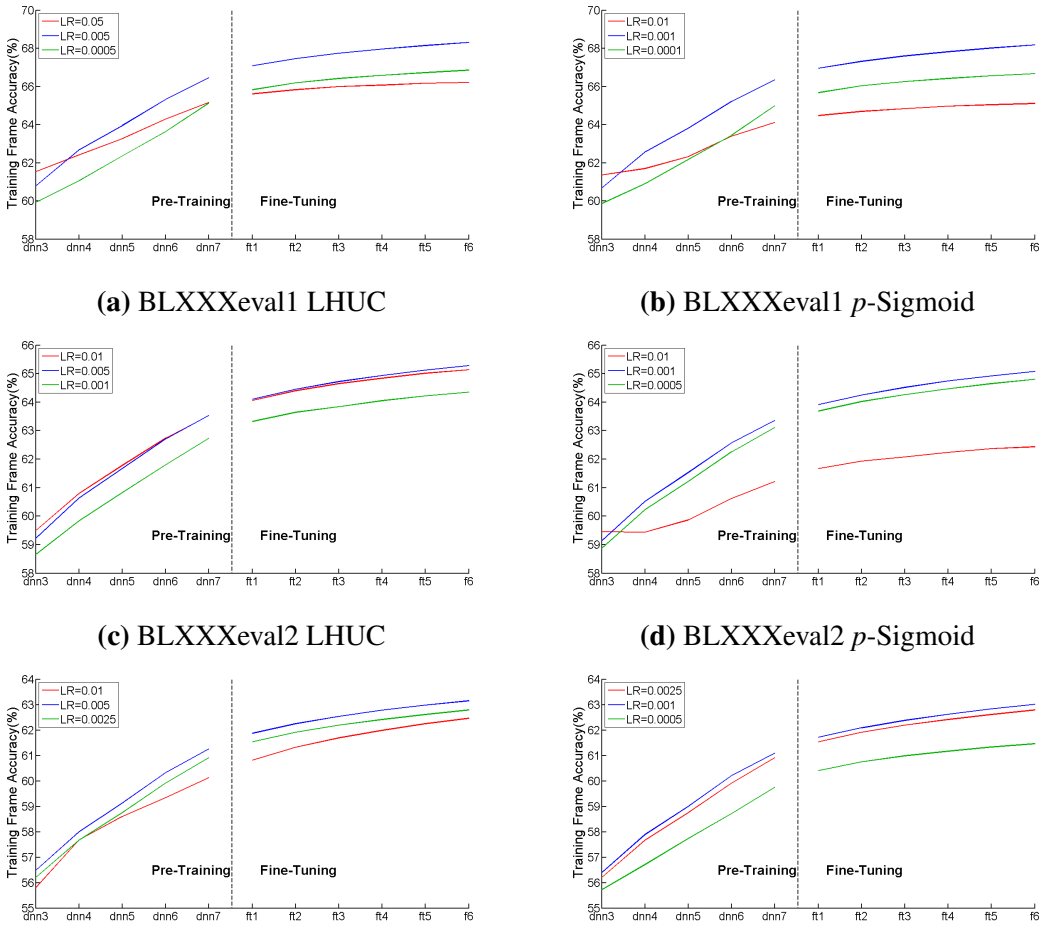
LHUC and *p*-Sigmoid adaptation was applied to the Hybrid-SI model using BLXXXeval1, BLXXXeval2 and BLXXXeval3 datasets. This kind of adaptation is achieved by two steps (similar to the DNN training process in Section 4.1.3): pre-training and fine-tuning. In pre-training step, the SD parameters $\mathscr{M}$ for the 5 hidden layers are trained layer by layer (marked as dnn3 - dnn7). In fine-tuning step, the SD parameters of the pre-trained adapted model in dnn7 will be further refined by 6 epochs. *Cross entropy* (CE) criterion is used in these two steps as explained in Section 3.2.1. In our experiments, the training frame accuracy is recorded as an indicator to measure the generality of classification. It represents the proportion of frames in which the maximum posterior agrees with the correct class. Different learning rates were tried to find the best one to get a best frame accuracy curve. The corresponding training frame accuracy is plotted in Figure 4.3.

The best value of the LR should make the learning curve converge and also obtain a relatively high frame accuracy at the end of fine-tuning process. Based on this standard, the best LRs for each situation can be chosen from Figure 4.3 that for all the three datasets, the relatively better $\mathrm{LR}^{\mathrm{spk}}_{\mathrm{LHUC}} = 0.005$ and $\mathrm{LR}^{\mathrm{spk}}_{p\mathrm{Sigmoid}} = 0.001$.

4.2.2 Feature Augmentation using i-vectors

Another adaptation approach is to augment the input acoustic features with extra SD informations before training, which is known as *speaker-aware training* (SaT). One of the most popular and simple way is to add i-vectors directly as in Section 3.3.2. The DNN model takes i-vectors as speaker-specific information and normalise original input features.

In this experiment, a SD model $\mathscr{M}$ was first trained on top of a 128-component UBM using data from Indian Gujarati speakers on BLXXXtrain. Then, i-vectors with different dimensions were extracted from $\mathscr{M}$ and concatenated with the original acoustic features to train the DNN model. As explained in Section 4.1.3, the original Hybrid-SI model is trained using CE and MPE criterion. Considering that the time consumption of the lattice generation in MPE training is quite expensive, only CE training is adopted in this part of experiment. The *cross-validation*(CV) frame accuracy curve during CE training is plotted in



(a) BLXXXeval1 LHUC

(b) BLXXXeval1 *p*-Sigmoid

(c) BLXXXeval2 LHUC

(d) BLXXXeval2 *p*-Sigmoid

(e) BLXXXeval3 LHUC

(f) BLXXXeval3 *p*-Sigmoid

Fig. 4.3 Training frame accuracy(%) with different LRs on BLXXXeval1, BLXXXeval2 and BLXXXeval3 using LHUC and *p*-Sigmoid speaker-level adaptation

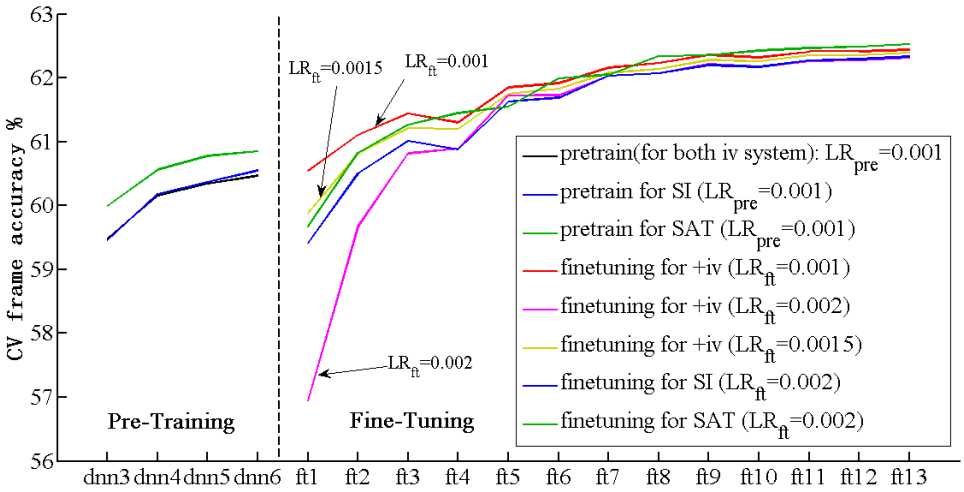


Fig. 4.4 Cross-validation frame accuracy(%) during CE training.

Figure 4.4, where the i-vector used is 10 dimension. It can be found that for Hybrid-SI and Hybrid-SAT models, the first epoch (*epoch1*) of fine-tuning with *fine-tuning LR* $\mathrm{LR_{ft}} = 0.002$ only decreases the CV frame accuracy in the last epoch (*dnn6*) of pre-training by no more than 1.0%. However, Feature augmentation using 10-dimensional i-vectors sees a sharp decline in epoch1 of fine-tuning if $\mathrm{LR_{ft}} = 0.002$. Intuitively, as mentioned in Section 4.1.3 epoch1 of fine-tuning actually tunes a DNN model with the last hidden layer randomly initialised, which can also be regarded as the pre-training for the last hidden layer. Due to that and considering Hybrid-SI and Hybrid-SAT situations, even the $\mathrm{LR_{tf}}$ (0.002) is different from the $\mathrm{LR_{pre}}$ (0.001), the CV frame accuracy will not have such a dramatic decrease. In this case, different $\mathrm{LR_{ft}}$ was tried to the same pre-trained model to examine the change of CV frame accuracy. From the red, yellow and pink lines in Figure 4.4 as can be seen that the CV frame accuracy of epcoh1 in fine-tuning gradually get closer to that of dnn6 in pre-training. The reason should be related to the newly added i-vectors. As i-vector is a compact low-dimensional presentation of the features in a subspace, when it concatenates with the original acoustic features, it is possible that the new feature space makes the DNN model become more sensitive to the LR. As a consequence, a sudden increase of LR from the end of pre-training to the start of fine-tuning may lead to overshooting problems in the SGD optimisation process, resulting in such a large drop of the CV frame accuracy. After that, due to the nature of *NEWBOB* updating schedule[29], the final fine-tuned models with various $\mathrm{LR_{ft}}$ show similar frame accuracies, but the model with initial $\mathrm{LR_{ft}} = 0.001$ still obtains slight higher frame accuracy than the other two. In that case, following experiments will be carried out using $\mathrm{LR_{ft}} = 0.001$.

4.3 Language-level adaptation

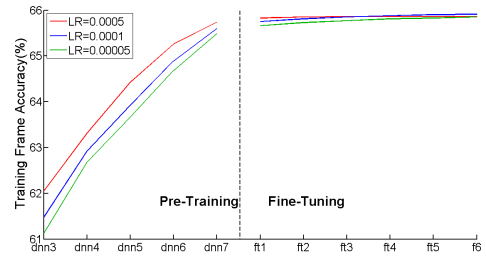
In this section, activation function adaptation was implemented to the Hybrid-SI at language-level. There are two parts of experiments using four datasets (BLXXXeval1, BLXXXeval2, BLXXXeval3, and BLXXXtrn04):

1. Investigate the influence of different L1s and the number of L1s in adaptation datasets on the language-level adaptation performance. BLXXXeval1, BLXXXeval2 and BLXXXeval3 are compared, as BLXXXeval1 has the same L1 - Gujarati as the training dataset, BLXXXeval2 has the different L1 - Spanish, and BLXXXeval3 has 6 different L1s consisting of European and Asian languages.

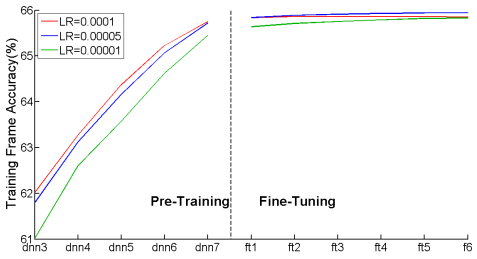
2. Investigate the influence of the quality of supervisions on the language-level adaptation performance. BLXXXeval2 and BLXXXtrn04 are chosen, as both of them contain the same L1 - Spanish but have different tri-gram decoding WERs from Hybrid-SI. Moreover, the crowd-sourced transcripts of BLXXXtrn04 are also used as the supervision to compare with the Hybrid-SI supervsion.

The training frame accuracies with various LRs are plotted in Figure 4.5. Compared with frame accuracy curves at speaker-level adaptation in Figure 4.3, the LRs used in language-level are around 10 to 50 times smaller. It is because at language-level, the SD parameters $\mathbf{M}$ are trained for each L1 rather than each speaker, it learns from a much larger number of data and becomes more susceptible to the LR. For instance, according to Section 4.1.1, except for BLXXXtrn04, the other 3 adaptation datasets contain approximately 200 speakers but only 1-6 L1s, which means the data used for training $\mathbf{M}$ in language-level are 30-200 times larger than in speaker-level. Consequently, the LRs at language-level should be reduced to avoid overshooting. For example, in Figure 4.5c the learning curve witnesses sudden drop after dnn6 in pre-training when the LR=0.005. Another trend that can be observed is that the frame accuracies almost stay at the same level on the fine-tuning step. The reason is that such a large number of language-level data and a relatively small LR make the adapted model become stable after pre-training and hard to be refined.

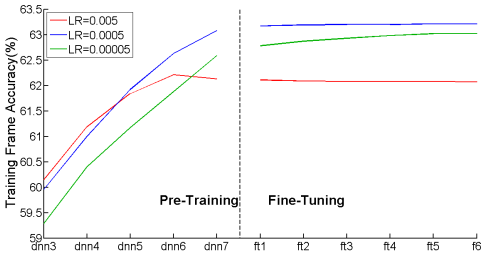
Based on the standard of in Section mentioned in Section 4.2.1, the best LRs for each adaptation and dataset are summarised in Table 4.4.



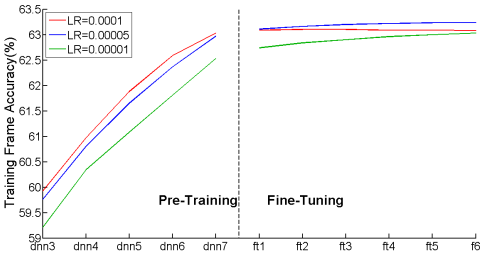
(a) BLXXXeval1 LHUC



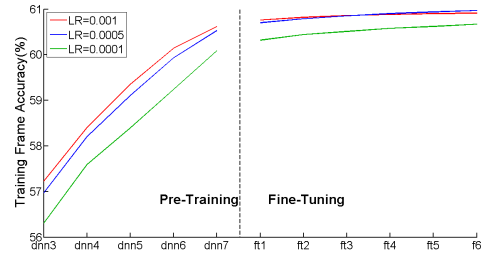
(b) BLXXXeval1 *p*-Sigmoid



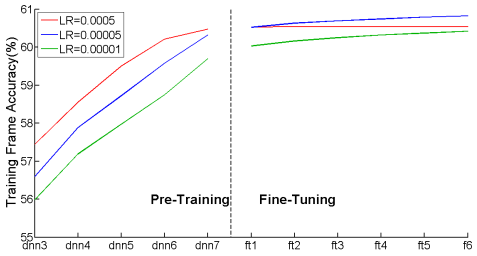
(c) BLXXXeval2 LHUC



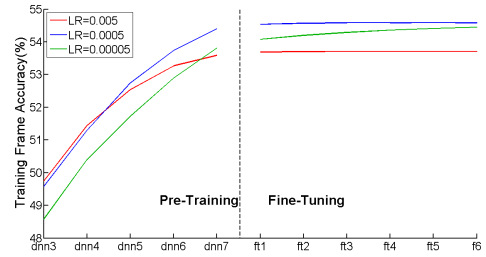
(d) BLXXXeval2 *p*-Sigmoid



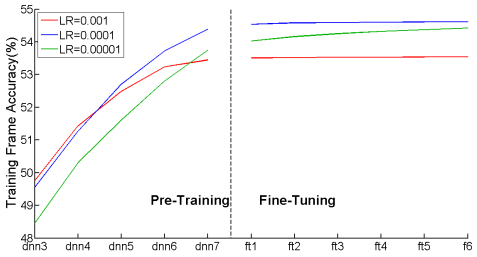
(e) BLXXXeval3 LHUC



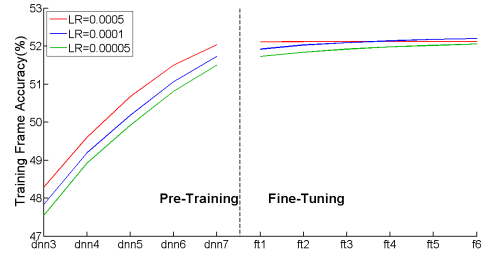
(f) BLXXXeval3 *p*-Sigmoid



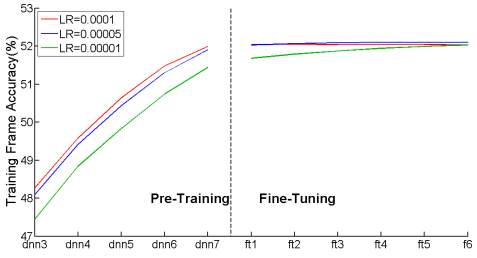
(g) BLXXXtrn04(Hybrid-SI) LHUC



(h) BLXXXtrn04(Hybrid-SI) *p*-Sigmoid



(i) BLXXXtrn04(crowd-sourced) LHUC



(j) BLXXXtrn04(crowd-sourced) *p*-Sigmoid

Fig. 4.5 Training frame accuracy(%) with different learning rate (LR) on BLXXXeval1, BLXXXeval2, BLXXXeval3 and BLXXXtrn04 using LHUC and *p*-Sigmoid language-level adaptation

Datasets (Supervision)	**Learning Rates**	
	LHUC	p-Sigmoid
BLXXXeval1(Hybrid-SI)	0.0001	0.00005
BLXXXeval2(Hybrid-SI)	0.0005	0.00005
BLXXXeval3(Hybrid-SI)	0.0005	0.00005
BLXXXtrn04(Hybrid-SI)	0.0005	0.00001
BLXXXtrn04(crowd-sourced)	0.0001	0.00005

Table 4.4 Learning rates chosen for LHUC and *p*-Sigmoid with different datasets

Chapter 5

Results and Analysis

This section discusses the results and findings of the experiments, which consists of two main parts. The first part, at speaker-level, investigates the effect of activation function adaptation on different L1s, and the results of feature augmentation using i-vectors with different dimensions. The second part analyses the language-level activation function adaptation results with respect to different L1s and different supervisions. The influence of different English-level speakers on the adaptation performance is also included.

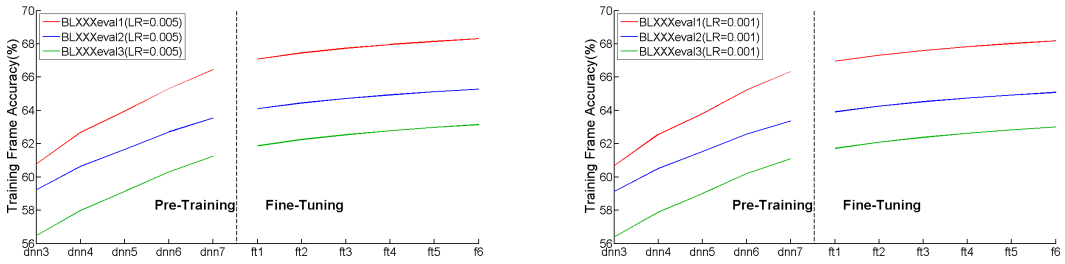
5.1 Speaker-Level Adaptation

Activation function adaptation like LHUC and p-Sigmoid, and feature augmentation using i-vectors are implemented in this section at speaker level. The speaker-level SD parameters $\mathbf{M}^{(m)}$ are trained using all the data belonging to speaker m, which makes the adaptation highly depends on the data of each speaker. Consequently, DNNs can learn the difference between individual speakers with different L1s, and reduce the mismatch caused by inter-speaker variabilities.

5.1.1 Activation Function Adaptation

Adaptation to Hybrid-SI Model

As explained in Section 4.2.1, activation function adaptation was applied to the Hybrid-SI model using BLXXXeval1, BLXXXeval2 and BLXXXeval3. The frame accuracy curves of each dataset with the best LRs are shown in Figure 5.1, and the scoring results of adapted models using LHUC and p-Sigmoid are shown in Table 5.1.



(a) LHUC **(b)** *p*-Simgoid

Fig. 5.1 Training frame accuracy(%) of LHUC and *p*-Sigmoid adaptation using BLXXXeval1, BLXXXeval2 and BLXXXeval3 with the best LRs at speaker level.

Datasets	L1s	SI	SAT	SI+LHUC	SI+pSigmoid	Speakers
BLXXXeval1	Gujarati	37.46	35.20	38.65	38.94	223
BLXXXeval2	Spanish	55.12	50.89	53.29	53.35	220
BLXXXeval3	Multi-L1s[1]	52.59	48.02	49.83	49.95	226

Table 5.1 Trigram WER(%) on BLXXXeval1,BLXXXeval2, and BLXXXeval3 using different adaptation methods at speaker level.

The first thing that can be noticed from Table 5.1 is that the both the supervision (Hybrid-SI) and the baseline (Hybrid-SAT) WERs of BLXXXeval1 are obviously lower than that of BLXXXeval2 and BLXXXeval3. It confirms that the mismatch between training and test dataset caused by speakers' L1s will degrade the ASR performance, as our training data are all Indian Gujarati speakers, which match with BLXXXeval1 but are different from BLXXXeval2 and BLXXXeval3.

In terms of the WERs of adapted models, the scoring results of BLXXXeval1 are unusual, where its WER_{LHUC} and $\text{WER}_{p\text{Sigmoid}}$ are even worse than the WER_{SI}. It also seems not to match Figure 5.1, where the frame accuracy of BLXXXeval1 witnesses an obvious increase after fine-tuning and pre-training. From the frame perspective, as Shire, M.L.(2001) mentioned in [23], a good frame accuracy is important for speech recognition but it does not have a strong and direct relationship with WERs. As frames are not equally important in the ASR system, a high frame accuracy may also lead to disappointing WERs. Especially for our datasets containing speakers with various English levels, it is likely that some frames outweigh some others. Another possible reason is also related to the nature of our datasets. As discussed in section 4.1.1, nearly 40% speakers are at A1 and A2 levels, whose speech recordings are incoherent and contain lost of grammar mistakes, repetition and hesitation. This part of data will significantly degrade our ASR system. However, these data also contains features of particular L1 speakers. In this case, due to the limited number of data,

Supervision	SI	SAT	LHUC	Speakers	Duration(hrs)
BLXXXeval1	29.39	28.18	29.74	223	24.11
BLXXXeval1.C			29.48	45	5.06

Table 5.2 Trigram WER(%) on C-level speakers from LHUC adapted models using BLXXXeval1 and BLXXXeval1.C respectively.

adding this kind of low quality data to the adaptation datasets also makes contributions to reducing the mismatch between different L1s. In that case, there is a trade-off between quantity and quality of the adaptation datasets. Only choosing high-level speakers can eliminate the negative effect on the ASR performance but causes the insufficient data issue, while using all-level speakers provides enough adaptation data to narrow the mismatch of L1s but may worse the recognition performance to some degree. As BLXXXeval1 has the same L1 speakers as the training data BLXXXtrain and its WER_{SI} is already low enough, the negative effect from low-level speakers predominates. In order to prove it, a contrast experiment was carried out on BLXXXeval1: a) The current decoding results were scored on C-level speakers only. b) A subset of BLXXXeval1 was built by selecting all the C-level speakers (marked as BLXXXeval1.C), and then adapted to the Hybrid-SI model using LHUC. c) This adapted model was decoded on C-level speakers and compared with the scoring results in a). The results are shown in Table 5.2. As can be seen that the adapted model using BLXXXeval1.C shows 0.3% lower WER than using the whole BLXXXeval1, and meanwhile it uses 5 times fewer data, it confirms the negative effect given by low-level speakers. Although its WER is still higher than Hybrid-SI, it is likely resulted from the insufficient adaptation data.

For BLXXXeval2 and BLXXXeval3, their L1s are apparently different from the training dataset, so the low-level speakers' positive effect of reducing mismatches surpass its negative effect to the WER. On both datasets, LHUC and p-Sigmoid adaptation show similar performance. It matches with Figure 5.1 where the LHUC and p-Sigmoid have almost the same frame accuracy curves. To be specific, the WER_{LHUC} are 2.40% and 1.81% higher than that of Hybrid-SAT for BLXXXeval2 and BLXXXeval3 respectively, and the corresponding $\text{WER}_{p\text{Sigmoid}}$ are 2.46% and 1.93% higher than that of Hybrid-SAT. Both of them are not so good as Hybrid-SAT but show considerable WER reduction. In addition, the scoring results also indicate the activation function adaptation is sensitive to the WER of the supervision, as both LHUC and p-Sigmoid show closer performance to Hybrid-SAT on BLXXXeval3 ($\text{WER}_{\text{SI}} = 52.59\%$) than BLXXeval2($\text{WER}_{\text{SI}} = 55.12\%$).

L1s	SI	SAT	SI+LHUC	SI+pSigmoid	Speakers
Dutch	45.17	42.25	42.33	42.82	33
Polish	46.85	43.78	44.03	44.29	40
French	48.92	44.78	45.53	45.86	37
Thai	54.56	50.36	51.95	52.16	37
Arabic	59.13	53.84	56.11	55.95	40
Vietnamese	61.14	53.19	58.28	57.80	39
Overall	52.59	48.02	49.83	49.95	226

Table 5.3 Trigram WER(%) of BLXXXeval3 divided by languages.

Especially, the speakers in BLXXXeval3 have 6 different L1s, so it would be interesting to investigate the adaptation performance of each L1. The scoring results of BLXXXeval3 divided by L1s are shown in Table 5.3.

A clear boundary can be firstly drawn from Table 5.3 that the WERs are all lower than 50% for European languages (Dutch, Polish and French) and higher than that for Asian languages (Thai, Arabic and Vietnamese). Besides, the Hybrid-SAT shows better improvement on Asian languages than European languages compared to Hybrid-SI, which decreases the WER_{SI} by 4.20%-7.95% for Asian languages while only 2.47%-3.07% for European languages.

In the contrast, the activation function adaptation methods show opposite impacts on these two kinds of L1s. For European languages, the performances of LHUC and *p*-Sigmoid are in narrow difference from Hybrid-SAT systems. The WER_{LHUC} of Dutch, Polish and French are only 0.08%, 0.25% and 0.75% higher than SAT. The corresponding *p*-Sigmoid performs slightly worse than LHUC but still close to SAT, where $\text{WER}_{p\text{Sigmoid}}$ are 0.57%(Dutch), 0.51%(Polish) and 1.08%(French) higher than SAT. However, for Asian languages, the difference between Hybrid-SAT and activation function adaptation is relatively larger. The WER_{LHUC} for Thai, Arabic and Vietnamese is 1.59%, 2.43%, and 5.08% higher than SAT, and the corresponding $\text{WER}_{p\text{Sigmoid}}$ is 1.70%, 2.32% and 4.61% higher.

Above all, the activation function adaptation has comparable performance to SAT when obvious mismatches between L1s exist and the WER of the supervision is low, but it shows very limited improvement or even worse performance when the WERs of the supervision are relatively high or there is only a small mismatch of the L1 between the training and test data.

Adaptation to Hybrid-SAT

Due to the performance of activation function adaptation and speaker adaptive training (SAT), it would be interesting to investigate if LHUC is applied to the Hybrid-SAT system [25]. The frame accuracy curves during SAT+LHUC adaptation on BLXXXeval1, BLXXXeval2 and

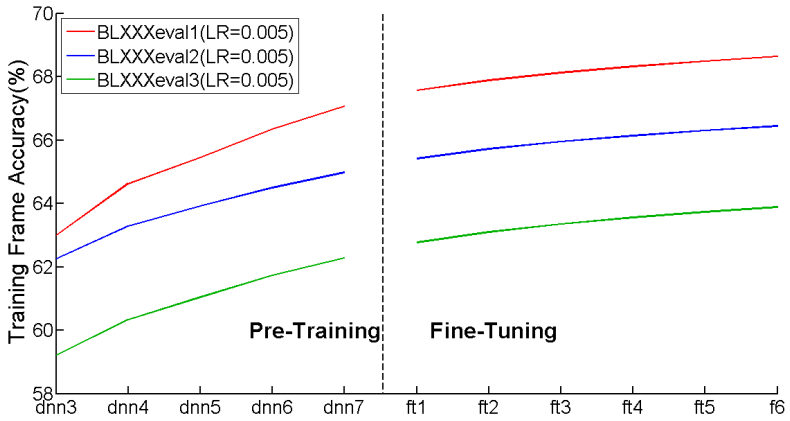


Fig. 5.2 Training frame accuracy(%) of SAT+LHUC.

Datasets	L1s	SI	SI+SAT	SAT	SAT+LHUC
BLXXXeval1	Gujarati	37.46	38.65	35.20	37.25
BLXXXeval2	Spanish	55.12	53.29	50.89	50.28
BLXXXeval3	Multi-L1s	52.59	49.83	48.02	47.25

Table 5.4 Trigram WERs(%) on BLXXXeval1, BLXXXeval2 and BLXXXeval3 of SAT+LHUC adaptation.

BLXXXeval3 are plotted in Figure 5.2 and the corresponding scoring results are shown in Table 5.4.

The adaptation process and decoding results of SAT+LHUC show similar trends to SI+SAT: The learning curves all gradually increase along the pre-training and fine-tuning process, and the $\text{WER}_{\text{SAT+LHUC}}$ becomes worse than WER_{SAT} on BLXXXeval1 while declines on BLXXXeval2 and BLXXXeval3. Furthermore, by plotting the absolute relative WER ($\text{WER}_{\text{SI+LHUC}} - \text{WER}_{\text{SI}}$ and $\text{WER}_{\text{SAT+LHUC}} - \text{WER}_{\text{SAT}}$) in Figure 5.3, it can be clearly seen that SAT+LHUC shows stronger negative influence on BLXXXeval1 and weaker improvement on BLXXXeval2 and BLXXXeval3 than SI+SAT. It is because the Hybrid-SAT has already reduced the mismatch between the training and test data, which makes the negative effect of low-level speakers counts more.

5.1.2 Feature Augmentation using I-Vectors

As discussed in Section 4.2.2, feature augmentation using i-vectors was implemented. In order to give an intuition of the effect of i-vectors, original features and augmented feature with i-vectors (*speaker-adapted* (SA) feature) were visualised on a 2D plan in Figure 5.4. To be specific, one arbitrary speaker selected from the training dataset BLXXXtrain was

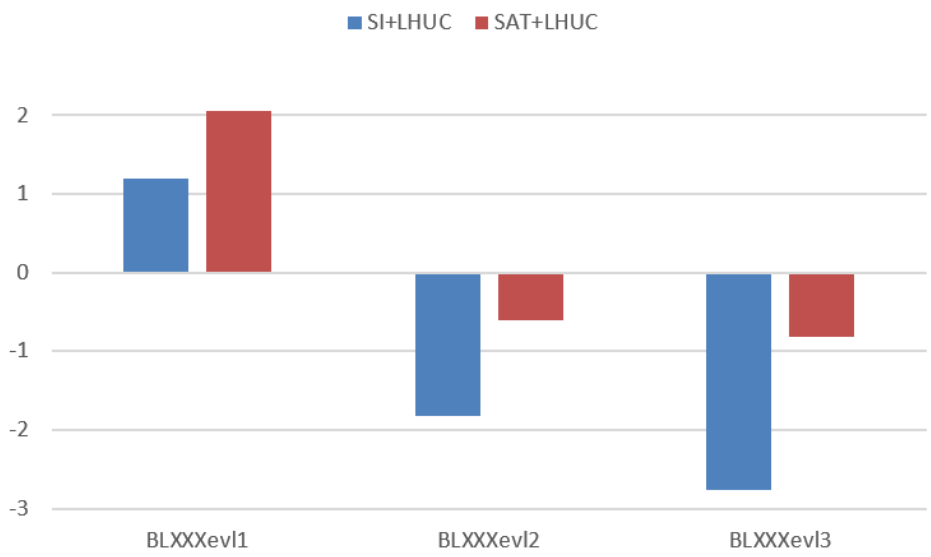
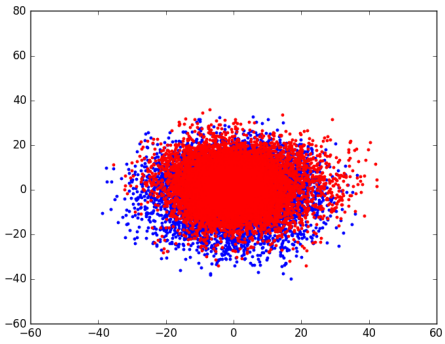


Fig. 5.3 Absolute relative WER(%) of SI+LHUC to SI and SAT+LHUC to SAT.

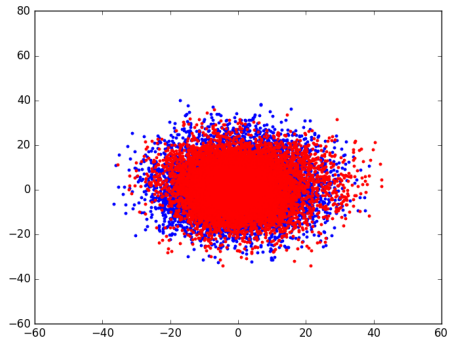
compared with another one selected from BLXXXtrain, BLXXXeval1 and BLXXXeval3 respectively. 10-dimensional i-vectors were taken as an example here. The original features and speaker-adapted features were projected to 2-dimensional vectors using PCA. The horizontal (the 1st dimension) and the vertical (the 2nd dimension) coordinates represent the major variations between the two speakers.

It can be seen that there is discrepancy in the feature space between the two speakers in Figure **??**, which indicates the mismatch of the two speakers. The more the discrepancy is, the larger the variability exists between the two speakers. Specifically, in Figure 5.4e, as the training data are all Indian Gujarati speakers while the BLXXXeval3 data are multi-L1 speakers, it turns up to obtain a relatively large non-overlapping area. Meanwhile, it can be noticed that the non-overlapping area in Figure 5.4f shrinks along the horizontal axis compared to Figure 5.4e. It reveals that the variability between the two speakers is reduced after concatenating i-vectors with the original acoustic features. On the other hand, in Figure 5.4a- 5.4d, the feature points of the two speakers almost cover the same area, and the reduction of non-overlapping areas after combining i-vectors is also small. Because the two speakers in BLXXXtrain and BLXXXeval1 are both Indian Gujarati speakers, the original distortion between them is already tiny, and i-vector augmentation makes little difference.

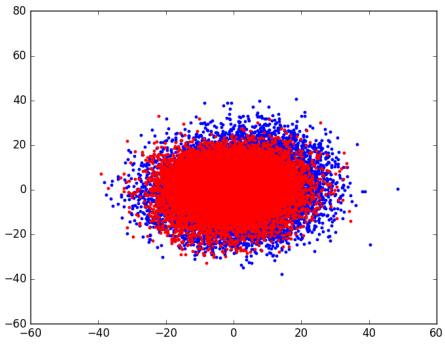
The decoding results of DNNs after the feature augmentation using CE criterion with 10- and 30-dimensional i-vectors are shown in Table 5.5. It can be found that for BLXXXeval1, the use of 10- and 30-dimensional i-vectors slightly reduces the WERs of Hybrid-SI by 0.31% and 0.34%, but they are 1.29% and 1.20% worse than Hybrid-SAT. If the dimension



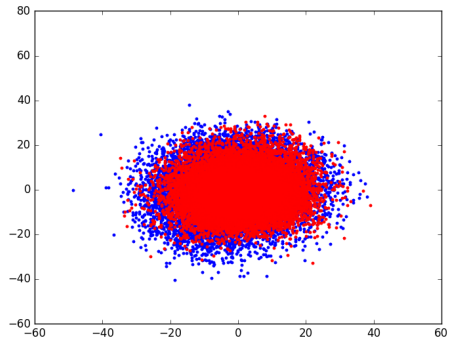
(a) BLXXXtrain vs. BLXXXtrain (original)



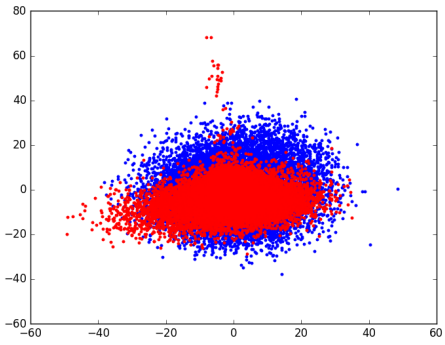
(b) BLXXXtrain vs. BLXXXtrain (+iv)



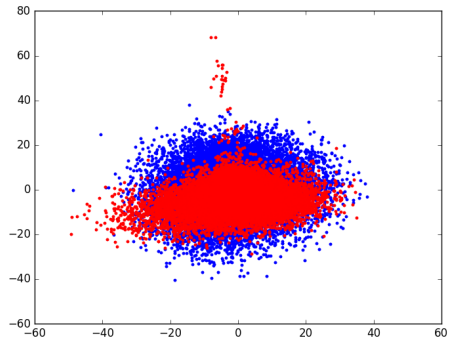
(c) BLXXXtrain vs. BLXXXeval1 (orignial)



(d) BLXXXtrain vs. BLXXXeval1 (+iv)



(e) BLXXXtrain vs. BLXXXeval3 (original)



(f) BLXXXtrain vs. BLXXXeval3 (+iv)

Fig. 5.4 The blue and red points represents features of two arbitrary speakers in BLXXXtrain training dataset and in BLXXXeval1 and BLXXXeval3 respectively.

Dataset	L1s	CE Training				Speakers
		SI	SAT	+i-vector		
				10-dim	30-dim	
BLXXXeval1	Gujarati	40.81	39.21	40.50	40.46	223
BLXXXeval3	Multi-L1s	53.44	50.91	53.49	53.52	226

Table 5.5 Tri-gram WERs(%) of feature augmentation using i-vectors

of the i-vector increases, it may lead to better improvement on BLXXXeval1 according to current results. As for BLXXXeval3, the i-vector seems even worsen the ASR performance, where the WERs of 10- and 30-dimensional i-vector augmentation are 0.05% and 0.08% higher than Hybrid-SI. One major reason is concerned with the i-vector extraction model. As it is trained using Gujarati speakers, it may not produce the suitable i-vectors for the different L1 speakers in BLXXXeval3.

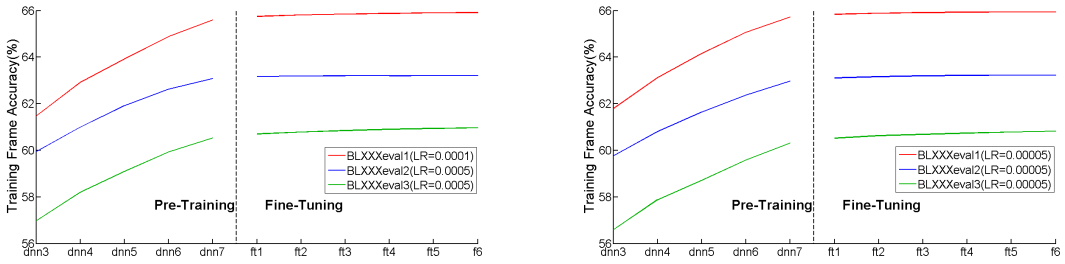
5.2 Language-Level Adaptation

Although the nature of speaker-level adaptation ensures the DNN model more robust to different speakers, it requires a number of data from every speaker. In practical, the speech data belonging to one speaker is very limited, which will lead the adaptation to the insufficient data issue and become over-fitting easily. In that case, language-level adaptation can be an alternative choice, which trains the set of SD parameters tied with each L1 rather than each speaker. It will make the adapted model more rough, but it is more practical to obtain more data from the each L1 rather than from each speaker. In other words, the language-level adaptation sacrifices its generality to obtain more data. As discussed in Section 4.3, experiments using activation function adaptation, LHUC and p-Sigmoid at language-level with the best learninig rates, were carried out.

5.2.1 Influence of L1 issues

The first part is to examine the performance of the language-level adaptation to different L1s. As explained in Section 4.3, the decoding hypotheses of BLXXXeval1, BLXXXeval2 and BLXXXeval3 from Hybrid-SI model were used as supervisions to adapt the Hybrid-SI model respectively. The frame accuracy curves with the best LRs for each dataset are drawn in Figure 5.5. These adapted models were then decoded on the corresponding dataset and scored. The scoring results are shown in Table 5.6.

It can be found that the language-level adaptation in this experiment on BLXXXeval1, BLXXXeval2 and BLXXXeval3 does not improve the recognition performance obviously,



(a) LHUC **(b)** p-Simgoid

Fig. 5.5 Training frame accuracy(%) of LHUC and p-Sigmoid adaptation using BLXXXeval1, BLXXXeval2 and BLXXXeval3 with the best LRs at language level.

Datasets	**L1s**	**SI**	**SAT** (speaker-level)	**SI+LHUC** (language-level)	**SI+pSigmoid** (language-level)	**Speakers**
BLXXXeval1	Gujarati	37.46	35.20	40.68	41.00	223
BLXXXeval2	Spanish	55.12	50.89	58.77	58.37	220
BLXXXeval3	Multi-L1s	52.59	48.02	51.80	53.18	226

Table 5.6 Trigram WERs(%) on BLXXXeval1, BLXXXeval2 and BLXXXeval3 using different adaptation methods at language level.

and even causes much higher WER in most situations. One possible reason is that the language-level adaptation is difficult to be fine-tuned, as can be seen from Figure 5.5 that the learning curve stays the same at fine-tuning step, which has been explained in Section 4.3. It limits the effect of adaptation. Another reason is related to the quality of the supervision. Compared with speaker-level adaptation, the SD parameters in language-level are tied from L1s, which means all the speakers who have the same L1 contribute to training the SD parameters jointly. Because the number of L1s in the adaptation dataset is much smaller than that of speakers, the distribution of English levels in the dataset will have more significant influence on the adaptation performance at language-level than speaker-level. As shown in Figure 4.1, there are around 40% of the speakers in these datasets are A1 and A2 level (the interpretation of English levels is in Table 4.2). The supervision WERs of each level are shown in Table 5.6, which indicates that the 40% of the supervision has extremely high WERs in all the datasets. Such a large proportion of low-level high-WERs data in supervision hypotheses will have a negative influence to the adaptation performance, and result in higher WERs after adaptation. As for BLXXXeval1 dataset, although its supervision WER is relatively low, it suffers more severely negative influence from A1 and A2 level speakers as discussed in section 5.1.1.

As can be noted in Figure **??** that LHUC and p-Sigmoid language-level adaptation on BLXXXeval3 show similar WERs to Hybrid-SI model, which is relatively better than single

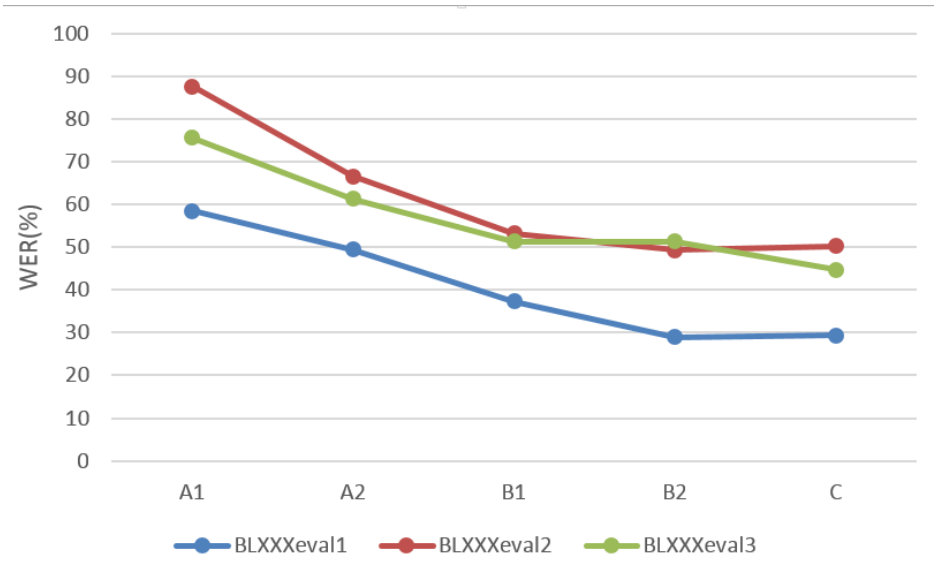


Fig. 5.6 WERs(%) of Hybrid-SI systems on BLXXXeval1, BLXXXeval2 and BLXXXeval3.

L1 datasets - BLXXXeval1 and BLXXXeval2. As discussed in Section 4.3, the language-level SD parameters **M** are trained using all the data belonging to different L1s. BLXXXeval3 contains 6 L1s, which means the **M** is trained and tuned according to 6 sets of data rather than one. It makes the adapted model obtain more generality. In addition, due to the nature of these L1s and scoring results demonstrated in section 5.1.1, the European languages and the Asian languages should share some inherent characteristics themselves, which implies the DNN can learn more information across different sets of data and offset the negative influence from the low-level speakers to some degree.

On the other hand, it would be interesting to investigate the adaptation performance of each L1 in BLXXXeval3. The scoring results divided by L1s are shown in table 5.7. The WERs of each L1 demonstrate consistent performance that the difference between Hybrid-SAT and activation function adaptation increases when the supervision WER increases, and p-Sigmoid is slightly worse than LHUC. Especially, it can be observed that for Dutch, its language-level adaptation shows quite similar performance to SAT, which is unusual compared to other L1s. One reason is related to its relatively low supervision WER, and another is that Dutch has an obviously smaller percentage of A1 and A2 level speakers(about 27% versus about 40% in other L1s) as illustrated in Figure 5.7. It confirms the impact of English level distributions on the language-level adaptation. For the other 5 L1s, their language-level adaptation performance is gradually far away from Hybrid-SAT and becomes even worse than Hybrid-SI for Arabic and Vietnamese.

L1	**SI**	**SAT** (speaker-level)	**SI+LHUC**	**SI+pSigmoid**	**Speakers**
			(language-level)		
Dutch	45.17	42.25	42.27	42.67	33
Polish	46.85	43.78	45.77	46.63	40
French	48.92	44.78	48.45	49.10	37
Thai	54.56	50.36	53.53	55.52	37
Arabic	59.13	53.84	59.99	62.33	40
Vietnamese	61.14	53.19	61.41	63.05	39
Overall	52.59	48.02	51.87	53.18	226

Table 5.7 Trigram WERs(%) on BLXXXeval3 after language-level activation function adaptation divided by L1s.

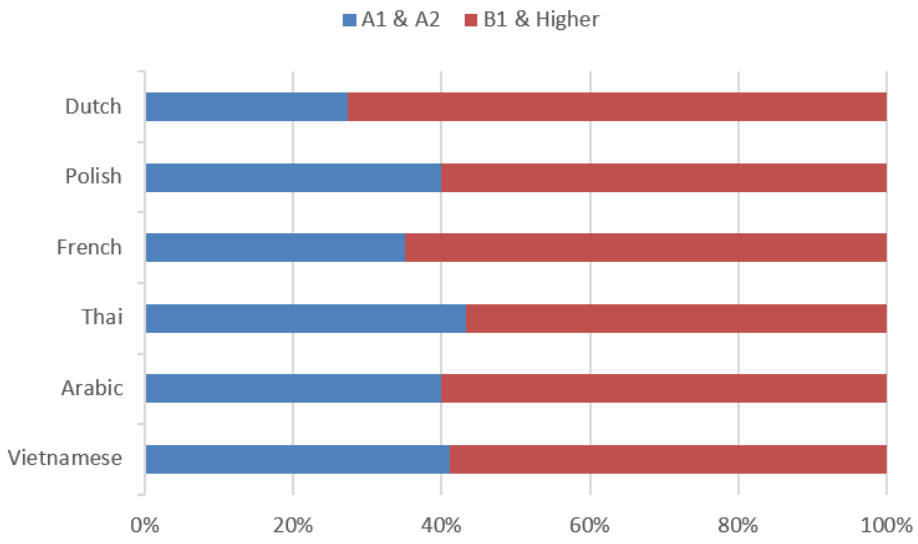
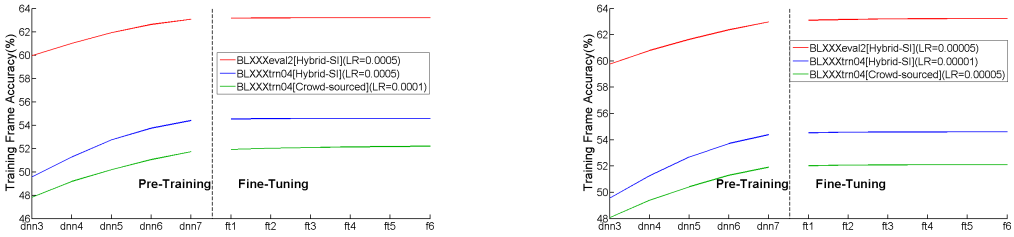


Fig. 5.7 English level distribution of L1s on BLXXXeval3.



(a) LHUC **(b)** *p*-Sigmoid

Fig. 5.8 Training frame accuracy(%) during LHUC and *p*-Sigmoid adaptation using different supervisions.

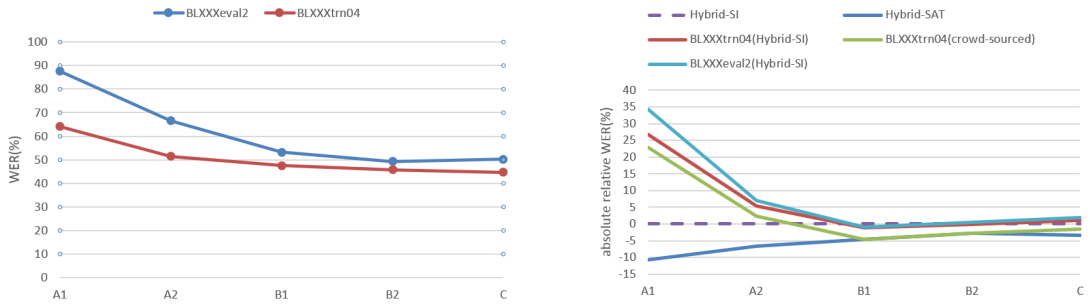
Datasets (Supervision)	**Supervision WER**	**SI**	**SAT** (speaker-level)	**SI+LHUC** (language-level)	**SI+pSigmoid** (language-level)	**Speakers**
BLXXXeval2 (Hybrid-SI)	55.12	55.12	50.89	58.77	58.37	220
BLXXXtrn04 (Hybrid-SI)	47.82			57.64	57.47	442
BLXXXtrn04 (Crowd-sourced)	-			54.48	54.64	442

Table 5.8 Trigram WER(%) on BLXXXeval2 from the adapted model using different supervisions. All the speakers are Latin American Spanish speakers.

5.2.2 Influence of supervision issues

From the previous section, it can be known that language-level adaptation is more sensitive to the WERs of the supervision than speaker-level, and it does not work on the dataset having the same L1 as training data (Gujarati). In this case, it will be useful to investigate the language-level adaptation performance with various-quality non-Gujarati supervisions. As explained in Section 4.3, in this section, BLXXXeval2 acted as the test dataset and three different supervisions were applied: Hybrid-SI decoding hypotheses of BLXXXeval2 and BLXXXtrn04, and crowd-sourced transcripts of BLXXXtrn04. All the speakers in BLXXXeval2 and BLXXXtrn04 are Latin American Spanish speakers, which are apparently mismatched with the training data. The frame accuracy curves during language-level adaptation are plotted in Figure ,and the scoring results are shown in Table 5.8.

Comparing the WERs of adapted models using BLXXXeval2 and BLXXXtrn04 Hybrid-SI decoding hypotheses as supervisions, it can be found that the language-level adaptation using BLXXXtrn04(Hybrid-SI) obtains relatively lower WER_{LHUC} and $\text{WER}_{p\text{Sigmoid}}$ than using BLXXXeval2(Hybrid-SI) by around 1.0%. The reason is that BLXXXtrn04 contains more speech data and lower supervision WERs than BLXXXeval2, confirming the language-level adaptation is sensitive to the supervision WERs. However, both of them show worse



(a) **(b)**

Fig. 5.9 Figure(a) shows the WER(%) of BLXXXeval2(Hybrid-SI) and BLXXXtrn04(Hybrid-SI) supervision divided by English levels. FIgure(b) shows the absolute relative WERs(%) of different supervisions to Hybrid-SI divided by English levels

WERs than WER_{SI}. It may be because that their supervision WERs are still too high to obtain reasonable adaptation improvement. Then, it is necessary to compare the last two rows, where the supervision of the last row is the crowd-sourced transcripts, which is assumed to be quite good quality. The WERs of the adapted models show that the crowd-sourced supervision results in a much better performance than the hypothesis supervision. The WER_{LHUC} and $\text{WER}_{p\text{Sigmoid}}$ using the crowd-sourced supervision are 0.64% and 0.48% lower than WER_{SI}, but 3.59% and 3.75% higher than speaker-level Hybrid-SAT.

Regarding to the English level issues discussed before, it is useful to further investigate the WER on different English-levels. As LHUC and *p*-Sigmoid indicate similar performance, LHUC is taken as an example. Figure 5.9 shows the supervision WERs and the absolute relative WERs with respect to the Hybrid-SI model divided by English levels. By comparing Figure 5.9a and 5.9b it can be found that the crowd-sourced supervision (the green line) demonstrates similar trend to hypothesis supervisions (the red and blue lines), where the WERs after LHUC adaptation become significantly worse than Hybrid-SI for A1 and A2 speakers. For B1 and higher levels, the adaptation performance becomes closer to SAT, especially for the crowd-sourced supervision. It implies that the WER of the supervision has influence on the adapted model, but the low-level speakers are the main reason that damages the performance of LHUC adaptation.

Chapter 6

Summary and Conclusions

This dissertation describes the work done in the ALTA MPhil project of automatic assessment of spoken English on improving the adaptation performance for non-native speakers. Several DNN-based unsupervised speaker adaptation approaches, such as activation function adaptation and feature augmentation using i-vectors, are implemented to a sate-of-art Hybrid-SI acoustic model and compared to SAT using CMLLR. Their performance to different L1s and different supervisions are also examined. Experiments are carried out at speaker-level and language-level separately, to investigate the difference between these two levels.

At speaker level, the results of activation function adaptation indicate that it worsen the WERs when there is no mismatch of L1s between training and test data. For the dataset containing different L1s from the training data, activation function adaptation shows considerable WER reduction to Hybrid-SI by around 1.8% to 3.3%, and its performance strongly relies on the supervision WERs. When compared to SAT, the WERs after speaker-level activation function adaptation are able to reach the similar level as SAT, but it requires L1-mismatched datasets and relatively low WERs (around 45%). In addition, we also demonstrate experiments of training DNNs with augmented features using i-vectors under CE criterion. The visualisation of augmented features proves the ability of i-vectors to reduce the inter-speaker variability. However, the DNN model trained by augmented features produces even worse WERs for the data with different L1s from training data, which reveals that the i-vector model cannot produce suitable i-vectors for unseen L1 speakers. As for matched L1s, the i-vector with 10 and 30 dimensions can slightly reduce the WERs by 0.05% and 0.08% respectively when compared with Hybrid-SI. It can be expected that larger degree of WER reduction can be achieved by applying higher dimensional i-vectors, and MPE training to DNNs, but further experiments are needed.

At language level, the SD parameters are trained from all data belonging to each L1, making it easier to acquire more adaptation data. Activation function adaptation at this

level is first implemented using BLXXXeval1, BLXXXeval2 and BLXXXeval3, which has the same configuration as speaker level. For the L1 matched dataset (BLXXXeval1), the adaptation causes higher WERs again, indicating that this kind of adaptation cannot reduce the WERs for datasets having the same L1 as training data in our experimental environment. The results from L1-mismatched datasets(BLXXXeval2 and BLXXXeval3) reveal that language-level adaptation is more sensitive to supervision WERs, and it obtains better performance to the multi-L1 dataset(BLXXXeval3). Especially, for the multi-L1 dataset, language-level adaptation can produce similar performance to SAT on particular L1s with low supervision WERs. Then the exploration of the influence of different supervisions is demonstrated. Comparison between hypotheses and crowd-sourced supervisions indicates that language-level adaptation strongly relies on the distribution of speakers' English levels. Even for crowd-sourced supervision, the part of data with low-level speakers(A1 and A2) cannot see the WER reduction. For higher-level speakers(B2 and higher), the adaptation performance is reasonable and can get closer to SAT with better-quality supervisions.

However, there are several limitations in this project. First of all, due to the limit of datasets, it is impossible to examine the performance of activation function adaptation using data containing sufficient high-level only Gujarati speakers. This kind of exploration can prove if the activation function adaptation can work for L1-matched datasets with only high-level speakers. As for the i-vector based techniques, higher dimensional i-vectors should be applied to see if the increasing of dimension can further improve the adaptation performance. MPE training is also expected to be applied to DNNs, which can provide a consistent performance to be compared with activation function adaptation methods. Moreover, the factorised feature transform using i-vectors is not implemented in this project, and it will be interesting to achieve it and compare its performance with the other adaptation approaches.

Bibliography

[1] LR Bahl, Peter F Brown, Peter V De Souza, and Robert L Mercer. Maximum mutual information estimation of hidden markov model parameters for speech recognition. In *proc. icassp*, volume 86, pages 49–52, 1986.

[2] J. K. Baker. The dragon system - an overview. *IEEE Transactions on Acoustics, Speech and Signal Processing*, 23(1):24–29,, 1975.

[3] Benjamin Chigier. Automatic speech recognition, June 10 1997. US Patent 5,638,487.

[4] Arthur P Dempster, Nan M Laird, and Donald B Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the royal statistical society. Series B (methodological)*, pages 1–38, 1977.

[5] Marc Ferras, Cheung Chi Leung, Claude Barras, and Jean-Luc Gauvain. Constrained mllr for speaker recognition. In *2007 IEEE International Conference on Acoustics, Speech and Signal Processing-ICASSP'07*, volume 4, pages IV–53. IEEE, 2007.

[6] Mark Gales. Acoustic factorisation for speech recognition and speech synthesis. 2012.

[7] Mark Gales and Steve Young. The application of hidden markov models in speech recognition. *Foundations and trends in signal processing*, 1(3):195–304, 2008.

[8] Mark JF Gales. Maximum likelihood linear transformations for hmm-based speech recognition. *Computer speech & language*, 12(2):75–98, 1998.

[9] Juri Ganitkevitch. Speaker adaptation using maximum likelihood linear regression. In *Rheinish-Westflesche Technische Hochschule Aachen, the course of Automatic Speech Recognition, www-i6. informatik. rwthaachen. de/web/Teaching/Seminars/SS05/ASR/Juri Ganitkevitch Ausarbeitung. pdf*. Citeseer, 2005.

[10] J-L Gauvain and Chin-Hui Lee. Maximum a posteriori estimation for multivariate gaussian mixture observations of markov chains. *IEEE transactions on speech and audio processing*, 2(2):291–298, 1994.

[11] Vaibhava Goel and William J Byrne. Minimum bayes-risk automatic speech recognition. *Computer Speech & Language*, 14(2):115–135, 2000.

[12] H. Hermansky. Perceptual linear prediction of speech. *Journal of the acoustic society of America*, 87(4):1738–1752, 1990.

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

[14] Sadik Kapadia, Valtcho Valtchev, and SJ Young. Mmi training for continuous phoneme recognition on the timit database. In *Acoustics, Speech, and Signal Processing, 1993. ICASSP-93., 1993 IEEE International Conference on*, volume 2, pages 491–494. IEEE, 1993.

[15] Penny Karanasou, Yongqiang Wang, Mark JF Gales, and Philip C Woodland. Adaptation of deep neural network acoustic models using factorised i-vectors. In *INTERSPEECH*, pages 2180–2184, 2014.

[16] Christopher J Leggetter and Philip C Woodland. Maximum likelihood linear regression for speaker adaptation of continuous density hidden markov models. *Computer Speech & Language*, 9(2):171–185, 1995.

[17] Yajie Miao, Hao Zhang, and Florian Metze. Speaker adaptive training of deep neural network acoustic models using i-vectors. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 23(11):1938–1949, 2015.

[18] Quoc Bao Nguyen, Jonas Gehring, Kevin Kilgour, and Alex Waibel. Optimizing deep bottleneck feature extraction. In *Computing and Communication Technologies, Research, Innovation, and Vision for the Future (RIVF), 2013 IEEE RIVF International Conference on*, pages 152–156. IEEE, 2013.

[19] Tsubasa Ochiai, Shigeki Matsuda, Xugang Lu, Chiori Hori, and Shigeru Katagiri. Speaker adaptive training using deep neural networks. In *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6349–6353. IEEE, 2014.

[20] Douglas Reynolds. Universal background models. *Encyclopedia of Biometrics*, pages 1547–1550, 2015.

[21] Lahiru Samarakoon and Khe Chai Sim. Learning factorized feature transforms for speaker normalization. In *2015 IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*, pages 145–152. IEEE, 2015.

[22] Andrew Senior and Ignacio Lopez-Moreno. Improving dnn speaker independence with i-vector inputs. 2014.

[23] Michael L Shire. Relating frame accuracy with word error in hybrid ann-hmm asr. In *INTERSPEECH*, pages 1797–1800, 2001.

[24] Pawel Swietojanski, Jinyu Li, and Steve Renals. Learning hidden unit contributions for unsupervised acoustic model adaptation. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 24(8):1450–1463, 2016.

[25] Pawel Swietojanski and Steve Renais. Sat-lhuc: Speaker adaptive training for learning hidden unit contributions. In *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5010–5014. IEEE, 2016.

[26] Pawel Swietojanski and Steve Renals. Learning hidden unit contributions for unsupervised speaker adaptation of neural network acoustic models. In *Spoken Language Technology Workshop (SLT), 2014 IEEE*, pages 171–176. IEEE, 2014.

[27] L Wang and PC Woodland. Discriminative adaptive training using the mpe criterion. In *Automatic Speech Recognition and Understanding, 2003. ASRU'03. 2003 IEEE Workshop on*, pages 279–284. IEEE, 2003.

[28] Chunyang Wu, Penny Karanasou, and Mark JF Gales. Combining i-vector representation and structured neural networks for rapid adaptation. In *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5000–5004. IEEE, 2016.

[29] Steve Young, Gunnar Evermann, Mark Gales, Thomas Hain, Dan Kershaw, Xunying Liu, Gareth Moore, Julian Odell, Dave Ollason, Dan Povey, Anton Ragni, Valtcho Valtchev, Phil Woodland, Chao Zhang, et al. The htk book (3.5 alpha (1)). *Cambridge University*, 2015.

[30] Kai Yu. *Adaptive training for large vocabulary continuous speech recognition*. PhD thesis, Citeseer, 2006.

[31] C Zhang and PC Woodland. Dnn speaker adaptation using parameterised sigmoid and relu hidden activation functions. In *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5300–5304. IEEE, 2016.

[32] Yu Zhang, Ekapol Chuangsuwanich, and James R Glass. Extracting deep neural network bottleneck features using low-rank matrix factorization. In *ICASSP*, pages 185–189, 2014.