Knowledge Constraints in Speaker Adaptation

Xintian Wu

M.S., Electronic Engineering, Tsinghua University, 1996
B.E., Electronic Engineering, Tsinghua University, 1993

A dissertation submitted to the faculty of the
Oregon Graduate Institute of Science and Technology
in partial fulfillment of the
requirements for the degree
Doctor of Philosophy
in
Computer Science and Engineering

April 2000

The dissertation “Knowledge Constraints in Speaker Adaptation” by Xintian Wu has been examined and approved by the following Examination Committee:

Yonghong Yan
Associate Professor
Thesis Research Adviser

Peter A. Heeman
Assistant Professor

Hynek Hermansky
Professor

Robert Yung
Ph.D.
Director of Intel China Research Center

Dedication

To those who establish the mathematical foundations

Acknowledgements

First and foremost, I would like to express my sincere gratitude to my advisor Dr. Yonghong Yan for his invaluable guidance, support, and encouragement. His attention to detail has taught me how to record and express my thoughts accurately. His insight, perspective, and vision are the continuous subject of my admiration.

Special thanks to Dr. Peter Heeman, who constantly offered his time to help me improve my research skills, and even my communication skills. Talking with him was fruitful and discussing my research problems with him often stimulated my new research ideas.

I would also like to thank my thesis committee members, Dr. Yonghong Yan, Dr. Peter Heeman, Dr. Hynek Hermansky, and Dr. Robert Yung, for reviewing my thesis and giving me precious feedback on how to improve my thesis.

My appreciation goes to the large vocabulary continuous speech recognition group members, former and present: Dr. Yonghong Yan, Johan Schalkwyk, Dr. ChangHyuck Oh, DongHwa Kim, Chaojun Liu, and Chengyi Zheng. Without the collective hard work, I would not have a competitive baseline system to finish my experiments. The experience I have earned from building such a large-scale system was a great help in stimulating my research.

Thanks to the faculty and students in the Center for Spoken Language Understanding for their continuous encouragement and support, which have made my research easier.

Thanks to Intel, ONR, Fonix, and CSLU member companies for supporting the research.

Contents

List of Tables

List of Figures

Abstract

Knowledge Constraints in Speaker Adaptation

Xintian Wu, B.E., M.S.

Ph.D., Oregon Graduate Institute of Science and Technology
April 2000

Thesis Advisor: Dr. Yonghong Yan

In speech recognition research, transformation-based adaptation algorithms provide an effective way of adapting acoustic models to more closely match the test-speaker acoustics thereby improving the recognition performance. However, when only limited amounts of adaptation data are available, the transformation is often poorly estimated, which causes performance degradation.

This thesis presents the Markov Random Field Linear Regression (MRFLR) algorithm, which constrains the adaptation by the correlations among acoustic parameters. The correlations are estimated from the training corpus and hypothesized as prior knowledge of acoustic models. By explicitly incorporating them into adaptation, robust and fast adaptation can be achieved. The Markov Random Field theory is used to model the correlations.

The hypothesis is tested by comparing MRFLR with Maximum Likelihood Linear Regression (MLLR), a widely used transformation-based adaptation algorithm. Experimental results show that MRFLR outperforms MLLR when adaptation data are sparse, and converges to the MLLR performance when more adaptation data are available.

Chapter 1

Introduction

Speech is the primary form of human communication. When speaking in a noisy room, talking on the phone, or meeting with different people with different dialects, a native speaker seldom has any difficulty in understanding others after exchanging a few sentences. This is nature's example of adaptation. The human brain serves as a wonderful adaptor without demanding any host attention.

However, when it comes to computer speech recognition, it is very difficult to deal with variations in speech. Often, a speaker change, or even an emotional change in the same speaker, causes recognition performance to degrade.

Speech variations come from many sources: background noise, channel condition, music, speaker emotion, dialects, disfluency, etc. Roughly, adaptation can be done in two directions: *Environment Adaptation* and *Speaker Adaptation*. The former focuses more on the background of speech and the later emphasizes the speech and the speaker. This thesis focuses on speaker adaptation.

1.1 Speech Recognition

Speech Recognition research originated as early as the 1950's and has been a continuous effort for the past fifty years due to its widespread use in a variety of applications including information retrieval, data entry, and general man-machine communication.

The performance of speech recognition systems has improved dramatically in the past few years. Figure 1.1 shows some bench-mark results from workshops that NIST (National Institute of Science and Technology) has been sponsoring since 1988 for promoting speech

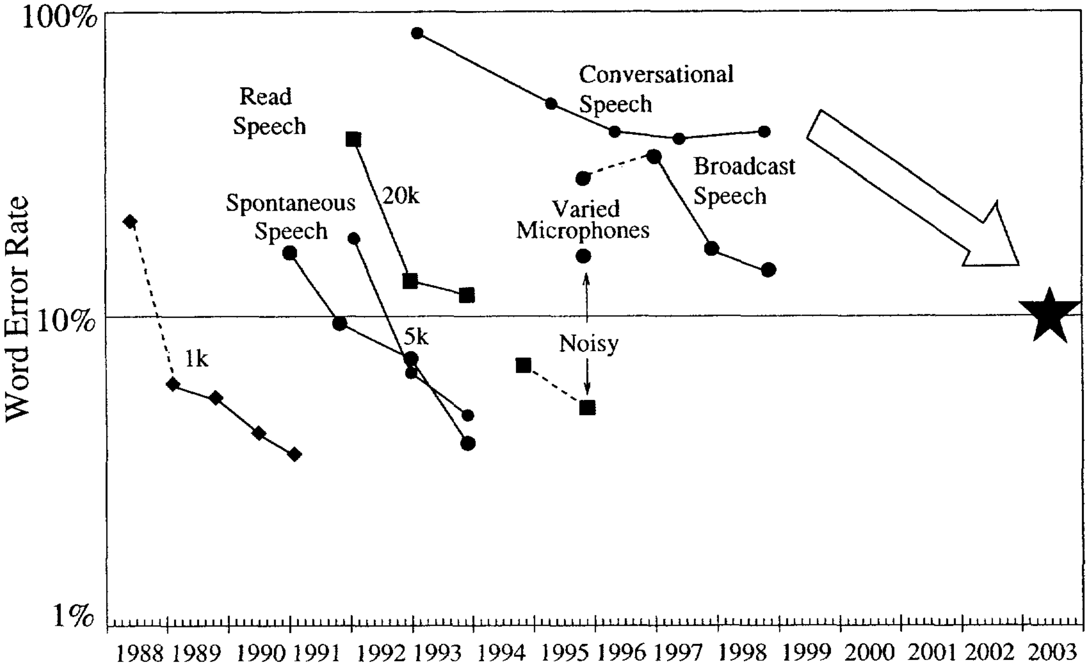

Figure 1.1: History of speech recognition benchmark tests

Rapid development is achieved in the speech recognition field. By the year 2003, the performance of speech recognition systems on the Conversational Speech task is expected to reach an acceptable WER of 10%.

recognition research.

In 1988, The *Resource Management* (RM) task, indicated by '1k' in the figure, was considered difficult for continuous speech recognition. The RM task is a 1000-word recognition task set in a quiet environment. The test utterances have restricted patterns such as commands and general descriptions. The *Word Error Rate* (WER) was about 20%. Three years later, the CMU system [17] successfully pushed the WER down to 3.6%. The same rapid improvement was achieved on the *World Street Journal* (WSJ) task, both for the read speech 20,000-word task and the spontaneous speech 5000-word task, indicated by '20k' and '5k' in the figure respectively. These two tasks use corpora of WSJ news text. The HTK system [42] took top honors.

In recent years, the *Broadcast News* task and the *Conversational Speech* task, which mix daily conversations with background noise, music, and commercials, have become the next two challenging targets to be conquered. The recognition performance on the

Conversational Speech task is expected (by NIST) to reach an acceptable WER of 10% by 2003.

While the research systems used in these competitions remain too complicated (regarding the memory requirement or running time) to move out of the laboratories, simplified real-time systems already have found their way into the dictation market, such as IBM's *via Voice*, and Dragon's *Naturally Speaking*.

1.2 System Structure

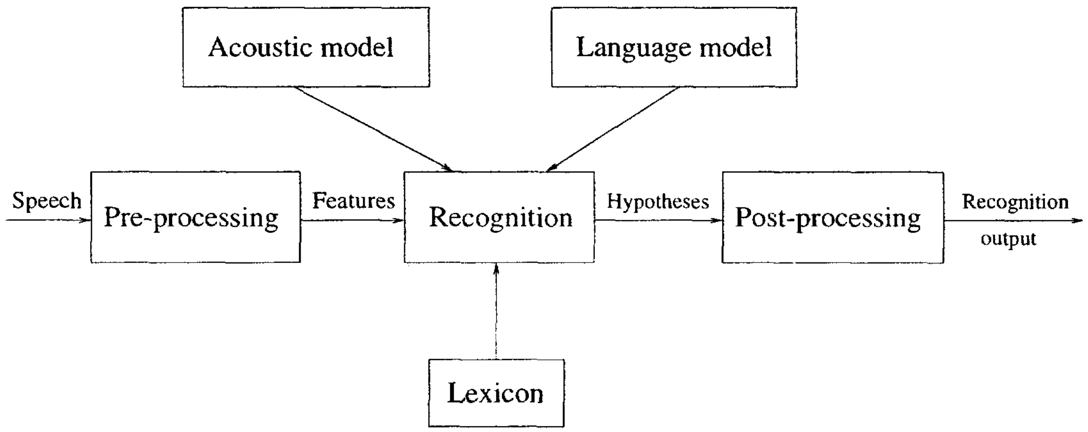

Figure 1.2: Structure of a speech recognition system

Speech sounds are first converted to a sequence of features and then the recognition is performed to find the hypotheses that best match the speech.

The structure of a conventional speech recognition system is shown in Figure 1.2. It usually includes:

1. Pre-processing: Feature generation

   In this step, a speech utterance is converted into a sequence of *speech features*, which are used as observations in the mathematical modeling of acoustics. Ideally, the speech features should contain as much information as possible about the linguistic content of the speech while being reasonably compact and free of redundant detail. Perhaps one of the most widely used speech features is *Mel-Frequency Cepstrum Coefficients* [9] (MFCC). The cepstrum coefficients are obtained by *Inverse Fourier*

Transformation of the logarithm of the speech spectrum. The mel-scale warping is done on the spectrum to reflect the non-linear perceptual characteristics of human hearing.

For some systems, the pre-processing step may also include segmentation, channel and gender detection, and speaker clustering.

Segmentation is the task of chopping long utterances into short ones for a better control of later recognition steps. Usually the speech recognizer is not capable of handling very long utterances due to the memory and speed restrictions.

Channel and gender detection is for those systems that use multiple acoustic models. Usually each acoustic model corresponds to one channel or gender condition. The acoustic model is a set of model parameters that describe the acoustic properties of phonemes, words, or utterances. The channel and gender detection step tags each test utterance for an appropriate acoustic model to use in the later steps.

Speaker clustering is used to gather together more speaker dependent data for a better adaptation performance (discussed later in Section 1.4). If speaker information is unknown for the test utterances, a speaker detection step is required.

2. Speech recognition: Hypotheses search

   In this recognition step, the speech recognizer performs a massive search to find the most likely word-sequence hypotheses that match the acoustic observations. The match is based on the acoustic model and constrained by the language model and the lexicon. The language model consists of the probabilities of word sequences. For example, the popular trigram language model includes the occurrences of most frequent word triples and word pairs. The lexicon is a list of vocabulary words in the system and their pronunciations.

3. Post-processing: Recognition result

   This optional post-processing step performs a refinement of the recognition hypotheses. The techniques used might include a high-order language-model re-scoring, or a second recognition pass with a more-detailed acoustic model and language model.

1.3 The Acoustic Model

The performance of a speech recognition system varies with the acoustic model. Although the language model and the lexicon play an important role in large vocabulary systems, this thesis focuses on improving the acoustic model accuracy.

For modeling common characteristics of many speakers, the acoustic model is usually trained with utterances from a large number of speakers. A *Speaker Independent* (SI) model is less sensitive to speaker variations. This speaker independent property of the acoustic model is desired for applications facing more than a single speaker since training one SD model for each speaker is impossible in these applications.

However, the performance of a SI model is not as satisfactory as that of a *Speaker Dependent* (SD) model. The reasons for this are as follows:

- The SI model represents only an average property of all the speakers in the training set. In other words, the model is not particular to any of the speakers. If a model could be trained particularly for a test speaker, the performance would usually be better than that of the SI model. For example, in the Febuary 91 ARPA RM1 (Resource Management) Benchmark test, the best performance for the SI system is 3.6% WER (CMU [17]) while the best performance for the SD system is 1.8% WER (MIT [30]).

- The performance of a SI system degrades dramatically if the test speaker is different from any of the speakers in the training set. Moreover, it is often impossible to cover all the possible speakers in a training set of practical size. For example, in the *WSJ Switchboard* 5000-word benchmark tasks, the performances of native speaker (English) tasks ranges from 8% WER to 20% WER while the performances of non-native speaker tasks range from 20% WER to 30% WER. There is a major degradation in performance with different speaker conditions.

Therefore, to improve performances of SI systems, a reasonable approach might be to dynamically adapt the SI model to each test-speaker according to some enrollment data. The adapted model is then used in the final recognitions.

1.4 Speaker Adaptation

Speaker adaption is the term used to describe the suite of techniques which modify the acoustic model to more closely match the test-speaker. Speaker adaption has been widely used in speech systems facing a large or changeable user base.

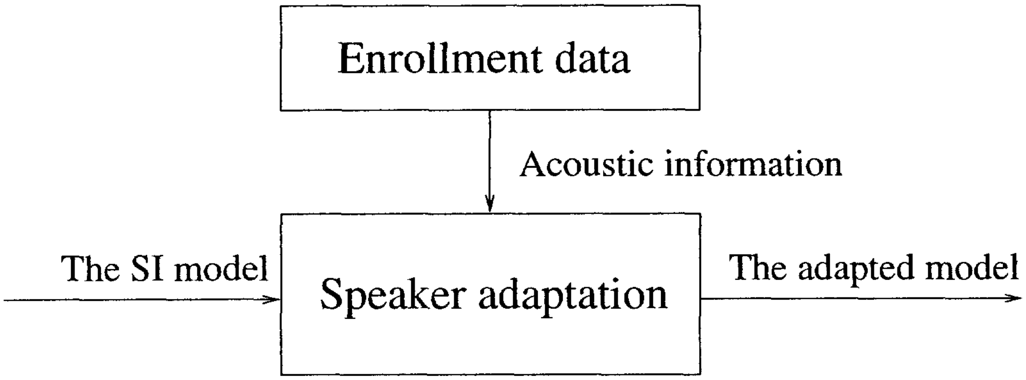

Figure 1.3: Speaker adaptation block diagram

Speaker information is extracted from the enrollment data and the SI model is updated to more closely match the test speaker.

The process of speaker adaptation is like a rapid training session, as illustrated in Figure 1.3. Given some enrollment data (also called adaptation data: a set of utterances from the test speaker and the corresponding references), acoustic information about the test speaker is extracted and the SI model is adapted towards the test speaker. The enrollment data for speaker adaptation can be obtained either from a "speaker training" session, in which the speaker is directed to say certain sentences, or from a previous recognition pass on the test data.

Unlike traditional training algorithms, which rely on a large amount of training data to update each model parameter individually, speaker adaptation algorithms make a broad adjustment of model parameters based on the limited amount of adaptation data. For example, the *Maximum Likelihood Linear Regression* (MLLR) algorithm estimates a transformation matrix from the adaptation data and then updates all the model parameters. Hence, the transformation matrix requires a robust estimation for its wide touch on the model parameters.

Due to the large number of parameters in an acoustic model, existing speaker adaptation algorithms still require quite a lot of adaptation data to ensure an accurate estimation. When there are not enough data to provide such an estimation, a negative adaptation effect may result.

Research on improving speaker adaptation can be done in the following two directions: (1) Extract more test-speaker information from the limited amount of adaptation data. (2) Incorporate prior knowledge of acoustic models into the adaptation. The latter is the topic of this thesis: *Knowledge Constraints in Speaker Adaptation.*

1.5 Knowledge Constraints

Acoustic parameters (or model parameters) considered as random variables are not independent of each other. These parameters are correlated with each other in various ways. The correlations mainly come from the following sources:

- Common structure of vocal tracts

  Humans have a common structure of vocal organs, such as oral tract, nasal tract and larynx, to produce sounds. This common physical structure results in the similarities of different people producing the same sounds, which is one of the reasons why people are able to communicate with each other. Therefore, if correlations exist in each SD model, there exists a common set of correlations among all the models. These correlations reflect the internal structure of speech sounds.

- Phonological rules, language, and dialect

  Phonological rules describe certain ways in which people pronounce words. According to these rules, some phonemes are more related to each other because it takes similar movements of the vocal organs to pronounce them. For example, in OGIBET (a phoneme system for English), phoneme /er/ (bird) and /axr/ (butter) are similar in terms of vocal movements. Therefore, there exists correlations among the model parameters that model these related phoneme acoustics.

These correlations among acoustic parameters are hypothesized as a representation of the internal structure of speech sounds, or prior knowledge of acoustic models. Assumed invariant among different environments, these correlations can be used in speaker adaptation to improve adaptation performance in the following two ways:

- If some parts of the acoustic parameters have enough data to be adapted, other parts of the parameters can also be adapted from the correlations.
- If more than one part of the parameters can be adapted. Their correlations can serve as a constraint on the possible outputs of the adaptation.

Intuitively, correlation-based adaptation (also called *Dependency Modeling* techniques) functions as follows: a local change of model parameters is made and then the change is smoothed to the neighbors, other model parameters determined by the correlations. For example, if phoneme /er/ is observed in the enrollment data, the model parameters that model phoneme /er/ are adapted. Furthermore, according to the correlations, the changes are smoothed to phoneme /axr/ so that model parameters that model phoneme /axr/ are also updated.

This scenario is different from those adaptation algorithms that rely only on the adaptation data to update the model parameters. Therefore, the use of prior knowledge in speaker adaptation is called *Knowledge Constraints.*

However, existing dependency modeling techniques do not meet the needs of fast adaptation since changes of model parameters are only made to observed phonemes in the enrollment data and their correlations. Due to of the complexity in modeling correlations, dependency modeling methods usually model only local correlations (correlations within a small neighborhood of a given phoneme). For an acoustic model that has millions of parameters, a large amount of adaptation data is required to make a significant change to the model.

1.6 This Thesis

In this thesis, various speaker adaptation algorithms and dependency modeling techniques have been studied. Existing problems of these techniques include:

1. Existing transformation-based adaptation algorithms make fast adaptation. However they require a large amount of adaptation data to robustly estimate the transformations. If the amount of adaptation data is insufficient, a negative adaptation effect may result.

2. Existing dependency modeling techniques make local modifications of model parameters. Thus they have a slow convergence to the test-speaker.

This thesis presents the *Markov Random Field Linear Regression* (MRFLR) algorithm, which is a transformation-based adaptation algorithm constrained by correlations. The contributions of this thesis are as follows:

1. It provides a novel way of incorporating the prior knowledge of acoustic models into the most widely used transformation-based adaptation method: MLLR.

2. It solves the difficulty that early attempts to combine these two techniques fail to find a closed-form solution for estimating the adaptation parameters.

The MRFLR algorithm has advantages of both fast adaptation and robust estimation. When there is only a limited amount of adaptation data available, the knowledge constraints in MRFLR improve the adaptation performance while normal adaptation methods often cause performance degradations.

Applications of the MRFLR algorithm are various speech dialogue systems that require a fast speaker adaptation. As an extension, the MRFLR adaptation can be also used to improve the SI model performance.

1.7 Organization of this Thesis

Section 2 gives a general description of modeling acoustics using *Hidden Markov Models*, the most widely used statistical approach for speech recognition. The discussion mainly focuses on the training part since the concepts and equations will be used in the later chapters.

Section 3 reviews some known speaker adaptation algorithms. The transformation-based adaptation method and dependency modeling techniques are presented in detail to show their relation to the MRFLR algorithm developed in this thesis.

Section 4 details the MRFLR algorithm. Section 5 outlines the baseline system for the evaluation of the MRFLR algorithm. Section 6 presents the experimental results. The evaluations are based on performance comparisons between MRFLR and MLLR on the WSJ task. Section 7 summarizes the thesis.

Chapter 2

Acoustical Modeling with HMMs

In speech recognition research, statistical methods are widely used to characterize both the time and spectral properties of speech sounds. A general description of the recognition procedure is to find a word string $W = w_1, w_2, ...w_L$ that maximizes the posterior probability of the string W given the speech observation $O = o_1, o_2, ...o_T$, that is:

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

The right hand side of Equation 2.1 can be re-written according to the *Bayes* rule:

$$P(W|O) = \frac{P(W)P(O|W)}{P(O)} \tag{2.2}$$

where $P(O)$ is the distribution of the speech observation.

Since $P(O)$ is irrelevant to the recognition and usually considered constant over the time period of interest, Equation 2.1 now becomes:

$$\hat{W} = \arg\max_{W} P(W)P(O|W) \tag{2.3}$$

The recognition procedure is a massive search over all the possible word sequences to find a word sequence $\hat{W}$ that maximizes Equation 2.3. $P(W)$ acts as a grammar constraint on the word sequence, and $P(O|W)$ measures how well the word sequence matches the observed speech sounds.

$P(O|W)$ is usually called the acoustic model. In large vocabulary systems, the acoustic modeling unit is usually each context triphone[1] rather than each word, as illustrated in

[1] Context triphones are phones that take into account their left and right context. The purpose of using triphones in the systems is to model the co-articulation effect of speech.

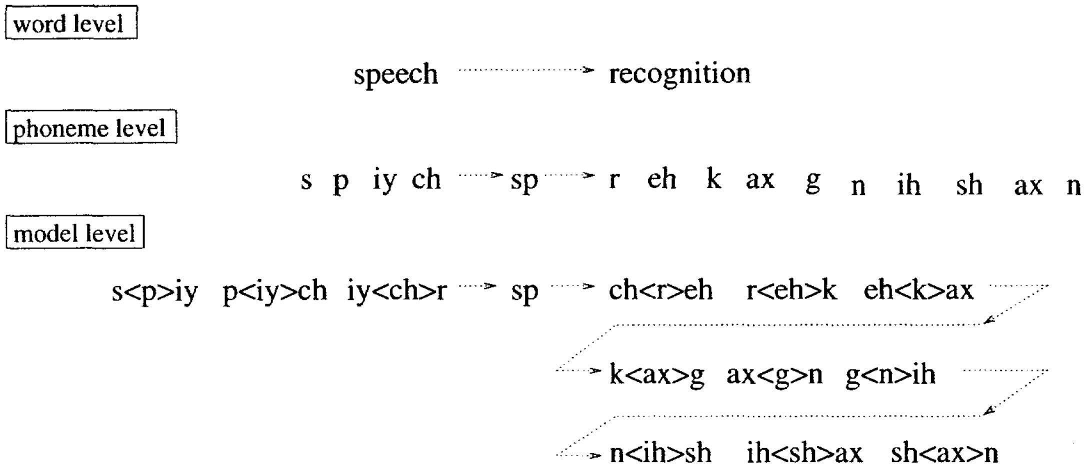

Figure 2.1: Three layers modeling of a sentence

The modeling unit is context triphones. They are obtained by first expanding each word in a sentence into its pronunciations and then constructing triphones according to the left and right context of each phoneme. The phoneme /sp/ represents an *optional* short pause between two words. Usually /sp/ is skipped when considering contexts.

Figure 2.1. A word is represented first by a series of phonemes according to its pronunciations and then by a series of context triphones. The construction of a triphone is to combine each phoneme with its left and right context. Each context triphone is modeled by a certain model. In the latter chapters, the acoustic model is denoted as λ following standard conventions.

2.1 Hidden Markov Model

For modeling acoustics, historically there are many techniques: dynamic template comparison, knowledge-based matching, neural network, and *Hidden Markov Model* (HMM). These techniques have their own advantages in certain applications. However, when the recognition tasks extend to unlimited vocabulary and an unconstrained environment, HMM is so far the most promising and effective approach. It is simple in theory and has efficient algorithms in both training and recognition.

A HMM is a finite state machine that changes its state once every time unit. Figure 2.2 illustrates a typical three-state HMM. (State 1 and 5 are pseudo states.) The transition

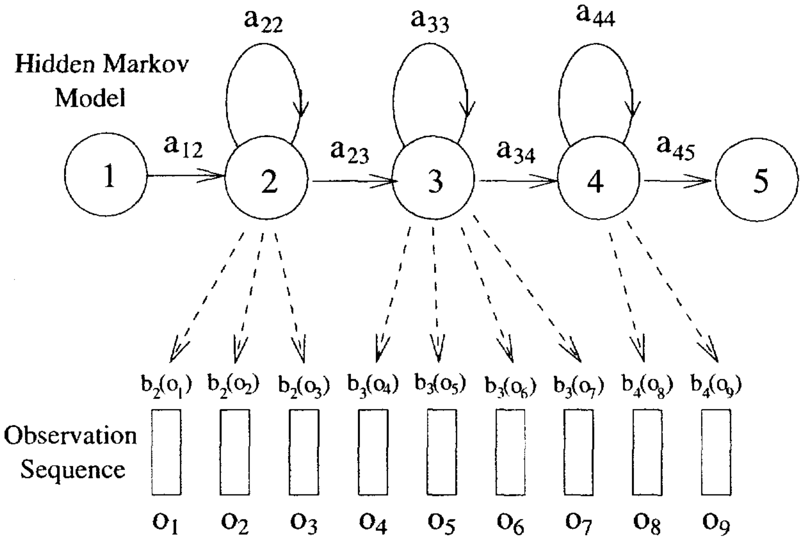
Hidden Markov Model
a22
a33
a44
a12
a23
a34
a45
1
2
3
4
5
b2(o1) b2(o2) b2(o3) b3(o4) b3(o5) b3(o6) b3(o7) b4(o8) b4(o9)
Observation Sequence
o1 o2 o3 o4 o5 o6 o7 o8 o9

Figure 2.2: The Hidden Markov Model

A HMM is a finite state machine with transition probability a_{ij} from one state i to another j. The observation sequence is generated with output probability density $b_j(o_t)$ represented by Gaussian mixture densities.

from state i to state j is determined by the probability a_{ij}. At each state j, a speech vector (observation) o_t is generated with probability density $b_j(o_t)$, represented by Gaussian mixture densities:

$$b_j(o_t) = \sum_{k=1}^{K_j} \omega_{jk} N(o_t;\, \mu_{jk},\, \Sigma_{jk}) \tag{2.4}$$

where K_j is the number of mixture components of state j, ω_{jk} is the weight of mixture component k of state j ($\sum_{k=1}^{K_j} \omega_{jk} = 1$), and $N(\cdot)$ is a multivariate Gaussian distribution with mean vector μ_{jk} and covariance matrix Σ_{jk}:

$$N(o_t;\, \mu_{jk},\, \Sigma_{jk}) = \frac{1}{\sqrt{(2\pi)^D \mid \Sigma_{jk} \mid}} \exp[-\frac{1}{2}(o_t - \mu_{jk})^T \Sigma_{jk}^{-1} (o_t - \mu_{jk})] \tag{2.5}$$

where D is the feature dimension and $(o_t - \mu_{jk})^T$ denotes the transposition of $(o_t - \mu_{jk})$.

There are two basic assumptions in the HMM framework:

1. The first-order Markov assumption enforces that the current system status (being at a state) depends only on its previous status: $P(q_t \mid q_{t-1}, q_{t-2}, ...q_0) = P(q_t \mid q_{t-1})$ where q_t stands for the system being at state q_t at time t.

2. The output independence assumption assumes that the output probability of a state depends only on the state at a certain time, regardless of when and how the state is entered.

Under these two assumptions, the probability that an observation sequence $O = o_1, o_2, ..., o_T$ is generated given the acoustic model λ moving through the state sequence $q = q_0, q_1, ..., q_T$ can be calculated as follows:

$$P(O, q \mid \lambda) = \prod_{t=1}^{T} a_{q_{t-1},q_t} b_{q_t}(o_t) \tag{2.6}$$

In practice, the state sequence q is hidden to us, which is why the model is called a hidden Markov model.

2.2 Recognition: The Viterbi Algorithm

The recognition computation includes the calculation of the likelihood function $P(O \mid \lambda)$, which measures how well the acoustic model matches the given speech observation. It is simply a summation over all the possible state sequences:

$$P(O \mid \lambda) = \sum_{q} P(O, q \mid \lambda) \tag{2.7}$$

Equation 2.7 can be computed efficiently by defining the forward probability (See the forward-backward algorithm [4]), which is the probability of the partial observation sequence $o_1, o_2, ...o_t$ and state i at time t, given the model λ:

$$\alpha_t(i) = P(o_1, o_2, ..., o_t, q_t = i \mid \lambda) \tag{2.8}$$

Therefore,

$$P(O \mid \lambda) = \sum_{i=1}^{N} \alpha_T(i) \tag{2.9}$$

where N is the number of states in the model.

Inductively, $\alpha_i(t)$ can be solved as follows:

$$\begin{aligned}
\alpha_1(1) &= 1 \\
\alpha_1(i) &= a_{q_1 q_j} b_{q_j}(o_1) \\
\alpha_t(i) &= \left[\sum_{j=2}^{N-1} \alpha_{t-1}(j) a_{q_j q_i} \right] b_i(o_t) \\
\alpha_T(N) &= \sum_{j=2}^{N-1} \alpha_T(j) a_{q_i q_N}
\end{aligned} \tag{2.10}$$

Replacing "$\sum$" with "max" in Equation 2.7, Equation 2.9 and Equation 2.10, these equations describe the Viterbi algorithm actually used in speech recognition systems. By considering only the best state sequence at any time, the Viterbi algorithm has the advantage of in-place calculation, that is, there is no need to store all the $\alpha_t(i)$ over time t. Therefore, it is memory efficient.

The Viterbi algorithm also includes saving traces of the search so that the best state sequence can be retrieved after the search. More details of the Viterbi algorithm can be found in Rabiner's book [32].

2.3 Training: The Baum-Welch Algorithm

Solving the HMM training problem is difficult due to the fact that the state sequence is hidden to us. There is no "correct" state sequence corresponding to a given observation sequence for all but the case of degenerate models. Therefore, no sufficient statistics of the state sequence, as well as of the mixture weight, is available to obtain an analytical estimation [20]. Under this circumstance, the Maximum Likelihood (ML) estimation is usually obtained through the *Baum-Welch* algorithm, also called the *Expectation Maximization* (EM) algorithm [10].

The EM algorithm is for obtaining the ML estimation in the general case of incomplete data. Hence the incomplete data is the speech observation O, and the complete data $Y = (O, \kappa, q)$ is the union of the speech observation O, the hidden mixture component κ, and the state sequence q. The algorithm iteratively maximizes an auxiliary function

$Q(\hat{\lambda}, \lambda)$, defined as the expectation of the complete data log likelihood $P(Y \mid O, \hat{\lambda})$ given the incomplete data O and the current model λ:

$$\begin{aligned} Q(\hat{\lambda}, \lambda) &= E\left[logP(Y \mid O, \hat{\lambda}) \mid O, \lambda\right] \\ &= \sum_{all\, q} \sum_{all\, \kappa} \frac{P(O, q, \kappa \mid \lambda)}{P(O \mid \lambda)} \log P(O, q, \kappa \mid \hat{\lambda}) \qquad (2.11) \end{aligned}$$

It is shown by Liporace [25] that maximizing the likelihood function is equivalent to maximizing the auxiliary function:

$$Q(\hat{\lambda}, \lambda) \geq Q(\lambda, \lambda) \Rightarrow P(O \mid \hat{\lambda}) \geq P(O \mid \lambda)$$

Therefore, by iteratively maximizing the auxiliary function, the EM algorithm converges to the ML estimation of the acoustic model.

In the output probability density Equation 2.4, since the mixture weights are summed to one ($\sum_{k=1}^{K_j} \omega_{jk} = 1$) and the mixture weight number K_j is fixed for each state j, it has been shown that a HMM state with a mixture density is equivalent to a multi-state single-mixture density model that each mixture component has the statistical population equal to the mixture weight [18]. With this property, Equation 2.11 can be expanded as:

$$\begin{aligned} Q(\hat{\lambda}, \lambda) &= \sum_{all\, q} \sum_{all\, \kappa} \frac{P(O, q, \kappa \mid \lambda)}{P(O \mid \lambda)} \left[\sum_{t=1}^{T} \log a_{q_{t-1} q_t} + \sum_{t=1}^{T} \log \omega_{q_t k_t} + \sum_{t=1}^{T} \log N(\cdot)\right] \\ &= Q(\hat{A}, \lambda) + Q(\hat{\kappa}, \lambda) + Q(\hat{N}, \lambda) \qquad (2.12) \end{aligned}$$

where

$$\begin{aligned} Q(\hat{A}, \lambda) &= \sum_{all\, q} \sum_{all\, \kappa} \frac{P(O, q, \kappa \mid \lambda)}{P(O \mid \lambda)} \sum_{t=1}^{T} \log a_{q_{t-1} q_t} \\ Q(\hat{\kappa}, \lambda) &= \sum_{all\, q} \sum_{all\, \kappa} \frac{P(O, q, \kappa \mid \lambda)}{P(O \mid \lambda)} \sum_{t=1}^{T} \log \omega_{q_t k_t} \\ Q(\hat{N}, \lambda) &= \sum_{all\, q} \sum_{all\, \kappa} \frac{P(O, q, \kappa \mid \lambda)}{P(O \mid \lambda)} \sum_{t=1}^{T} \log N(\cdot) \end{aligned}$$

Because of the separability of $Q(\hat{\lambda}, \lambda)$ into three independent terms, $Q(\hat{\lambda}, \lambda)$ can be maximized by separately maximizing the individual terms: the transitions, the mixture weights, and the Gaussians.

The further deductions to maximize each individual item in Equation 2.12 are similar. As an example, the Gaussian auxiliary function $Q(\hat{N}, \lambda)$ can be expanded as:

$$\begin{aligned} Q(\hat{N}, \lambda) &= \sum_{all\, q} \sum_{all\, \kappa} \frac{P(O, q, \kappa \mid \lambda)}{P(O \mid \lambda)} \sum_{t=1}^{T} \log N(o_t; \hat{\mu}_{q_t k_t}, \hat{\Sigma}_{q_t k_t}) \\ &= \sum_{j=1}^{N} \sum_{k=1}^{K_j} \sum_{t=1}^{T} \frac{P(O = o_t, q_t = j, k_t = k \mid \lambda)}{P(O \mid \lambda)} \log N(o_t; \hat{\mu}_{jk}, \hat{\Sigma}_{jk}) \\ &= \sum_{j=1}^{N} \sum_{k=1}^{K_j} \sum_{t} \gamma_t(j,k) \log N(o_t; \hat{\mu}_{jk}, \hat{\Sigma}_{jk}) \end{aligned} \tag{2.13}$$

where $\gamma_t(j,k) = P(O = o_t, q_t = j, k_t = k \mid \lambda)/P(O \mid \lambda)$.

In Equation 2.13, $\gamma_t(j,k)$ is used as an *a-posteriori* probability. In the EM algorithm it is estimated, according to the current model λ, as the probability of observation o_t at state j, mixture component k, and then the value is used to estimate the new model $\hat{\lambda}$.

The calculation of $\gamma_t(j,k)$ is through the forward and backward probabilities [4]. The forward probability and its calculation are discussed in Section 2.2. The backward probability is defined as the probability of the partial observation sequence from $t+1$ to the end, given state i at time t and model λ:

$$\beta_t(i) = P(o_{t+1}, o_{t+2}, ...o_T \mid q_t = i, \lambda) \tag{2.14}$$

and the inductive calculation of $\beta_i(t)$ is as follows:

$$\begin{aligned} \beta_i(t) &= a_{q_i q_N} \\ \beta_i(t) &= \sum_{j=2}^{N-1} a_{q_i q_j} b_{q_j}(o_{t+1}) \beta_j(t+1) \\ \beta_1(1) &= \sum_{j=2}^{N-1} a_{q_1 q_j} b_{q_j}(o_1) \beta_j(1) \end{aligned} \tag{2.15}$$

The estimate $\gamma_t(j,k)$ is then calculated as [28]:

$$\gamma_t(j,k) = \left[\frac{\alpha_j(t)\beta_j(t)}{\sum_{r=1}^{T} \alpha_r(t)\beta_r(t)} \right] \left[\frac{\omega_{jk} N(o_t, \mu_{jk}, \Sigma_{jk})}{\sum_{r=1}^{K_j} \omega_{jr} N(o_t, \mu_{jr}, \Sigma_{jr})} \right] \tag{2.16}$$

Finally the EM estimation of the HMM parameters is as follows:

$$
\begin{aligned}
\hat{\omega}_{jk} &= \frac{\sum_{t=1}^{T} \gamma_t(j,k)}{\sum_{t=1}^{T} \sum_{k=1}^{K_j} \gamma_t(j,k)} \\
\hat{\mu}_{jk} &= \frac{\sum_{t=1}^{T} \gamma_t(j,k) \cdot o_t}{\sum_{t=1}^{T} \gamma_t(j,k)} \\
\hat{\Sigma}_{jk} &= \frac{\sum_{t=1}^{T} \gamma_t(j,k) \cdot (o_t - \mu_{jk})(o_t - \mu_{jk})^T}{\sum_{t=1}^{T} \gamma_t(j,k)}
\end{aligned}
$$

2.4 HMM Configurations

The previous two sections discussed the mathematical equations for the HMM training and recognition. In a practical system with a training corpus of certain size, two questions have to be answered before one can start to build an acoustic model: (1) How many acoustic parameters should there be in the model? (2) How can these parameters be robustly trained? The former question is related to the modeling ability of the model while the latter has influence on the robustness of the model.

In a typical context triphone system, the modeling unit is a triphone. There are often more triphones in the model than those observed in the training corpus. Certain tying of acoustic parameters is required to robustly train the model.

The tying of acoustic parameters can be in different levels: state level, mixture component level, or mean-vector/covariance-matrix level, which results in different HMM configurations. In this thesis, the state-tying scheme illustrated in Figure 2.3 is used. All the equations are derived based on the state-tying scheme. With simple modifications, the algorithm developed in this thesis can be easily applied to other HMM configurations.

In the state-tying scheme, HMM states are clustered together according to their acoustic similarities. All the states in such a state cluster share the same mixture parameters including the number of mixture components and parameters of each mixture component (mixture weights, mean vectors and covariance matrix).

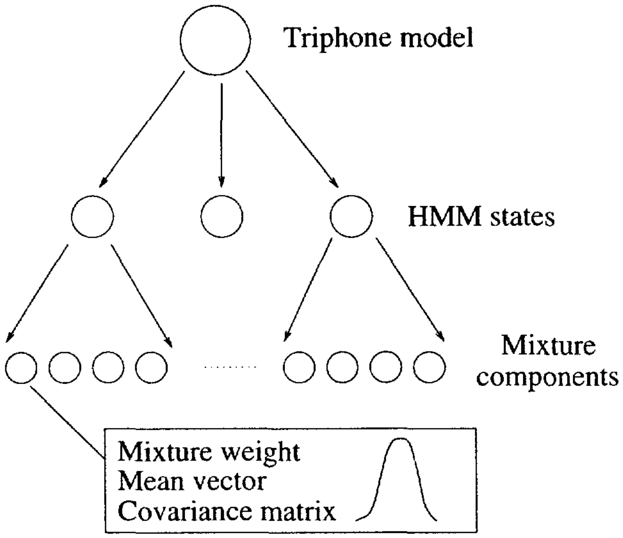

Figure 2.3: State-tying HMM configuration

Each triphone is modeled by a three-state HMM. The HMM states are tied to reduce the total number of parameters in the acoustic model. Each distinct HMM state consists of multiple mixture components. Parameters of each mixture component include the mixture weight, the mean vector, and the covariance matrix.

2.5 Notations

In the above state-tying configuration, each mixture component can be viewed as drawn from a pool of mixture components. Therefore, the mixture components can be uniquely indexed for notation simplicity and consistency.

In the rest of the thesis, the notations for the parameters of mixture component k of HMM state j will be simply addressed by a mixture component index m ($m = 1...M$), where M is the number of mixture components in the acoustic model. (The state index can be easily retrieved from the tying structure given the index m.) These parameters include: mixture weight ω_m, mean vector μ_m, covariance matrix Σ_m and mixture occupation $\gamma_m(t)$. For example, Equation 2.13 will be re-written as follows:

$$Q(\hat{N}, \lambda) = \sum_{m=1}^{M} \sum_{t=1}^{T} \gamma_t(m) \log N(o_t; \hat{\mu}_m, \hat{\Sigma}_m) \tag{2.17}$$

A detailed list of notations used in the thesis can be found in Appendix A.

Chapter 3

Overview of Adaptation Techniques

The purpose of Speaker Adaptation is to compensate for the mismatch between the acoustic model and the acoustic characteristics of the test speaker. By adapting the SI model to each particular test-speaker, Speaker Adaptation techniques help improve the recognition performance.

3.1 Adaptation Scenarios

Supervised or *unsupervised*

Depending on whether the source of the references (of the enrollment data) is known or unknown, applying speaker adaptation can be in the *supervised* or *unsupervised* mode.

Supervised mode refers to the situation where the references of the enrollment data are known. For example, in the "speaker training" session of some dictation systems, a speaker may be directed to say certain sentences for enrollment.

Unsupervised mode refers to the situation where the references of the enrollment data are unknown. The recognition results with the SI model are usually used as the references. The errors in these references can be smoothed by accumulating the re-estimation parameters over a certain amount of adaptation data. Positive adaptation effect can be expected in average if the SI system has a high enough recognition rate. The lexicon and language model also help achieve such a positive effect during recognition.

Batch mode, incremental mode or auto-adaptation

With respect to the source of the enrollment data, applying Speaker Adaptation can be in *Batch* mode, in *Auto-adaptation* mode, or in *Incremental* mode.

In Batch mode adaptation, additional data from the test-speaker are used as the enrollment data, that is, the enrollment data and the test data are two different sets of speech from the same test-speaker. The Batch mode adaptation is suitable for applications in which the enrollment data of the test-speaker can be pre-collected.

In Auto-adaptation mode, the same data are used for both testing and enrollment. The references of the enrollment data are obtained from a previous recognition pass using the SI model. Then the SI model is adapted to the test-speaker and the test data is re-recognized using the adapted model. This mode is suitable for speech recognition that does not real-time response.

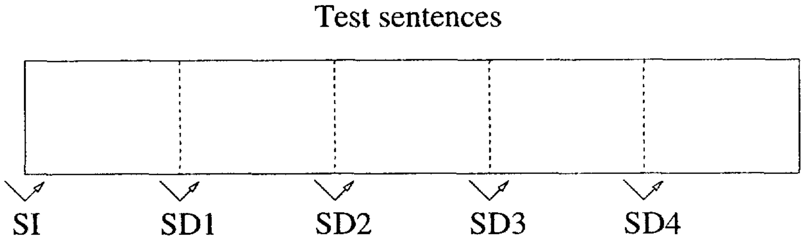

Figure 3.1: Incremental mode adaptation

In each incremental step, an acoustic model is adapted from the previous model using increasing amount of enrollment data. This adapted model will be used as a base model for the next incremental step.

In the Incremental mode (Figure 3.1), the adaptation is performed in intervals as the test-data accumulate. The first several utterances (of the test data) are recognized with the SI model and then used as the enrollment data for the first adaptation. The references come from the recognition results. The adapted model is then used to recognize the next several sentences for the successive adaptations. This procedure repeats until all the test sentences are recognized. The advantages of the Incremental mode adaptation are that (1) there is little time delay for adaptation and (2) more adaptation data can be used as more test data become available. The Incremental mode is typically used for online adaptation.

3.2 Various Adaptation Techniques

Adaptation techniques can be categorized roughly into two classes: those that normalize individual speaker variations and those that adapt the SI model to individual speaker variations. The former is also called *Speaker Normalization.*

3.2.1 Speaker Normalization

Vocal Tract Length Normalization

Vocal Tract Length Normalization (VTLN) [12, 22, 39] normalizes speaker variations in the signal space. It compensates the *Vocal Tract Length* (VTL) among individual speakers. The VTL contributes to the variability of speech waveforms. The first order effect of a difference in VTL is a scaling of the frequency axis [12].

The procedures of VTLN include first obtaining an optimal frequency warp scale, by either scanning a possible warping range or by estimating the warp scale from the formant positions, and then computing the feature vectors from the accordingly warped frequency axis. The resulting feature space is speaker normalized.

Speaker Adaptive Training

Speaker Adaptive Training (SAT) [2, 1, 3] is motivated by the observation that variability in SI models can be attributed to both within-speaker phonetic variation and variation among the speakers of the training population. These two variations can be decoupled and the inter-speaker variation can be eliminated. The resulting compact model is more suitable for adaptation than the SI model.

The removal of inter-speaker variation is done by normal Speaker Adaptation techniques. The exact procedures consist of estimating a transformation matrix for each training speaker to account for the individual speaker variation and jointly estimating a new model so that the likelihood of the training data is maximized.

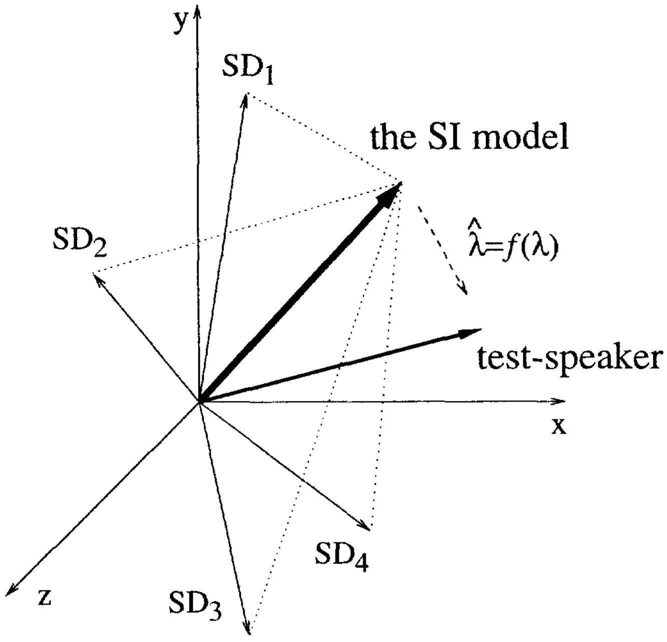

Figure 3.2: Adaptation using a mapping function

If a mapping function $f(\cdot)$ can be estimated, the SI model can be mapped to the test-speaker dependent model in the acoustic hyper-space.

3.2.2 Transform Based Adaptation

Consider the acoustic hyper-space where the SI model λ and the test-speaker dependent model $\hat{\lambda}$ (unknown in practice) are two vectors, as illustrated in Figure 3.2. The SI model can be viewed as a combination of all the SD means in the training corpus. If a mapping function $\hat{\lambda} = f(\lambda)$ can be estimated from the adaptation data, then the SI model can be adapted to the test speaker.

The mapping function $f(\cdot)$ can be either linear or nonlinear. However, only linear mapping has been widely studied due to its simplicity in mathematics and effectiveness in practice.

Maximum Likelihood Linear Regression

Maximum Likelihood Linear Regression (MLLR) [24] uses linear transformation matrices to adapt the mean vectors of the SI model. Usually only the mean vectors are adapted because they are considered to contribute most to speaker variations. There have been also studies on variance compensation [13], which report additional performance improvements.

Given a mean vector μ, a transformation matrix A (of size $D \times D$) and an offset vector β, the adapted mean vector $\hat{\mu}$ is as follows:

$$\hat{\mu} = A\mu + \beta \tag{3.1}$$

Denoting the transformation matrix $W = [A\ \beta]$ and the augmented mean vector $\tilde{\mu} = \left[\mu^T\ 1\right]^T$,Equation 3.1 can be simplified as follows:

$$\hat{\mu} = W \cdot \tilde{\mu} \tag{3.2}$$

The transformation matrix W is estimated from the adaptation data under the criterion that the likelihood of the adaptation data is maximized. According to Section 2.3, it is equivalent to maximizing the auxiliary function $Q(\hat{\lambda}, \lambda)$.

Referring to Equation 2.12, the auxiliary function can be divided into three parts. Ignoring the mixture weights $Q(\hat{\kappa}, \lambda)$ and the transitions $Q(\hat{A}, \lambda)$ since they are not changed during adaptation, the auxiliary function can be written as:

$$Q(\hat{\lambda}, \lambda) = \sum_{m=1}^{M} \sum_{t=1}^{T} \gamma_t(m) \log N(o_t; W \cdot \tilde{\mu}_m, \Sigma_m) \tag{3.3}$$

Assume $\gamma_t(m)$ is not affected by the adaptation. The estimation of the transformation matrix W can be obtained by solving:

$$\frac{\partial Q(\hat{\lambda}, \lambda)}{\partial W} = 0 \tag{3.4}$$

Equation 3.4 becomes:

$$\sum_{m=1}^{M} \sum_{t=1}^{T} \gamma_t(m) \Sigma_m^{-1} o_t \tilde{\mu}_m^T = \sum_{m=1}^{M} \sum_{t=1}^{T} \gamma_t(m) \Sigma_m^{-1} W \tilde{\mu}_m \tilde{\mu}_m^T \tag{3.5}$$

If the covariance matrices of the acoustic model are diagonal[1], Equation 3.5 has a computationally feasible closed-form solution:

$$W_i^T = [G(i)]^{-1} Z_i^T \tag{3.6}$$

[1] Most speech recognition systems use diagonal covariance matrices to save memory and reduce computation during recognition

where W_i denotes the ith row vector of matrix W, Z_i denotes the ith row vector of matrix Z, and

$$\begin{aligned} G_{hl}(i) &= \sum_{m=1}^{M} V_{ii}(m) D_{hl}(m) && (3.7)\\ Z &= \sum_{m=1}^{M}\sum_{t=1}^{T} \gamma_t(m)\Sigma_m^{-1} o_t \tilde{\mu}_m^T && (3.8)\\ V(m) &= \sum_{t=1}^{T} \gamma_t(m)\Sigma_m^{-1} && (3.9)\\ D(m) &= \tilde{\mu}_m \tilde{\mu}_m^T && (3.10) \end{aligned}$$

More than one matrix can be used in MLLR to transform the mean vectors. Each transformation matrix corresponds to a certain phonetic class (defined as a cluster of mixture components) called a *regression class*. The usage of *regression classes* refines the MLLR adaptation but also requires more data to be effective. Details of how to generate *regression classes* can be found in Gales's technical report [14]. In Section 4.7, a decision-tree based regression-class generation method will also be introduced.

3.2.3 Dependency Modeling

Dependency modeling has been studied as a way of incorporating prior knowledge of acoustic models into adaptation. The goal is to improve the adaptation performance given a limited amount of adaptation data. In this case, popular adaptation algorithms, such as MLLR, generate ill-structured transformation matrices due to a lack of constraints in the matrix estimation.

The motivation for modeling dependency comes from the observation that acoustic parameters in the same phonetic class or across different phonetic classes are highly correlated. These parametric dependencies reflect the internal structure of acoustic models and can be used as parameter predictions (predicting unseen model parameters) and knowledge constraints (avoiding erroneous parameters) in adaptation.

The next three algorithms demonstrate three different ways of modeling the dependencies. The emphasis is on how the correlations are modeled.

Predictive Speaker Adaptation

The *Predictive Speaker Adaptation* method (PSA) [8] assumes that linear correlations occur in different acoustic units (word, phone, triphone, etc). During training, correlations among all the pairs of the mean vectors are estimated so that for any mean vector index p and q, $\hat{\mu}_p = a_{pq} + b_{pq}\mu_q$, where a_{pq} and b_{pq} are the coefficients.

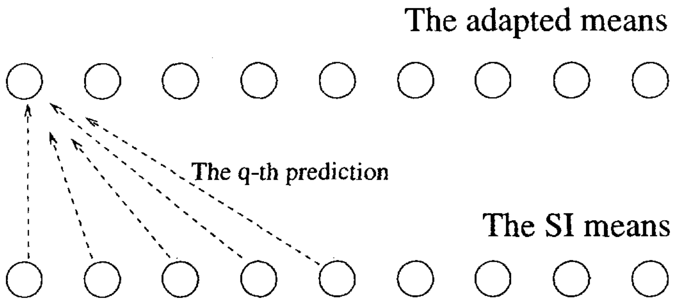

Figure 3.3: Predictive Speaker Adaptation

The adapted means are linearly correlated with the SI model means.

As illustrated in Figure 3.3, let $y_q = a_{pq} + b_{pq}\mu_q$ denote the qth prediction for $\hat{\mu}_p$ and let Y denote the set of all the y_q. The distribution of y_q is simplified to be normal with mean $\hat{\mu}_p$, and covariance σ_q^2. Therefore,

$$P(\hat{\mu}_p \mid Y) = \prod_q \frac{1}{\sqrt{2\pi}\sigma_q} \exp\left[-\frac{1}{2}\frac{(y_q - \hat{\mu}_p)^2}{\sigma_q^2}\right] \tag{3.11}$$

The ML estimation of $\hat{\mu}_p$ can be obtained by maximizing Equation 3.11.

Extended Maximum A Posteriori

The *Extended Maximum A Posteriori* (EMAP) adaptation [45] models the correlations both across speakers and within acoustic models.

Assuming feature dimensions are not correlated, EMAP adapts each feature dimension individually. Let R denote the number of speakers in the training set, M the number of mean vectors in the acoustic model, and let d denote the current feature dimension of concern. Let $\mu_m^{(r)}$ denote the dth element of mean vector m ($m = 1...M$) of speaker r ($r = 1...R$). The correlations are modeled among columns and rows of the following mean-vector matrix:

$$\begin{array}{cccc} \mu_1^{(1)} & \mu_2^{(1)} & \cdots & \mu_M^{(1)} \\ \mu_1^{(2)} & \mu_2^{(2)} & \cdots & \mu_M^{(2)} \\ \cdots & \cdots & \cdots & \cdots \\ \mu_1^{(R)} & \mu_2^{(R)} & \cdots & \mu_M^{(R)} \end{array}$$

For the correlations across all the speakers, the distribution of the mean values in column m is assumed normal with mean μ_m and covariance matrix Σ_m. For the correlations within acoustic models, the distribution of the mean values in row r is also assumed normal with mean μ_0 and covariance matrix Σ_0. These assumptions are used to represent the distribution of the acoustic model $P(\lambda)$.

EMAP incorporates correlations into adaptation under the *Maximum A Posteriori* (MAP) framework [21, 15, 45]. The MAP criterion is as follows:

$$\begin{aligned} \hat{\lambda} &= \arg\max_{\lambda} P(\lambda \mid O) \\ &= \arg\max_{\lambda} P(O \mid \lambda) P(\lambda) \end{aligned} \tag{3.12}$$

Recall that the ML criterion is:

$$\hat{\lambda} = \arg\max_{\lambda} P(O \mid \lambda) \tag{3.13}$$

Comparing Equation 3.12 with Equation 3.13, the introduction of the prior distribution of acoustic model $P(\lambda)$ provides us a way to incorporate prior information of the acoustic model into the adaptation. If the prior $P(\lambda)$ is assumed constant but unknown ($P(\lambda) \propto$ *constant*), the MAP estimation reduces to the ML estimation. In practice, the prior $P(\lambda)$ is approximated with estimates from the training corpus. In EMAP, it is represented by the previously discussed assumptions.

Markov Random Field

Like EMAP, the *Markov Random Field* (MRF) adaptation [34, 35, 36] also models the correlations to represent the prior distribution of the acoustic model $P(\lambda)$. The difference is that the representation is based on the MRF theory rather than the simplified normal-distribution assumptions.

MRF arranges all the mean vectors into a Markov random field Q in such a way that the column number represents the index of the mean vectors and the row number represents the feature dimensions. For example, each point (p, q) in the field represents the qth element of mean vector p in an acoustic model.

Under the MRF theory, $P(\lambda)$ has the following form:

$$P(Q) = \frac{1}{z} \exp\left[-\frac{\beta}{2} \sum_{c} V_c(Q)\right] \tag{3.14}$$

In Equation 3.14, z and β are constants. The summation is over all the *cliques*, defined as pairs of points (in field Q) whose values show high correlations during training. $V_c(Q)$ measures contributions from the cliques and has a simple form similar to the linear correlation:

$$V_c(Q) = \frac{1}{\lambda_{min}(1 + b_c^2)}(y_c - a_c - b_c x_c)^2 \tag{3.15}$$

where x_c and y_c are the two points in clique c and $(\lambda_{min}, a_c, b_c)$ are coefficients of clique c to be estimated.

The MRF adaptation incorporates correlations into adaptation also under the MAP framework. By solving Equation 3.12 with the prior from Equation 3.14, the MRF adaptation has an iterative solution.

The Markov random field has a theoretically equivalent (Luettgen [26]) alternative called the multi-scale representation, from which efficient recursive solutions to both the parameter estimation and adaptation can be derived [7, 19].

More discussions of the MRF theory will be presented in Chapter 4 since the algorithm developed by this thesis also uses the MRF theory to model the correlations.

3.3 This Thesis in Perspective

The dependency modeling methods have explored possible ways of incorporating the correlations into adaptation. During adaptation, they share the common trait that each individual parameter is updated and then the change of a parameter is smoothed 'to' or 'by' its neighbors. This adaptation strategy results in a slow convergence to the test-speaker: a large amount of adaptation data is required to make a sufficient change to the acoustic model.

The transformation-based adaptation algorithms have the advantage of making broad adjustments of the parameters. However, when the adaptation data are sparse or not representative, the estimated transformation matrices are ill-structured.

This thesis proposes the MRFLR (Markov Random Field Linear Regression) algorithm, which constrains the transformation estimation with the correlations among acoustic parameters. The purpose is to prevent the estimation from generating ill-structured transformation matrices, thereby improving the adaptation performance when only a limited amount of adaptation data is available. It can also be viewed as a combination of the linear transformation-based adaptation and the dependency modeling adaptation.

Chapter 4

Markov Random Field Linear Regression

In this chapter, the MRFLR (Markov Random Field Linear Regression) algorithm will be developed. It is an extension of MLLR under the MAP criterion. The hypothesis of MRFLR is that the correlations among acoustic parameters represent the internal structure of speech sounds, that is, prior knowledge of acoustic models. These correlations can be modeled and explicitly used as a constraint on the possible outputs of the transformation estimations. The resulting transformation matrices will be more robust compared with the traditional MLLR approach. MRFLR is particularly useful for speech applications in which the amount of adaptation data is limited.

Early attempts of extending MLLR to incorporate prior knowledge have faced various difficulties. For example, ignoring the Gaussian transitions and the mixture weights[1], the prior distribution of the acoustic model is normal-Wishart as explained below:

$$P(\lambda) \propto P(\mu, \Sigma) = P(\mu \mid \Sigma)P(\Sigma) \tag{4.1}$$

The first item $P(\mu \mid \Sigma)$ in Equation 4.1 is normal:

$$P(\mu \mid \Sigma) \propto \frac{1}{\sqrt{(2\pi)^p \mid \Sigma_0 \mid}} \exp\left\{-\frac{1}{2}(\mu - \mu_0)^T \Sigma_0^{-1}(\mu - \mu_0)\right\} \tag{4.2}$$

and the second item $P(\Sigma)$ is Wishart:

$$P(\Sigma) \propto \frac{\mid \Sigma \mid^{(n-p-1)/2}}{\mid \Phi \mid^{n/2}} \exp\left\{-\frac{1}{2} tr\ \Phi^{-1}\Sigma\right\}$$

where Φ is a scale matrix of size $p \times p$, n is a degrees-of-freedom parameter, p is the

[1] Leggetter [23] has a detailed analysis of adapting various HMM parameters using the transformation-based adaptation. The conclusion is that the effect of the Gaussian transitions and mixture weights is small in a continuous density system. It will not adversely affect the adaptation performance by not adapting these parameters.

dimension $p \equiv M \cdot D$ (M is the Gaussian mixture-component number and D is the feature dimension), and μ_0 (size $p \times 1$) and $\mathbf{\Sigma}_0$ (size $p \times p$) are parameters of the normal distribution.

The representation of the prior knowledge with the normal-Wishart distribution has no closed-form solution for the matrix estimation. Moreover, modeling both the mean vectors and the covariance matrices involves too many computations to be feasible in a realistic adaptation. A reasonable scaled-down task would be to assume the distribution of the covariance matrices $P(\mathbf{\Sigma})$ to be an unknown constant and model only the mean vectors, since the mean vectors are regarded as contributing the most to speaker variations.

The representation of the prior knowledge with the normal distribution (as in Equation 4.2) has the drawback that it is difficult to use empirical Bayesian methods [40] to estimate the prior parameters due to the excessive size of the parameter space.

MRFLR, as the name implies, uses an implementation based on the Markov Random Field (MRF) theory [11] to model the correlations among the mean vector field. The advantage of using MRF is that only large correlations exhibited on prior speakers will be included to represent the prior. The closed-form solution for the matrix estimation can be obtained under a reasonable simplification.

The following sections will first focus on the modeling of the correlations and then the formula deduction of MRFLR. Discussion and implementation issues will be presented thereafter.

4.1 Markov Random Field

Consider a two-dimensional random variable field Q that consists of all the mean vectors of an acoustic model λ, as illustrated in Figure 4.1. The size of the field is $D \times M$, where D is the feature dimension and M is the number of Gaussian mixture components in the acoustic model. Each column number represents the index of a Gaussian mixture component and each row number represents the index of a feature dimension. For example, each point p, q refers to the qth element of mean vector p and each column vector represents the corresponding mean vector in the acoustic model.

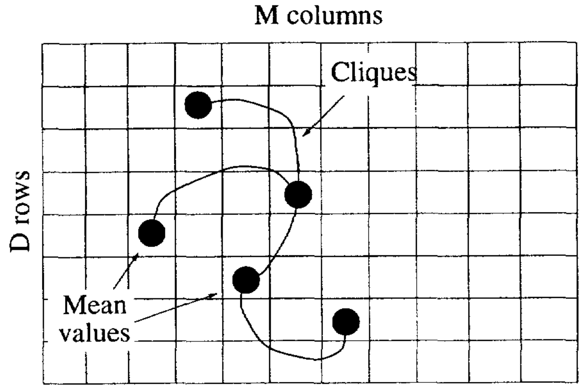

Figure 4.1: MRF representation of mean values

The random field consists of all the mean vectors of an acoustic model. Each column number represents the index of a Gaussian mixture component and each row number represents the index of a feature dimension. Cliques are defined as pairs of points that show high correlations during training.

The idea of arranging the Gaussian means into a field was brought up by Shahshahani [34] in 1995. There is a slight difference between his approach and the one described above. In his approach, the mean vectors in the field should be normalized by removing the inter-speaker scalars (See SAT, Section 3.2.1). Therefore only the phonetic variations are modeled by the field. This thesis does not normalize the mean vectors and thus models both the phonetic variations and the inter-speaker variations.

In such a random variable field, correlations exist as some variables (mean values) depend on other variables. *Cliques* are defined and detected as pairs of points[2] that show high correlations during training. (Details of how to detect cliques will be discussed in Section 4.6.) The dependencies can be expressed as follows:

$$P(x_s \mid \{x_r, r \neq s\}) = P(x_s \mid \{x_r, r \in \Omega_s\}) \tag{4.3}$$

where x_s and x_r are points in the field, Ω_s is the set of clique points of x_s, and $\{x_r, r \neq s\}$ is an abbreviation of all the points in the field except point x_s.

The left part of Equation 4.3 is called the *local characteristics* of point x_s. According

[2] According to Shahshahani's research work [34], using three(or more)-point cliques do not necessarily improve the ability of modeling the correlations. Therefore, two-point cliques are used in this thesis for simplicity. Adding a bit of complexity, estimation equations of MRFLR using arbitrary-point cliques can be obtained.

to Besag [5], the joint distribution of the field $P(Q)$ is uniquely determined by all these conditional probabilities (local characteristics).

However, specifying the local characteristics is extremely difficult (Chellappa and Kashyap [6]). This seeming limitation of the MRF theory has been addressed by a theorem [38, 5] that reveals the equivalence of the Markov random fields and Gibbs fields (Hassner and Sklansky [16]). Thus, for the equivalent Gibbs field Q, the joint distribution of field Q has the following form:

$$P(Q) = \frac{1}{z} \exp\left\{-\frac{\beta}{2}\sum_{c} V_c(Q)\right\} \tag{4.4}$$

where z is a normalizing constant (called the *partition function*), β is a scale (or the *temperature* $T = \frac{2}{\beta}$), and $V_c(Q)$ is called the *potential function* whose form will be discussed next. The summation is over all the cliques.

These terminologies come from statistical physics, wherein such measures are "equilibrium states" for physical systems. The potential function $V_c(Q)$ represents contributions to the total energy from external fields. In MRFLR, since only two-point cliques are involved, the potential function is proposed based on the straight-line fitting.

Let x_c and y_c denote the two points in clique c. The χ^2 merit function measures how well $a_c + b_c x_c$ fits y_c given samples of x_c and y_c:

$$\chi^2(a_c, b_c) = \sum_{r=1}^{R} \frac{(y_c^{(r)} - a_c - b_c x_c^{(r)})^2}{\sigma_{y_c}^2 + b_c^2 \sigma_{x_c}^2} \tag{4.5}$$

where a_c and b_c are coefficients, $x_c^{(r)}$ is the rth sample of point x_c and σ_{x_c} is the standard deviation of x_c. The same notation convention applies to y_c.

The potential function $V_c(Q)$ is thus defined as follows:

$$V_c(Q) = \omega_c (y_c - a_c - b_c x_c)^2 \tag{4.6}$$

where $\omega_c = 1/(\sigma_{y_c}^2 + b_c^2 \sigma_{x_c}^2)$.

In the latter formula deductions, the index c of the clique parameters will be dropped for notation simplicity.

4.2 MRFLR Adaptation

As MLLR estimates the transformation matrix under the ML criterion, MRFLR estimates the transformation matrix under the MAP criterion. Referring to Equation 3.3, the MAP criterion has the following auxiliary function taking into account the contributions from $P(\hat{\lambda})$:

$$Q(\hat{\lambda}, \lambda) = \sum_{m=1}^{M} \sum_{t=1}^{T} \gamma_t(m) \log N(o_t; \hat{\mu}_m, \Sigma_m) + \log P(\hat{\lambda}) \tag{4.7}$$

In Equation 4.7, $\hat{\lambda}$ is the updated acoustic model, whose mean vectors have been adapted by a transformation matrix W as in MLLR:

$$\hat{\mu} = W \cdot \tilde{\mu} \tag{4.8}$$

where W is a $D \times (D+1)$ matrix and $\tilde{\mu} = \left[\mu^T \, 1\right]^T$ is the augmented mean vector.

Since only the mean vectors are modeled and the covariance matrices are assumed unchanged during adaptation, $P(\hat{\lambda})$ has the following form (Equation 4.4 and Equation 4.6):

$$P(\hat{\lambda}) \propto \frac{1}{z} \exp\left\{ -\frac{\beta}{2} \sum_{c} \omega(\hat{y} - a - b\hat{x})^2 \right\} \tag{4.9}$$

where $\hat{x}$ and $\hat{y}$ are the two points of clique c in the adapted Markov random field, that is, each column vector (the mean vector) has been transformed according to Equation 4.8.

In Equation 4.9, the two points of a clique can be from different feature dimensions, which makes further equation deductions difficult. Equation 4.9 needs some simplifications before one can proceed. The following facts have been considered:

- In HMM systems, diagonal covariance matrices are often the choice to reduce the computational complexity and storage requirement. The underlying assumption of using diagonal covariance matrices is that feature correlations across different dimensions are less important than those within the same dimensions. The performance loss of using diagonal covariance matrices can often be compensated by using more Gaussian mixture components.

- In Leggetter's technical report of MLLR [23], the adaptation effects of using diagonal transformation matrices have been studied. The experimental results show that although diagonal transformation matrices are less effective than full transformation matrices, similar performances can be achieved by using more diagonal transformation matrices so that there are equivalent number of matrix parameters in both diagonal and full transformation matrices.

- Table 4.1 shows some experimental observations for the MFCC feature (12 mel frequency cepstral coefficients, energy, and their first and second derivatives, total of 39 parameters). Linear correlations were detected on acoustic models trained for the WSJ 20K task using the standard SI284 training set.

  In Table 4.1, the number of cliques for each feature dimension is shown from the same dimension as well as from different dimensions. Although some dimensions have a relative higher percentage (all of them <6%) of cross-dependencies than others, the overall number of cross-dependencies is only 1.17% of the total. Therefore, it is numerically safe to ignore these cross-dimensional cliques.

Based on the above observations, The assumption is made that contributions from those cross-dimensional dependencies are small in numbers compared with those equal-dimensional dependencies. Therefore, the cross-dimensional cliques are ignored in Equation 4.9.

Equation 4.9 can be re-written as:

$$\begin{aligned} P(\hat{\lambda}) &\propto \frac{1}{z}\exp\left\{-\frac{\beta}{2}\sum_{c}\omega(\hat{y}-a-b\hat{x})^2\right\} \\ &= \frac{1}{z}\exp\left\{-\frac{\beta}{2}\sum_{c}\omega\left[E^{(d)}(\hat{\mu}_y-a-b\hat{\mu}_x)\right]^T\left[E^{(d)}(\hat{\mu}_y-a-b\hat{\mu}_x)\right]\right\} \\ &= \frac{1}{z}\exp\left\{-\frac{\beta}{2}\sum_{c}\omega(\hat{\mu}_y-a-b\hat{\mu}_x)^T E^{(d)}(\hat{\mu}_y-a-b\hat{\mu}_x)\right\} \end{aligned} \tag{4.10}$$

where $\hat{\mu}_x$ and $\hat{\mu}_y$ are the corresponding column mean vectors of the field, and $E^{(d)}$ is a $D \times D$ matrix, $E^{(d)}_{ij} = \begin{cases} 1 & i=j=d \\ 0 & otherwise \end{cases}$, with d being the feature dimension of the two points in clique c.

Table 4.1: MFCC correlations distributions

Feature dimension	Cross-dimensional correlations	Same-dimensional correlations	Total correlations	Cross-dimensional correlations percent
0	97	408548	408645	0.02%
1	16	369917	369933	0.00%
2	1058	381721	382779	0.28%
3	18753	399843	418596	4.48%
4	2922	386921	389843	0.75%
5	4971	389692	394663	1.26%
6	4021	375066	379087	1.06%
7	1786	365291	367077	0.49%
8	20596	363223	383819	5.37%
9	5445	361314	366759	1.48%
10	16427	368585	385012	4.27%
11	10748	345910	356658	3.01%
12	115	445634	445749	0.03%
13	536	229914	230450	0.23%
14	95	240793	240888	0.04%
15	1545	253629	255174	0.61%
16	7856	264486	272342	2.88%
17	463	245238	245701	0.19%
18	1061	246057	247118	0.43%
19	1942	243470	245412	0.79%
20	1602	232537	234139	0.68%
21	2812	227878	230690	1.22%
22	4666	226405	231071	2.02%
23	8433	225923	234356	3.60%
24	2114	213024	215138	0.98%
25	173	318099	318272	0.05%
26	1179	214129	215308	0.55%
27	85	213606	213691	0.04%
28	205	229985	230190	0.09%
29	1981	230188	232169	0.85%
30	224	221591	221815	0.10%
31	246	221522	221768	0.11%
32	468	211002	211470	0.22%
33	390	204881	205271	0.19%
34	635	200562	201197	0.32%
35	1132	202550	203682	0.56%
36	2534	201597	204131	1.24%
37	435	184163	184598	0.24%
38	515	309186	309701	0.17%
Total	130282	10974080	11104362	1.17%

Plugging Equation 4.8 and Equation 4.10 into Equation 4.7, the MRFLR auxiliary function is re-written to be:

$$\begin{aligned} Q(\hat{\lambda}, \lambda) &= \sum_{m=1}^{M} \sum_{t=1}^{T} \gamma_t(m) \log N(o_t; W \cdot \tilde{\mu}_m, \Sigma_m) - constant \\ & -\frac{\beta}{2} \sum_{c} \omega (W \cdot \tilde{\mu}_y - a - bW \cdot \tilde{\mu}_x)^T E^{(d)} (W \cdot \tilde{\mu}_y - a - bW \cdot \tilde{\mu}_x) \qquad (4.11) \end{aligned}$$

Up to present, all the pieces have been put together to solve the matrix estimation problem. The transformation matrix can be estimated by maximizing the MRFLR auxiliary function. Setting the differential of Equation 4.11 *w.r.t.* W to zero, it goes:

$$\begin{aligned} & \sum_{m=1}^{M} \sum_{t=1}^{T} \gamma_t(m) \Sigma_m^{-1} o_t \tilde{\mu}_m^T - \beta \sum_{c} a \omega E_d^{(d)} (\tilde{\mu}_y - b\tilde{\mu}_x)^T = \\ & \quad \sum_{m=1}^{M} \sum_{t=1}^{T} \gamma_t(m) \Sigma_m^{-1} W \tilde{\mu}_m \tilde{\mu}_m^T - \beta \sum_{c} \omega E^{(d)} W (\tilde{\mu}_y - b\tilde{\mu}_x)(\tilde{\mu}_y - b\tilde{\mu}_x)^T \qquad (4.12) \end{aligned}$$

where $E_d^{(d)}$ is the dth column vector of $E^{(d)}$.

Define Z, $V^{(1)}(m)$, $V^{(2)}(c)$, $D^{(1)}(m)$, and $D^{(2)}(c)$ as follows:

$$\begin{aligned} Z &= \sum_{m=1}^{M} \sum_{t=1}^{T} \gamma_t(m) \Sigma_m^{-1} o_t \tilde{\mu}_m^T - \beta \sum_{c} a \omega E_d^{(d)} (\tilde{\mu}_y - b\tilde{\mu}_x)^T \qquad (4.13) \\ V^{(1)}(m) &= \sum_{t=1}^{T} \gamma_t(m) \Sigma_m^{-1} \qquad (4.14) \\ V^{(2)}(c) &= \omega E^{(d)} \\ D^{(1)}(m) &= \tilde{\mu}_m \tilde{\mu}_m^T \\ D^{(2)}(c) &= (\tilde{\mu}_y - b\tilde{\mu}_x)(\tilde{\mu}_y - b\tilde{\mu}_x)^T \end{aligned}$$

Equation 4.12 is re-written as follows:

$$Z = \sum_{m=1}^{M} V^{(1)}(m) W D^{(1)}(m) - \beta \sum_{c} V^{(2)}(c) W D^{(2)}(c) \qquad (4.15)$$

If the covariance matrix Σ_m is diagonal ($E^{(d)}$ is always diagonal), MRFLR has a computational-feasible close-form solution. Equation 4.15 becomes:

$$\begin{aligned} Z_{il} &= \sum_{m=1}^{M} \sum_{h=1}^{D+1} V_{ii}^{(1)}(m) W_{ih} D_{hj}^{(1)}(m) - \beta \sum_{c} \sum_{h=1}^{D+1} V_{ii}^{(2)}(c) W_{ih} D_{hj}^{(2)}(c) \\ &= \sum_{h=1}^{D+1} G_{hl}(i) W_{ih} \qquad (4.16) \end{aligned}$$

where

$$G_{hl}(i) = \sum_{m=1}^{M} V_{ii}^{(1)}(m) D_{hl}^{(1)}(m) - \beta \sum_{c} V_{ii}^{(2)}(c) D_{hl}^{(2)}(c) \qquad (4.17)$$

Fixing i, Equation 4.16 can be viewed as a set of linear algebraic equations for solving W_{ih} (indexed by l). Therefore, the MRFLR solution is as follows:

$$W_i^T = [G(i)]^{-1} Z_i^T \qquad (4.18)$$

where W_i is the ith row vector of W and Z_i is the ith row vector of Z.

4.3 Understanding The MRFLR Formula

For an intuitive understanding of the MRFLR estimation, Equation 4.18 has two accumulators, the numerator matrix G (Equation 4.17) and the denominator matrix Z (Equation 4.13). Both of the two matrices consist of a left part coming from the adaptation data (the same as that in MLLR, Equation 3.7), and a right part coming from the cliques. The left parts reflect the influence of the adaptation data while the right parts reflect the influence of the prior knowledge.

For a clearer explanation, the accumulators in the MRFLR solution Equation 4.18 can be simplified to scalars:

$$\begin{aligned} w &= \frac{z_l + z_r}{g_l + g_r} \qquad (4.19) \\ &= \eta \cdot \frac{z_l}{g_l} + (1 - \eta) \cdot \frac{z_r}{g_r} \end{aligned}$$

where $\eta = g_l / (g_l + g_r)$.

In Equation 4.19, the value z_l / g_l represents the estimate of value w from the adaptation data, and the value z_r / g_r denotes the estimate of value w from the training corpus. The ratio η is determined by the amount of adaptation data. The MRFLR solution of value w is one of the following:

1. If the ratio η is very small ($\eta \to 0$), then $w \approx z_r/g_r$. This case indicates that when adaptation data are sparse, the knowledge constraints dominate the estimation and preserve the structural information of the transformation matrix obtained from the training data.

2. If the ratio η is very big ($\eta \to 1$), then $w \approx z_l/g_l$. This case indicates that when more adaptation data are available, the adaptation data dominate the estimation. The MRFLR performance converges to the MLLR performance.

3. Otherwise, w is somewhere between z_l/g_l and z_r/g_r. The effect of knowledge constraints on the estimation is determined by the amount of available adaptation data.

4.4 Incremental Adaptation Formula

The MRFLR solution presented in Equation 4.18 assumes that adaptation data have been observed before the estimation takes place. In the incremental adaptation mode, only up-to-present adaptation data are in sight. The incremental adaptation formula can be obtained by re-arranging the estimation equations to separate those components dependent on time.

Equation 4.13 is re-arranged as:

$$Z = \sum_{m=1}^{M} \left[\sum_{t=1}^{T} \gamma_t(m) \right] \Sigma_m^{-1} o_t \tilde{\mu}^T - \beta \sum_{c} a \omega E_d^{(d)} (\tilde{\mu}_y - b\tilde{\mu}_x)^T \tag{4.20}$$

Equation 4.14 is re-arranged as:

$$V^{(1)}(m) = \left[\sum_{t=1}^{T} \gamma_t(m) \right] \Sigma_m^{-1} \tag{4.21}$$

With Equation 4.20 and Equation 4.21, the MRFLR estimation can be carried out with up-to-present adaptation data.

4.5 Multiple Transform Matrices

As mentioned in Section 3.2.2, multiple transformation matrices can be estimated in MLLR to improve the adaptation performance. Each transformation matrix corresponds to a regression class: a set of Gaussian mixture components with similar acoustic properties. This scheme can also be used in MRFLR.

To extend MRFLR to multiple transformation matrices, one has to overcome the difficulty that the correlations among acoustic parameters might be across different regression classes. Hence, a joint estimation of all the transformation matrices is required to achieve the optimal solution, which is computationally expensive if not impossible.

However, if it can be assumed that most of the correlations occur within the same regression classes, then the estimation can be carried out separately for each transformation matrix, ignoring the correlations that cross different regression classes. This assumption is reasonable considering the acoustic parameters are more similar within the same regression classes than across different classes.

Table 4.2 shows the correlations of acoustic models trained for the WSJ 20K task[3]. The regression classes are indexed by the numbers in the first column. Correlations within the same regression classes are shown in the second column and correlations across different regression classes are shown in the third column. The correlation pairs across different regression classes occupy only a small percentage (less than 3%) compared with the correlations within the same regression classes.

The re-estimation equation remains the same as in Equation 4.18 except that the summation of the Gaussian mixture components and the summation of the cliques are from each corresponding regression class.

[3] The number of cliques in Table 4.2 is bigger than that in Table 4.1 because we intentionally include more cliques in the table.

Table 4.2: Correlations statistics for regression classes

r-class	across	within	%	r-class	across	within	%
0	0	31877809	0.00%	32	12849	846553	1.50%
1	2368	14158356	0.02%	33	1573	1092585	0.14%
2	10029	8359881	0.12%	34	9075	826916	1.09%
3	7882	4559837	0.17%	35	431	4872199	0.01%
4	4279	1517758	0.28%	36	281	1714207	0.02%
5	5896	3039786	0.19%	37	265	3157877	0.01%
6	6811	760340	0.89%	38	119	873082	0.01%
7	6317	2272214	0.28%	39	225	2284716	0.01%
8	5436	3796755	0.14%	40	1112	10053792	0.01%
9	2551	1324098	0.19%	41	3388	6644794	0.05%
10	2987	2472555	0.12%	42	436	1805181	0.02%
11	3000	510951	0.58%	43	145	910102	0.02%
12	8010	1953581	0.41%	44	136	683166	0.02%
13	2868	5787946	0.05%	45	43	226902	0.02%
14	1584	1452948	0.11%	46	561	894809	0.06%
15	2337	483330	0.48%	47	3015	4839550	0.06%
16	1570	967295	0.16%	48	52	171530	0.03%
17	1872	4334410	0.04%	49	2982	4668001	0.06%
18	161	632660	0.03%	50	3041	2398182	0.13%
19	1833	3701628	0.05%	51	3388	279590	1.20%
20	1811	410725	0.44%	52	4753	2113492	0.22%
21	2659	3288266	0.08%	53	3729	244056	1.50%
22	2529	1984643	0.13%	54	5602	1864858	0.30%
23	11755	478409	2.40%	55	1822	2267938	0.08%
24	5281	1491727	0.35%	56	2729	381554	0.71%
25	145	1303608	0.01%	57	5251	1880226	0.28%
26	674	675470	0.10%	58	3484	3403238	0.10%
27	1803	625806	0.29%	59	605	725353	0.08%
28	3617	17713468	0.02%	60	2922	2677842	0.11%
29	2738	7659443	0.04%	61	1184	1506815	0.08%
30	2312	2787239	0.08%	62	2146	1170619	0.18%
31	3952	1949608	0.20%				

4.6 Implementation Issues

4.6.1 The Training and Adaptation Procedure

The MRFLR training procedures consist of detecting cliques and estimating the clique parameters. We first go through the steps and then focus on some specific issues.

1. Preparation of SD models

   In both clique detection and clique parameter estimation, a set of SD models is used as samples of the field variables. Collecting samples from different speakers ensures that the extracted correlations are independent of speakers, thus represent a common set of prior knowledge within acoustic models.

   These SD models can be trained using the traditional EM algorithm if there are enough training data for each speaker, or can be derived from the SI model using MLLR (adapting the SI model to each training speaker) otherwise.

2. Clique detection

   The clique detection step requires massive computations. Correlations between all pairs of points in the field are calculated and those less correlated pairs are discarded. Section 4.6.2 has some discussions on possible pruning that is necessary to accomplish this step, due to the big size of the field.

   The correlation can be either linear or nonlinear. In this thesis we consider two kinds of correlations: the *Pearson* algorithm for linear correlation and the *Spearman* algorithm for rank-based correlations [31].

   Given samples of two points $x^{(r)}$ and $y^{(r)}$, $r = 1...R$, the Pearson linear correlation is calculated as follows:

   $$\xi = \frac{\sum_{r=1}^{R}(x^{(r)} - \bar{x})(y^{(r)} - \bar{y})}{\sqrt{\sum_{r=1}^{R}(x^{(r)} - \bar{x})^2}\sqrt{\sum_{r=1}^{R}(y^{(r)} - \bar{y})^2}} \tag{4.22}$$

   where $\bar{x}$ and $\bar{y}$ are the means of $x^{(r)}$'s and $y^{(r)}$'s, respectively.

   The value ξ lies between -1 and 1, inclusive. The absolute value of ξ taking 1 means a perfect straight-line correlation while a zero value of ξ means not linearly correlated.

The concept of the rank-based correlation is to replace the value of each $x^{(r)}$ by the value of its *rank* among all the other $x^{(r)}$'s in the sample, that is, $1, 2, ..., R$. The resulting list of numbers will be drawn from a perfect uniform distribution function.

The Spearman rank-based algorithm uses the same linear correlation equation as Equation 4.22 with the sample values replaced with ranks for both $x^{(r)}$'s and $y^{(r)}$'s. Theoretically, the rank-based correlation is more robust than the linear correlation because of the known sample distributions.

3. Clique parameter estimation

   Given a set of SD means, the estimation of clique parameters is to minimize the χ^2 merit function (Equation 4.5). The *Linear Fitting* algorithm can be used. It is a standard algorithm to estimate the coefficients by fitting a straight line according to the given two points with uncertainties on both coordinates. The theory behind it can be found in the *Numerical Recipes* [31].

4. Accumulator initialization

   In Equation 4.17 and Equation 4.13, the right parts of the G Z accumulators involve no adaptation data. They serve as initial values for the accumulators and can be calculated before-hand. This feature is nice in practice since all the computations happen in the training stage. There is no additional computation required during adaptation, compared with MLLR.

The adaptation procedures are the same as those in MLLR except for the add-on items for the GZ accumulators.

1. Obtain statistics for each model

   Run the Forward-Backward algorithm to obtain the statistics for each Gaussian mixture component over some adaptation data. The statistics can also be obtained through the Viterbi alignment if speed or memory is in concern.

   The adaptation targets come from either pre-determined resources (in the supervised adaptation mode) or a previous recognition pass (in the unsupervised adaptation mode).

During the process, the G Z matrices are accumulated according to Equation 4.17 and Equation 4.13.

2. Calculate the transformation matrices

   In the batch mode or auto-adaptation mode, the transformation matrices are calculated after all the adaptation data have been observed. In the incremental mode, the transformation matrices are calculated after each incremental step. The calculation follows Equation 4.18.

3. Update the mean vectors

   All the mean vectors are transformed according to Equation 4.8. Certain models may be intentionally untouched for they are less speaker dependent. For example, a silence model and an optional short pause model are often well trained in the SI model and reflect more environmental property rather than acoustic characteristics of a speaker. The silence model is used to model silence of certain duration in speech, and the optional short pause model is used to model the optional short pauses between two words.

4. Perform recognition with the adapted model

   Perform recognition on the test utterances using the adapted acoustic model. For the unsupervised mode, this is a second recognition pass on the test utterances.

4.6.2 Clique Detection

Clique detection is a step in MRFLR to identify cliques. Due to the big size of the field, some sort of pruning schemes have to be used to accomplish the task. For example, the SI model used in this thesis has 7,500 states and each state 12 mixture components, which is about 90,000 mixture components. The field size is $39 \times 90,000$ (39 is the feature dimension). The computation is about $90,000 \times 90,000 \times 39/2 \approx 1.58 \times 10^{11}$ times correlation calculations.

The proposed pruning method groups all the Gaussian mixture components into a hierarchy tree according to their acoustic similarity. The clique detection is proceeded as a top-down process: If two parent nodes are considered correlated, then the detection continues with more details for their child nodes. Otherwise, their child nodes are assumed not correlated.

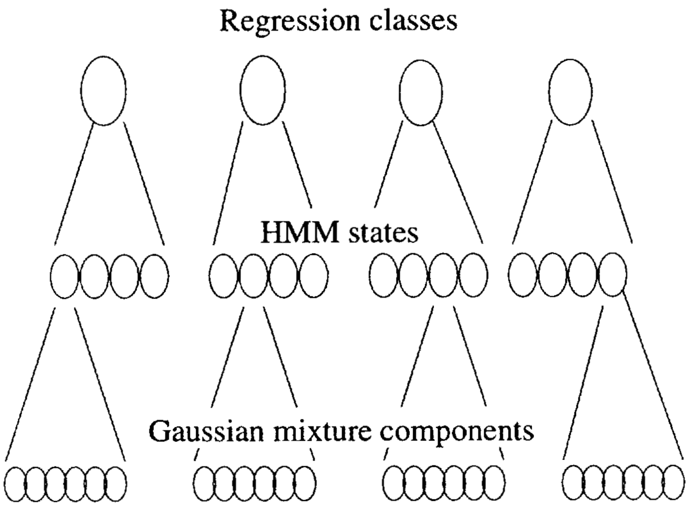

Figure 4.2: Pruning hierarchy for clique detection

The top level considers correlations among regression classes. The middle level considers correlations among HMM states. The bottom level considers correlations among Gaussian mixture components.

As illustrated in Figure 4.2, the three-level pruning-tree is arranged as follows:

- The bottom level

  This level consists of all the Gaussian mixture components.

- The middle level

  The middle level consists of all the HMM states. According to the HMM tying discussed in Section 2.4, each node in the middle level has its child nodes (in the bottom level) that are Gaussian mixture components associated with the corresponding state.

  As different HMM systems have different model configurations, the middle level may have flexible settings accordingly.

- The top level

The first level consists of a set of regression classes. Each regression class has child node that are HMM states in the middle level.

For each feature dimension, the clique detection and pruning proceed as follows:

1. Data preparation (bottom to top)

   (a) Construct a vector for each bottom level nodes. Each element of the vector comes from the mean values of the prepared set of SD models (Section 4.6.1). For example, if a bottom level node is associated with mixture component m and the current feature dimension is d, then the rth element of the vector is the dth element of the mean vector of Gaussian mixture component m in SD model r.

   (b) Calculate a vector for each middle level node as an average over all the vectors in the child nodes.

   (c) Calculate a vector for each top level node as an average over all the vectors in the child nodes.

2. Clique detection (top to bottom)

   (a) Calculate correlations among all the nodes in the top level and drop those pairs of nodes that are less correlated.

   (b) Expand the remaining pairs to the middle level.

   (c) Calculate correlations among the corresponding nodes and drop those pairs of nodes that are less correlated.

   (d) Expand the remaining pairs to the bottom level.

   (e) Calculate correlations among the corresponding nodes and save those cliques that show high correlations.

Let M denote the number of Gaussian mixture components in the acoustic model. Without any pruning, the clique detection step requires $\frac{1}{2}M^2$ times correlation calculations in average for each feature dimension. With the proposed pruning algorithm, the clique detection step requires $\frac{1}{8}r^2M^2$ times correlation calculations in average for each feature dimension. The parameter $0 \leq r \leq 1$ represents an average correlation rate, that is, how many correlation pairs have high enough correlations to be passed to the next level.

From the above analysis, the pruning algorithm does not change the complexity of the clique detection but speeds up the process by reducing a constant factor in the complexity measure.

4.7 The Regression Class Tree

In both MLLR and MRFLR, assigning a transformation matrix to a regression class can be pre-wired (fixed) or through a dynamical approach such as the *regression-class tree* (*regression tree* from now on).

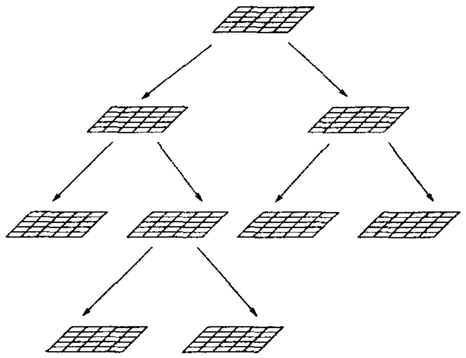

Figure 4.3: A regression class tree

Each tree node is a regression class with each parent node consisting all the Gaussian mixture components of its child nodes. During adaptation, the transformation matrices are assigned through a top-down search to the lowest tree nodes whose frame-counts satisfy a given threshold.

Figure 4.3 illustrates a typical regression tree. Each tree node is a possible regression class. A parent class includes all the Gaussian mixture components of its children. Hence the root node represents the global regression class and the children of each node represent more specific classes.

During adaptation, transformation matrices are assigned through a top-down search to the lowest possible regression classes that satisfy a given frame-count (or an occupation count) threshold. The threshold is set to ensure the robust estimation of each transformation matrix.

Gales [14] presented some of the popular regression-tree generation algorithms. These algorithms use a bottom-up clustering scheme and are driven only by the training data. Hence there exists a problem that the regression tree could be biased by the training data. It is desired that the generation procedure be constrained by some phonetic knowledge.

This thesis proposes a regression-tree generation algorithm, which is an extension of the phonetic decision-tree algorithm [44] widely used in acoustic training. The advantages of using the proposed algorithm are as follows:

1. Combine the phonetic knowledge and the data-driven method together to prevent the tree generation procedure from being biased by the training data.

2. By using similar methods in both acoustic training and regression-tree generation, the resulting regression tree is more similar to the training configuration than those traditional regression tree derived from data-driven-only methods. There is less mismatch between the training and the testing configurations.

As stated in Section 2.4, HMM parameters are tied to balance the modeling ability and trainability. The phonetic decision-tree algorithm is used to cluster acoustically similar HMM states together to provide a state-level tying.

The phonetic decision tree is a binary tree with a yes-or-no phonetic question attached to each node. Initially all the states (usually of a specific central monophone) are placed at the root node. Depending on each answer, the pool of states is split and this continues until the states have trickled down to leaf nodes. All the states in the same leaf node are then tied. For example, in Figure 4.4, the decision tree partitions all the states into six subsets (leaf nodes), and the state parameters are tied together for each subset.

The question attached to each node is selected from a set of phonetic questions so as to maximize the likelihood of the training data given the corresponding state tying. Usually the means vectors, the variance matrices, and the state occupations of single Gaussian distributions are used as sufficient statistics to calculate the likelihood increase. Splitting

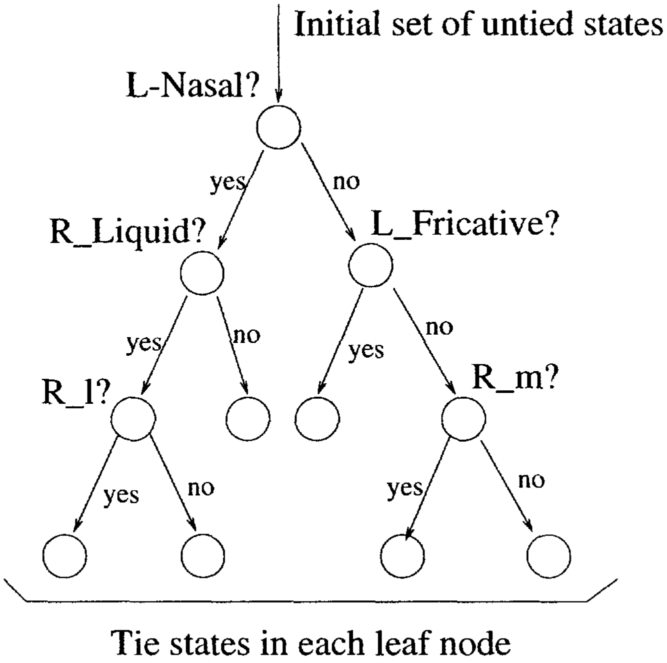

Figure 4.4: Phonetic decision tree

Each phonetic question splits the HMM states into two subsets and increases the overall likelihood of the training data. The questions are selected from a question set so as to maximize the likelihood increases. The HMM states in each leaf-node cluster will be tied together to reduce the total number of parameters in the acoustic model.

any pool into two will increase the likelihood since it provides twice as many parameters to model the same amount of data.

The proposed regression-tree generation algorithm has made the following modifications to the decision-tree algorithm:

1. Assume Gaussian mixture components within a HMM state are already acoustically similar enough so that they are by default clustered together (belonging to the same regression classes).

2. The phonetic questions are extended to include both state questions and central monophone questions as follows:

    (a) state questions: For example, is this state the central state of some HMMs?

    (b) left context questions: For example, is the left context vowel?

(c) right context questions: For example, is the right context nasal?

(d) central monophone questions: For example, is the central monophone 'a'?

The procedure of the regression-tree generation algorithm is as follows:

1. Create the root node and associate it with all the HMM states.
2. For each leaf node, evaluate each question in the question set: Calculate the likelihood increase if the HMM states are split according to the current question. The best question is recorded that results in the maximum likelihood increase.
3. Select the leaf node that has the maximum likelihood increase.
4. If the selected leaf node has the total likelihood value below a given stopping threshold, the algorithm stops.
5. Split the selected node into two child nodes. The left child is associated with HMM states (from the selected node) that answers yes to the best question and the right child is associated with HMM states that answers no.
6. Repeat Step 2 to 5 until the likelihood increase below a given threshold.

The resulting algorithm generates a regression tree for the Gaussian mixture components. For example, in Figure 4.5, nine regression classes are generated. During adaptation, transformation matrices will be assigned to the lowest tree nodes whose frame counts satisfy a given threshold.

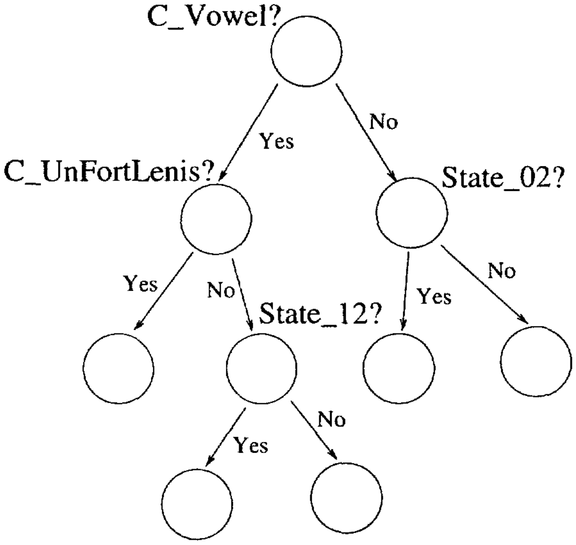

Figure 4.5: Phonetic regression class tree

With the expanded question set, the phonetic regression tree clusters the Gaussian mixture components into regression classes. The question "C_Vowel" asks if the central monophone is vowel and the question "state_12" asks if the Gaussian mixture component belongs to state 1 or 2.

Chapter 5

The Baseline System

5.1 Baseline System

The 1998 OGI-FONIX [43] large vocabulary system was used as the baseline system. It is a continuous HMM-based system designed mainly for transcribing Broadcast News. This chapter presents the training and recognition sub-systems used in the evaluation of the MRFLR algorithm.

To speed up the experimental cycle without losing much generality of the test environments, the ARPA WSJ 20K task has been chosen as the test bed. With about 20,000 words in the system vocabulary, the test set has an out-of-vocabulary rate of 4%.

5.2 Acoustic Feature

The acoustic front end extracts 12 mel frequency cepstral coefficients, energy, and their first and second derivatives (total of 39 parameters) from every 10 ms frame. *Cepstral Mean Normalization* is performed on each utterance.

5.3 Acoustic Training

The standard SI-284 acoustic training set was used to train the SI model. The SI-284 training set contains 283 speakers, 110 to 150 sentences for each speaker. The acoustic model is state-tied based on the phonetic decision-tree algorithm and trained with the standard EM algorithm. The training procedure is shown in Figure 5.1 and explained as follows:

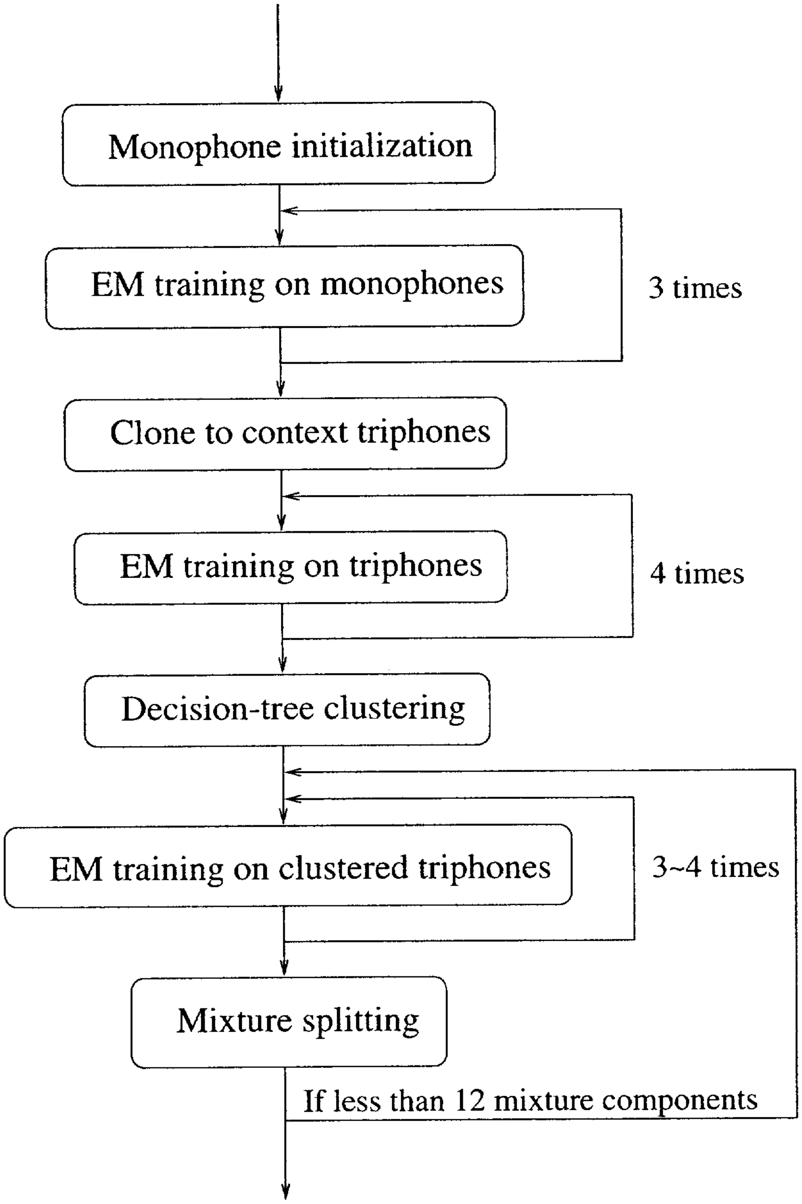
Monophone initialization
EM training on monophones
3 times
Clone to context triphones
EM training on triphones
4 times
Decision-tree clustering
EM training on clustered triphones
3~4 times
Mixture splitting
If less than 12 mixture components

Figure 5.1: The acoustic training procedure

1. Monophone initialization and training

   The monophone set has 48 monophones. They were initialized from the TIMIT corpus[1] and then trained using the traditional EM algorithm on the SI-284 training set. Three iterations of the EM training were performed to ensure the convergence.

2. Context triphone training

   The monophone models were cloned to their corresponding context triphones (only for those observed triphones in the training set). Another four iterations of the EM training were done to tune the triphone models to the SI-284 training set and to collect statistics for the next clustering step.

3. Clustering and state tying

   The phonetic decision-tree algorithm was used to perform state clustering and then state-tied to the corresponding mixture parameters (mixture weight, mean vector and covariance matrix). Up to this step there is only one Gaussian mixture component for each HMM state.

4. Clustered triphone training

   The clustered triphone model was again trained with the EM algorithm on the SI-284 training set. After every three or four iterations of training, each HMM state in the model was split to more mixture components. The final acoustic model has 7500 states with each state consisting of 12 Gaussian mixture components.

5.4 Language Model Training and The Lexicon

The CMU-Cambridge language model package V2.0 was used. The text materials included the WSJ language model data obtained from LDC (Linguistic Data Corporation). The *Good-Turing* method was used to estimate the back-off trigram and bigram language models. They contained 17M trigrams and 7M bigrams respectively. The perplexity on the ARPA Nov92 test set is 129.

[1] The TIMIT corpus of read speech includes time-aligned phonetic transcription of American English of 630 speakers and 8 major dialects

According to the occurrence of words in the text training corpus, the lexicon was selected to have 19979 words. The dictionary was selected from the most frequent pronunciations in the SI-284 training set and has 23714 pronunciation entries (including multiple pronunciations for some words).

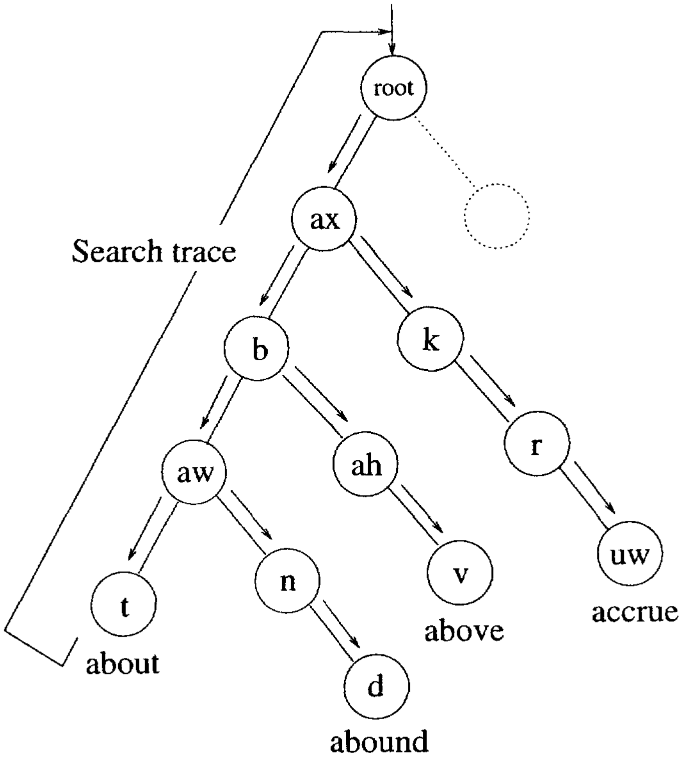

Figure 5.2: Recognition on the lexical tree

Words with the same prefix pronunciations share the same tree paths. The search is performed as the search paths passing through the lexical tree.

5.5 Tree-search Recognition

In recognition, the search for the most likely word sequence is based on the *lexical tree*, a prefix pronunciation tree, as illustrated in Figure 5.2. Each tree node (except the pseudo root node) is a phoneme of some word pronunciations. Words with the same prefix pronunciations share the same tree paths. Each leaf node is associated with a word whose pronunciation is represented by the path from the pseudo root node to the leaf node.

The recognition is processed as the search paths passing through the lexical tree. The search is initialized from the pseudo root node. The context triphone is rendered and the corresponding acoustic model is used to calculate the acoustic score. The search path splits at the tree branches so that every possible path will be traversed.

At each leaf node, the language model score is attached. The word associated with the leaf node is recorded as the search history and the search path goes back to the pseudo root node for searching the successive words.

At the end of search, the search path with the best likelihood score is traced back to retrieve the most likely word sequence and output as the recognition result.

During the search, many pruning strategies are explored so as not to exhaust memory. For example, the state pruning method discards those search paths that have too low acoustic scores (unlikely to be the best at the end of the search). More details of efficient search algorithms can be found in Young's papers [37, 29], and Ravishankar's thesis [33].

5.6 Baseline

This baseline system achieved 12.6% WER on the Nov92 test set, as in Table 5.1. There are 8 speakers in the test set with each speaker 41 to 45 utterances.

Table 5.1: Baseline WER on WSJ20K Nov92

Speaker	Sub	Del	Ins	WER
440	9.1	1.2	3.2	13.4
441	15.3	1.9	3.6	20.8
442	8.5	0.4	3.1	12.0
443	9.9	1.1	2.9	14.0
444	9.4	1.4	3.1	14.0
445	4.6	0.8	1.6	7.0
446	5.9	0.8	2.3	9.0
447	8.6	0.9	2.0	11.5
Sum/Avg	8.9	1.1	2.7	12.6

Chapter 6

Evaluation of MRFLR

This chapter studies the performance of the MRFLR adaptation presented in Chapter 4 by conducting a series of experiments on the ARPA speech corpora. The performance of MRFLR will be compared with that of the most popular MLLR adaptation since MRFLR is a direct extension of MLLR and also since MLLR is the most widely used adaptation algorithm in speech recognition systems. These experiments include:

1. MRFLR vs MLLR with supervised sparse data in the batch mode
2. MRFLR vs MLLR with unsupervised sparse data in the auto-adaptation mode
3. MRFLR with different scaling factors
4. MRFLR vs MLLR with multiple transformation matrices
5. MRFLR vs MLLR in the incremental mode

6.1 Statistical Analysis

For statistical analysis on the performances of two systems: MRFLR vs MLLR, the statistical tool provided along with the WSJ20K benchmark test is used. The tool has implemented: (1) matched-pairs test on sentence segments, (2) Wilcoxon signed-rank test on speaker accuracy rate.

Matched-pairs Test

The matched-pairs test is based on sentence segments. The sentence segments are detected using a state-machine illustrated in Figure 6.1. The term "correct" means both the two systems correctly recognize the current word. The term "error" means at least one system incorrectly recognizes the current word. A sentence segment is thus a sequence of words with the segment's end being given the number (min_good) of correctly recognized words for both systems. The value min_good is set to one word in this thesis.

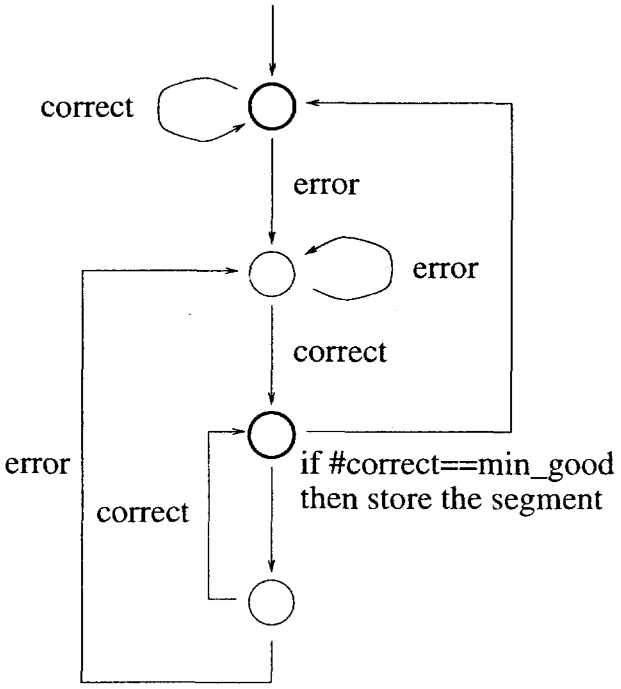

Figure 6.1: State machine for locating sentence segmentations

The term "correct" means both the two systems correctly recognize the current word. The term "error" means at least one system incorrectly recognizes the current word. A segment is a sequence of words that end with given the number (min_good) of correctly recognized words.

For each segment i, the variate d_i is defined as the difference of the number of misrecognized words from the two systems. The hypotheses of the matched-pair test are as follows, denoting $\bar{d}$ as the mean of the differences:

$$\textit{The null hypothesis } H_0: \quad \bar{d} = 0$$

$$\textit{The alternative hypothesis } H_a: \quad \bar{d} \neq 0$$

Defining the standard normalized variate $z = \sqrt{n}\bar{d}/\sigma$ (where n is the sample size and σ is the standard deviation), the decision of the matched-pair test can be made on the following decision rule: reject H_0 if $|z| > z_\alpha$, where z_α is a *critical value* [27] from a standard normal table corresponding to the confidence level $100(1-\alpha)\%$.

Wilcoxon Signed Rank Test

The Wilconxon test [41] is a non-parametric test that utilizes information on both the signs and the magnitudes of the differences of the speaker accuracy in the two systems. Define d_i the differences of the word accuracy rates of speaker i of the two systems. Omit zero differences and reduce the sample size n if necessary. The hypotheses of the Wilconxon test are as follows, denoting $\bar{d}$ the mean of the differences:

$$\begin{aligned} \textit{The null hypothesis } H_0 &: \quad \bar{d} = 0 \\ \textit{The alternative hypothesis } H_a &: \quad \bar{d} \neq 0 \end{aligned}$$

Order the absolute values of the differences $|d_i|$ from smallest to largest, and assign ranks from 1 to n to the ordered absolute values. Determine the sum T_+ of the ranks assigned to the positive d_i and the sum T_- of the ranks assigned to negative d_i. Let $T = min(T_+, T_-)$.

The decision of the Wilconxon test is made on the following rule: reject H_0 if the approximate normal test statistic $z = \frac{T-n(n+1)/4}{\sqrt{n(n+1)(2n+1)/24}}$ is in the lower $100(\alpha/2)\%$ tail of the standard normal distribution.

6.2 Speaker Dependent Training Data

The SD models required by the MRFLR training procedures were obtained using the same SI-284 training data. Instead of using the traditional ML algorithm, the MLLR adaptation was used to adapt the SI model for each training speaker, since the speaker dependent data were not enough to fully train an acoustic model.

The MLLR adaptation used a regression tree that had 64 leaves at the bottom and in total 127 regression classes. The occupation threshold was set to 1500 to for pruning the regression tree.

6.3 Clique Detection

The clique detection was based on a three-level pruning tree (Section 4.6.2) as follows:

1. The top level consists of 256 clusters of Gaussian mixture components, obtained from the phonetic decision-tree clustering. The rank-based correlation was calculated using the Spear algorithm. Those pairs were dropped whose correlation value was less than 0.6. The remaining pairs were expanded to the middle level.

2. The middle level consists of 7,503 clusters of Gaussian mixture components, the number identical to the HMM states in the model. As in Step 1, the less correlated pairs were dropped and the remaining pairs were expanded to the bottom level.

3. The bottom level consists of all the 90,022 mean vectors. The highly correlated pairs detected in this level were saved as the detected cliques of the field.

Table 6.1 shows the number of cliques in each level.

Table 6.1: Clique detection stages

level	cluster number	clique number
top	256	126594
middle	7503	3172939
bottom	90022	11104362

6.4 Single Transform Matrix With Sparse Data

As stated in Chapter 4, the MRFLR adaptation constrains the matrix estimation with the prior knowledge of acoustic models. It is particularly useful for sparse adaptation data cases. When the amount of adaptation data is limited, the adaptation relies more on the prior knowledge rather than the information extracted from the adaptation data.

6.4.1 Data Preparation

There are 8 speakers in the WSJ20K Nov92 test set. Each speaker has two sets of data, one for (adaptation) enrollment and the other for testing. Each data set has 41 to 43 utterances, whose lengths are from 6 to 11 seconds with a few short exceptions of 2 to 3 seconds.

To draw the performance curve of MRFLR, a series of enrollment sets should be constructed with each set being the superset of its immediate precedent. Such enrollment sets could be constructed as follows: the first set contained the first test utterance, and the second set contained the first and the second test utterances, etc. However, since each utterance has a relatively long duration, the amount of adaptation data would soon reach a point where the performances of MRFLR and MLLR have no significant difference. Therefore, for a thorough study, the enrollment sets are actually constructed as follows:

1. Generate an enrollment list by including all the enrollment utterances and then randomly shuffling the orders.

2. For each x from 1 second to 20 seconds with an incremental step of one second, construct an enrollment set by selecting the first several utterances from the enrollment list. The summation of the length of these utterances matches length x. If an utterance is too long to be included, it is chopped at certain word boundary. The word boundaries are obtained from either a force-alignment pass (in the supervised mode) or a previous recognition pass (in the unsupervised mode) on the utterance.

6.4.2 Supervised Batch Mode

The supervised batch mode tests were performed on the Nov92 test set. A series of enrollment sets was generated from the Nov92 enrollment data using the method described in Section 6.4.1. Table 6.2 shows the actual lengths of the enrollment sets and their monophone coverage (averaged over each speaker). Figure 6.2 shows the performance curves of MRFLR vs MLLR. Both these two algorithms used a global transformation matrix.

For the performance curve of MLLR, with adaptation data less than 6 seconds, MLLR makes a terrible estimation of the transformation matrix and the performance curve is well above the baseline due to the lack of adaptation data. In practical applications, it is desired not to turn on MLLR until the amount of adaptation data accumulates. With adaptation

Table 6.2: The supervised adaptation set

adaptation set	actual length (s)	monophone coverage (%)
1s	1.28	14
2s	2.22	31
3s	3.26	43
4s	4.37	51
5s	5.31	56
6s	6.46	58
7s	7.18	62
8s	8.23	66
9s	9.25	69
10s	10.33	72
11s	11.32	72
12s	12.23	74
13s	13.26	76
14s	14.27	77
15s	15.23	77
16s	16.23	78
17s	17.45	79
18s	18.32	80
19s	19.23	82
20s	20.38	84

data from 6 to 13 seconds, MLLR begins to be effective. The adaptation performance is getting better as more adaptation data are available. With adaptation data more than 13 seconds, the performance curve becomes stable with only small glitches.

For the performance curve of MRFLR, the recognition performance is improved in all the tests (compared with the baseline), which clearly shows us the advantage of incorporating prior knowledge. From 1 second to 13 seconds, the MRFLR performance curve has a smaller slope than the MLLR curve, which indicates that MRFLR is less sensitive to the adaptation data due to the constraint from the correlations. MRFLR continuously outperforms MLLR until the amount of adaptation data reaches the 13-second point where both MRFLR and MLLR become stable. There are some *betters and worses* during the merging period.

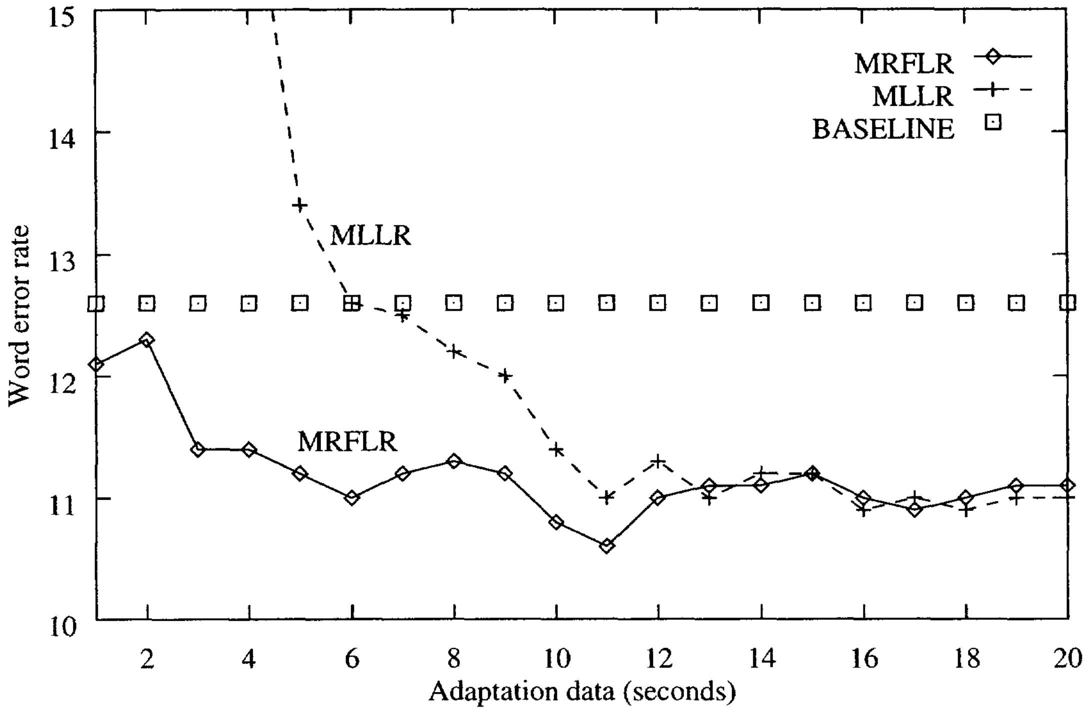

Figure 6.2: MRFLR vs MLLR with supervised sparse data

Statistical analyses were performed for each test set. Based on a 95% confidence interval, the matched-pairs tests show that MRFLR and MLLR are different with adaptation data from 1 second to 12 seconds. For example, at the 12-second point, there are totally 378 segments detected. The mean of the differences (of the number of mis-recognized words) is -0.053 and the standard deviation is 0.475. The z statistic is -2.167, whose absolute value is greater than $z_\alpha = 1.960$. Therefore the null hypothesis is rejected.

Based on the 95% confidence interval, the Wilcoxon tests show that MRFLR and MLLR are different with adaptation data from 1 second to 10 seconds. For example, at the 10-second point, Table 6.3 shows the Wilcoxon calculation table. The sum of seven positive ranks is 34 and the sum of one negative rank is 2. The z statistic is -2.24, whose absolute value is greater than $z_\alpha = 1.960$. Therefore, the null hypothesis is rejected.

Table 6.3: The Wilcoxon test for supervised 10-second data

speaker	Word Correct Rate (%)		difference	rank	signed rank
	MRFLR	MLLR			
440	90.63	90.34	0.29	3	3
441	83.98	82.49	1.50	8	8
442	88.58	88.58	0.00	1	1
443	88.24	87.39	0.84	7	7
444	87.65	87.05	0.60	4	4
445	92.87	93.00	-0.13	2	-2
446	90.59	89.83	0.76	6	6
447	90.30	89.59	0.70	5	5

6.4.3 Unsupervised Auto-adaptation Mode

The unsupervised auto-adaptation tests were performed on the Nov92 test set. A series of enrollment sets was generated from the same test set (using targets from a previous recognition pass). Table 6.4 shows the actual lengths of the enrollment sets and their monophone coverage (averaged over each speaker). Figure 6.3 shows the performance curves of MRFLR and MLLR. Both methods used a global transformation matrix.

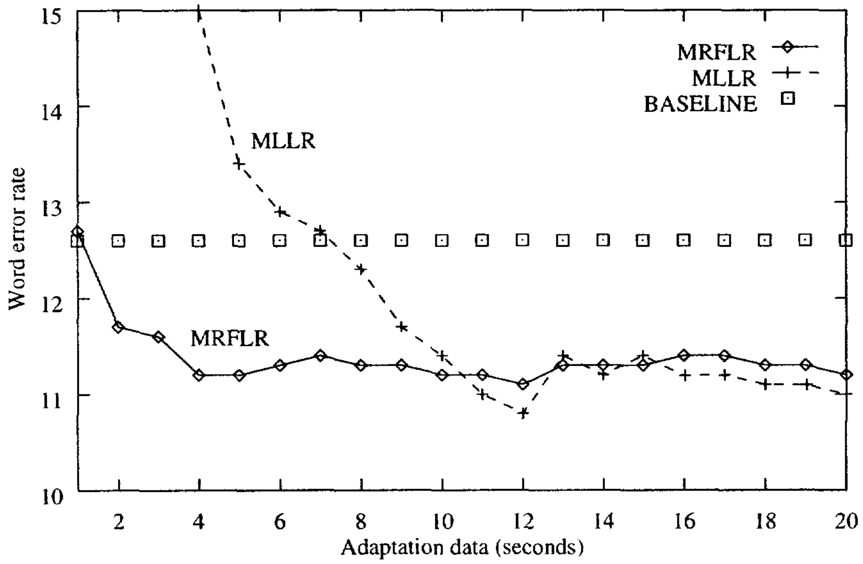

Figure 6.3: MRFLR vs MLLR with unsupervised sparse data

Table 6.4: Unsupervised adaptation Set

adaptation set	actual length (s)	monophone coverage (%)
1s	1.07	21
2s	2.07	38
3s	2.88	44
4s	3.89	51
5s	4.93	57
6s	5.77	61
7s	6.59	64
8s	7.36	67
9s	8.27	71
10s	9.08	72
11s	9.96	75
12s	11.07	76
13s	12.06	77
14s	12.73	78
15s	13.55	78
16s	14.50	79
17s	15.39	80
18s	16.22	80
19s	17.23	82
20s	18.13	82

The curves in Figure 6.3 are like those in Figure 6.2 although there are more observable glitches due to errors in the enrollment references.

For the performance curve of MLLR, with adaptation data less than 7 seconds, MLLR makes poor adaptations and the performance curve is well above the baseline. With adaptation data from 7 seconds to 12 seconds, MLLR begins to improve the recognition performance. After the 13-second point, the MLLR performance becomes stable.

Compared with the baseline, MRFLR improves the recognition performance in all the tests except the one-second test set, which again shows us the strength of utilizing knowledge constraint in speaker adaptation.

MRFLR outperforms MLLR with adaptation data less than 10 seconds. The smaller slope of the performance curve indicates that MRFLR is less sensitive to adaptation data as already observed in the supervised experiments. MRFLR converges to the MLLR performance with more than 13 seconds adaptation data.

For the statistical analysis results, based on the 95% confidence interval, the matched-pairs tests show significant differences of the two systems with adaptation data from 1 second to 8 seconds. For example, at the 8-second point, totally there are 418 segments detected. The mean of the differences is -0.117 and the standard deviation is 0.861. The z statistic is -2.783, whose absolute value is greater than $z_\alpha = 1.960$. Therefore, the null hypothesis is rejected.

Table 6.5: The Wilcoxon test for unsupervised 5-second data

	Word Correct Rate (%)				signed
speaker	MRFLR	MLLR	difference	rank	rank
440	90.19	90.78	-0.59	1	-1
441	82.49	74.10	8.38	8	8
442	88.30	89.00	-0.70	2	-2
443	86.83	81.56	5.28	7	7
444	87.95	85.57	2.38	6	6
445	93.14	92.19	0.94	3	3
446	91.65	90.59	1.06	5	5
447	89.76	88.70	1.05	4	4

With the same confidence interval, the Wilcoxon tests show significant differences of the two systems from 1 second to 5 seconds. For example, at the 5-second point, Table 6.5 shows the calculation table. The sum of the 6 positive ranks is 33 and the sum of the 2 negative ranks is 3. The z statistic is -2.10, whose absolute value is greater than $z_\alpha = 1.960$. Therefore, the null hypothesis is rejected.

6.5 Scaling Factor of MRFLR

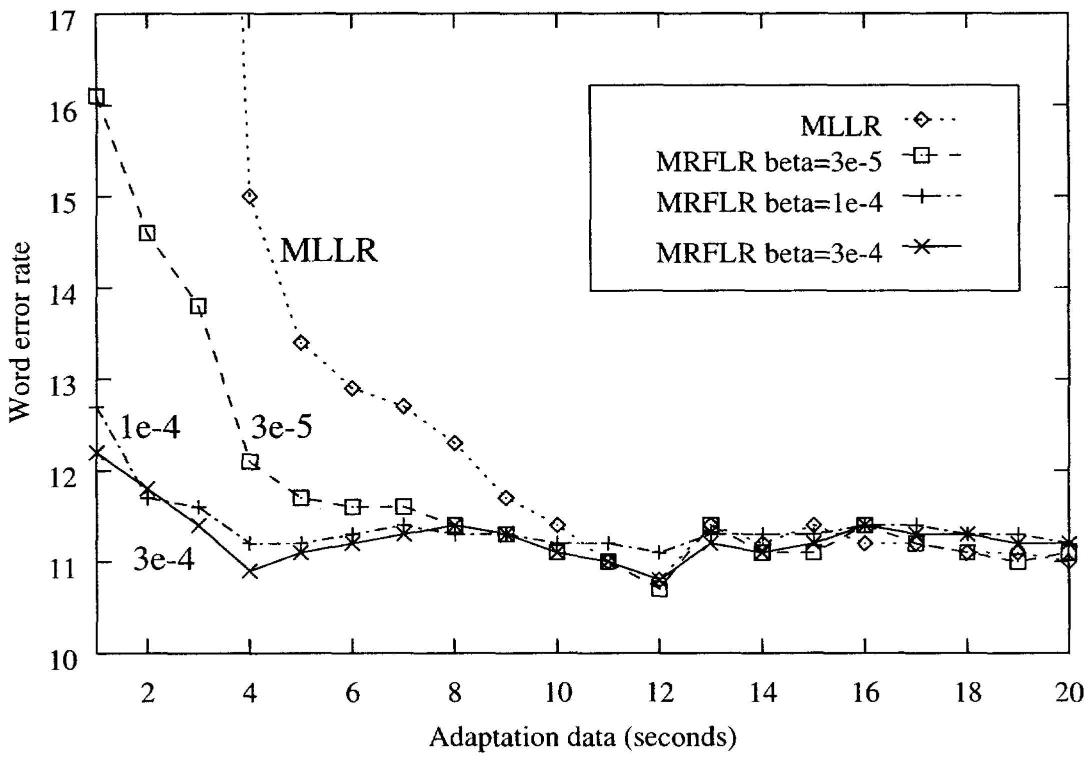

Figure 6.4: MRFLR with different scaling factors

In MRFLR solution (Equation 4.18), the scaling factor β controls the influence of knowledge constraints. Figure 6.4 shows the MRFLR performances with different scaling factors. The MLLR performance curves are also shown for comparison.

When β is very small, the MRFLR curve is close to the MLLR curve. As an extreme case, MRFLR is identical to MLLR if $\beta = 0$.

Turning the scaling factor bigger, the knowledge constraints gain more influences over the adaptation data. The performance curve becomes smoother and less sensitive to the adaptation data. However, more adaptation data are required to make MRFLR converge to the MLLR performance.

At present, MRFLR has only implemented with a constant scaling factor for all the cases. Theoretically, for a little amount of adaptation data, a large scaling factor is called for putting more emphases on the prior knowledge. When there are more adaptation data available, it is desirable to have a smaller scaling factor so that MRFLR can quickly converge to the test-speaker. An ideal solution would be to automatically adjust the scaling factor according to the amount of adaptation data. This scheme is being studied.

6.6 Multiple Transform Matrices

To test performances of MRFLR vs MLLR with multiple transformation matrices, a regression class tree was generated using the phonetic decision-tree based generation method described in Section 4.7. There are 32 leaves and totally 63 nodes in the tree.

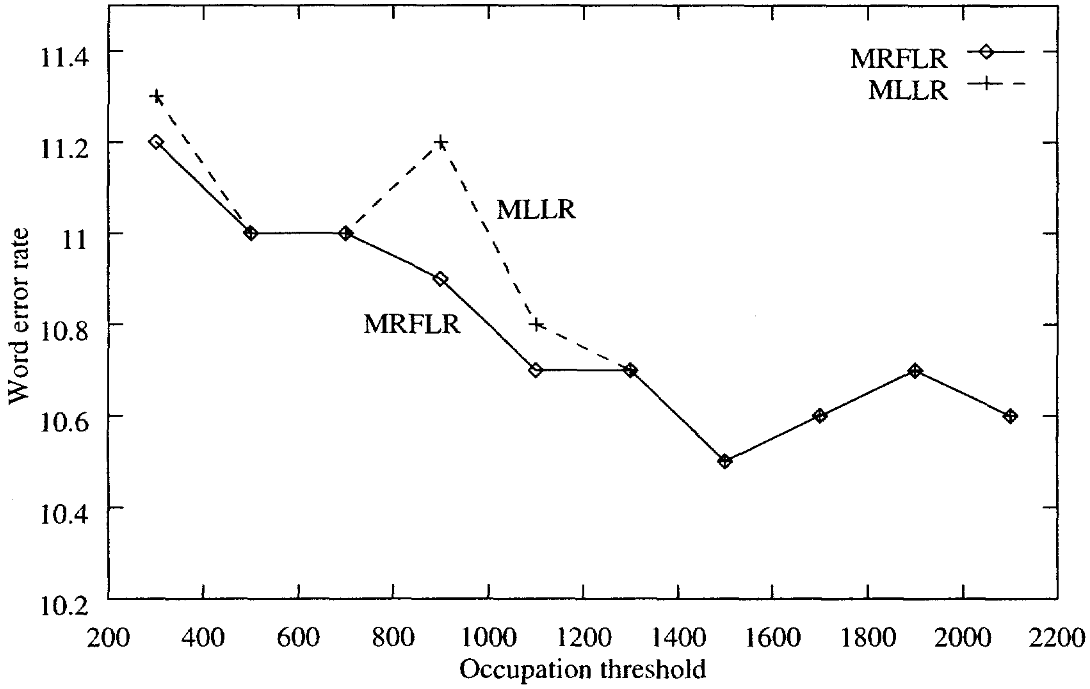

Figure 6.5: MRFLR vs MLLR with multiple matrices

The adaptation tests were performed in the auto-adaptation mode on the Nov92 test set. Figure 6.5 shows the test results given different pruning thresholds. The pruning threshold determines both the number of transformation matrices used in the adaptation and the amount of adaptation data for each transformation matrix.

For both MRFLR and MLLR, the performance peak is at threshold 1500. With a smaller threshold, MRFLR has better performance than MLLR and with a bigger threshold, no performance difference is observed.

This result is not surprising because MRFLR is designed for functioning with limited amount of adaptation data. When adaptation data are sufficient as in the multiple matrices case, the effect of knowledge constraints is overshadowed (Section 4.3). Although MRFLR is unable to outperform MLLR at the peak performance, there are still improvements with off-peak thresholds. In practice, the best pruning threshold may vary with the test environment. A preset pruning threshold may not always be optimal. MRFLR helps prevent performance degradations in worse cases.

6.7 Incremental Adaptation

The incremental mode tests were performed on the Nov92 test set. A series of incremental steps were tested. The same regression tree was used as in the multiple transformation matrices experiments.

Table 6.6: Incremental mode with pruning threshold 1500

Inc Step (#uttr)	MLLR WER	MRFLR WER
2	11.3	11.3
4	11.4	11.4
6	11.4	11.4
8	11.5	11.5
10	11.4	11.4

Table 6.6 shows the performances with a pruning threshold of 1500. No performance difference was observed. This result is consistent with the experimental results in Section 6.6. With a pruning threshold of 1500, there are already enough adaptation data to

overshadow the effect of MRFLR.

Table 6.7: Incremental mode with pruning threshold 1100

Inc Step (#uttr)	MLLR WER	MRFLR WER
2	11.4	11.4
4	11.2	11.2
6	11.6	11.5
8	11.3	11.3
10	11.5	11.5

Table 6.7 shows the performances of MRFLR vs MLLR with a pruning threshold of 1100. Except in the six utterances case, no performance difference was observed.

The explanation to this result lies in the amount of adaptation data. For this test set, the amount of adaptation data in every incremental step is about 12 seconds. Referring to Figure 6.2 and Figure 6.3, MRFLR and MLLR show no significant performance difference with more than 12 seconds adaptation data. Therefore, it is no surprise to see no performance difference between MRFLR and MLLR in incremental mode. It might be good to do more experiments on other tasks to further prove the conclusion. However, in incremental mode, we may see that the amount of adaptation data accumulates quickly as more test utterances become available. Therefore, the effect of knowledge constraints in MRFLR is overshadowed in general.

Chapter 7

Conclusion and Future Work

7.1 Conclusion

This thesis has studied Speaker Adaptation techniques with a limited amount of enrollment data. Existing adaptation methods rely greatly on the enrollment data to extract the test-speaker information. When there are not enough enrollment data to ensure a robust estimation, a poorly structured estimation will often degrade the recognition performance.

The hypothesis of the thesis is that the correlations among acoustic parameters represent the internal structure of speech sounds. These correlations can be viewed as the prior knowledge of acoustic models. By modeling them and explicitly incorporating them into adaptation, the adaptation performance can be improved. The knowledge constraints play the following two roles: (1) Predict acoustic parameters that are not observed in the enrollment data and (2) constrain the possible estimation outputs.

This thesis presents the MRFLR algorithm that uses the MRF theory to model the correlations among acoustic parameters and uses the linear transformation matrices to adapt the mean vectors of a SI model to match the test-speaker. Under the feature-dimension independent assumption, a closed-form solution has been obtained for the MRFLR matrix estimation.

The MRFLR algorithm was evaluated through a series of comparison experiments with the MLLR algorithm. The following experimental results were observed:

1. MRFLR can be used in the batch mode adaptation and in the auto adaptation mode. In these modes, when the amount of adaptation data is limited, MRFLR

outperforms MLLR. As the amount of adaptation data increases, the performance of MRFLR converges to that of MLLR.

2. In extending the MRFLR algorithm to multiple transformation matrices, no significant performance improvement over MLLR was observed. The experiments include using MRFLR with multiple transformation matrices and using MRFLR in the incremental adaptation mode. One reason is that the amount of adaptation data is already enough to robustly estimate the transformation matrices. Therefore the advantage of MRFLR is overshadowed.

The contributions of this thesis are as follows:

1. It provides a novel way of incorporating the prior knowledge of acoustic models into the transformation-based adaptation method.

2. It solves the difficulty that early attempts of incorporating knowledge constraints into transformation-based adaptation algorithms fail to find a closed-form solution for the matrix estimation.

7.2 Extension to This Work

The following issues of MRFLR can be improved in the future.

The scaling factor controls the influence of the prior knowledge to the the adaptation data. As stated in Section 6.5, an ideal solution would be to adjust the scaling factor according to the amount of adaptation data. The more adaptation data, the less the scaling factor.

How many cliques should be appropriate for a practical system is an open question. With more cliques, more correlation relations can be embedded within the prior while the computational requirement increases, as well as the storage requirement. With less cliques, the modeling ability degrades. Hence, the tradeoff has to be made.

In the multiple transformation matrices estimation, the assumption is made that any two transformation matrices are not correlated. This assumption ensures the estimation can be carried out separately for each transformation matrix. However, when using the back-off scheme in regression-tree pruning, there are cases that a parent node and a child node are selected at the same time, which violates the assumption. A joint estimation of these correlated transformation matrices is desired. This problem also exists in MLLR.

The clique detection step in the training procedure requires massive computations and a fairly strong pruning method. The future work is to explore efficient algorithms to speed up the process. It might be to reduce the size of the field, or to find some mathematical equivalence. The multi-scale tree representation of Markov random field is a possible solution to this problem.

In MRFLR, the covariance matrices are not changed. This is partially because of the assumption that the mean vectors represent the most contributions to the speaker variation, and partially because compensating covariances requires huge computation. In MLLR, there is a covariance compensation algorithm which gives an additional 5% to 10% performance improvement. If the covariances can be compensated in MRFLR somehow, and the computation problem can be solved, the resulting adapted model would be more compact for the test-speaker.

Bibliography

[1] ANASTASAKOS, T., MCDONOUGH, J., AND MAKHOUL, J. Speaker adaptive training: A maximum likelihood approach to speaker normalization. In *Proceedings of the 1997 International Conference on Acoustics, Speech, and Signal Processing* (1997), vol. 2, pp. 1043–1046.

[2] ANASTASAKOS, T., MCDONOUGH, J., SCHWARZ, R., AND MAKHOUL, J. A compact model for speaker adaptive training. In *Proceedings of Fourth International Conference on Spoken Language Processing* (1996), vol. 2, pp. 1137–1140.

[3] AUBERT, X., AND THELEN, E. Speaker adaptive training applied to continuous mixture density modeling. In *EUROSPEECH'97* (1997), vol. 4, pp. 1851–1854.

[4] BAUM, L. An inequality and associated maximization techniques in statistical estimation for probabilistic functions of Markov processes. In *Inequalities: Proceedings, academic press* (1972), vol. 3, pp. 1–8.

[5] BESAG, J. Spatial interaction and the statistical analysis of lattice systems. *Journal of Royal Statistical Society, 36* (1974), 192–236.

[6] CHELLAPPA, R., AND KASHYAP, R. L. Digital image restoration using spatial interaction models. *IEEE Transactions on Acoustics, Speech, and Signal Processing, 1* (1982), 461–472.

[7] CHOU, K., WILLSKY, A., AND BENVENISTE, A. Multiscale recursive estimation, data fusion, and regularization. *IEEE Transactions on Automatic Control, 39* (1994), 464–491.

[8] COX, S. Predictive speaker adaptation in speech recognition. *Computer Speech and Language, 9* (1995), 1–17.

[9] DAVIS, S., AND MERMELSTEIN, P. Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences. *IEEE Transactions on Acoustics, Speech, and Signal Processing, 28* (1980), 357–366.

[10] Dempster, A., Laird, N., and Rubin, D. Maximum likelihood from incomplete data via the EM algorithm. *Journal of Royal Statistics Society, Series B, 39* (1977), 1–38.

[11] Dobrushin, R. L. The description of a random field by means of conditional probabilities and conditions of its regularity. *Theory of Probability and its Applications, 13* (1968), 197–224.

[12] Eiden, E., and Gish, H. A parametric approach to vocal tract length normalization. In *Proceedings of the 1996 International Conference on Acoustics, Speech, and Signal Processing* (1996), vol. 1, pp. 346–349.

[13] Gales, M., and Woodland, P. Variance compensation within the MLLR frame work. Tech. Rep. CUED/F-INFENG/TR.242, Cambridge University Engineering Department, February 1996.

[14] Gales, M. J. F. The generation and use of regression class trees for MLLR adaptation. Tech. Rep. CUED/F-INFENG/TR.263, Cambridge University Engineering Department, August 1996.

[15] Gauvain, J.-L., and Lee, C.-H. Maximum a posteriori estimation for multivariate Gaussian mixture observations of Markov chains. *IEEE Transactions on Speech Audio Processing, 2* (1994), 291–298.

[16] Hassner, M., and Sklansky, J. The use of Markov random fields as models of texture. *Computer Graphics and Image Processing, 12* (1980), 357–370.

[17] Hon, H. W., and Lee, K. F. Recent progress in robust vocabulary-independent speech recognition. In *Fourth DARPA Workshop on Speech and Natural Language* (1991). n.p.

[18] Juang, B., Levinson, S., and Sondhi, M. Maximum likelihood estimation for multivariate observations of Markov chains. *IEEE Transactions on Information Theory, 32(2)* (1986), 307–309.

[19] Kannan, A., and Ostendorf, M. Modeling dependence in adaptation of acoustic models using multiscale tree processes. In *Proceedings of European Conference on Spoken Language Processing* (1997), vol. 1, pp. 1863–1866.

[20] Koopman, B. On distributions admitting a sufficient statistic. *Transactions of the American Mathematical Society, 39* (1936), 399–409.

[21] Lee, C.-H., Lin, C.-H., and Juang, B.-H. A study on speaker adaptation of the parameters of continuous density hidden Markov models. *IEEE Transactions on Signal Processing, 39* (1991), 806–814.

[22] Lee, L., and Rose, R. Speaker normalization using efficient frequency warping procedures. In *Proceedings of the 1996 International Conference on Acoustics, Speech, and Signal Processing* (1996), vol. 1, pp. 353–356.

[23] Leggetter, C. J., and Woodland, P. C. Speaker adaptation using linear regression. Tech. Rep. CUED/F-INFENG/TR.181, Cambridge University Engineering Department, 1994.

[24] Leggetter, C. J., and Woodland, P. C. Maximum likelihood linear regression for speaker adaptation of HMMs. *Computer Speech and Language, 9* (1995), 171–186.

[25] Liporace, L. A. Maximum likelihood estimation for multivariate observations of Markov sources. *IEEE Transactions on Information Theory, 28* (1982), 729–734.

[26] Luettgen, M., Karl, W., Willsky, A., and Tenney, R. Multiscale representations of Markov random fields. *IEEE Transactions on Signal Processing, 41* (1993), 3377–3395.

[27] Mason, R. L., Gunst, R. F., and Hess, J. L. *Statistical Design and Analysis of Experiments.* John Wiley & Sons Inc, 1989.

[28] Odell, J. J. *The Use of Context in Large Vocabulary Speech Recognition.* PhD thesis, Queen's College, University of Cambridge, March 1995.

[29] Odell, J. J., Valtchev, V., Woodland, P. C., and Young, S. J. A one pass decoder design for large vocabulary recognition. In *Proceedings of the 1993 International Conference on Acoustics, Speech, and Signal Processing* (1993), vol. 2, pp. 405–410.

[30] Paul, D. B. Experience with a stack decoder-based HMM CSR and back-off n-gram language models. In *Fourth DARPA Workshop on Speech and Natural Language* (1991). n.p.

[31] Press, W. H., Flannery, B. P., Teukolsky, S. A., and Vetterling, W. T. *Numerical Recipes in C.* Cambridge University Press, 1990.

[32] Rabiner, L., and Juang, B.-H. *Fundamentals of Speech Recognition.* Prentice Hall, 1993.

[33] Ravishankar, M. K. *Efficient Algorithms for Speech Recognition.* PhD thesis, Carnegie Mellon University, 1996. CMU-CS-96-143.

[34] Shahshahani, B. M. A Markov random field approach to Bayesian speaker adaptation. Tech. Rep. TR54.885, IBM Technical Report, January 1995.

[35] Shahshahani, B. M. A Markov random field approach to Bayesian speaker adaptation. In *Proceedings of the 1996 International Conference on Acoustics, Speech, and Signal Processing* (1996), vol. 2, pp. 697–700.

[36] Shahshahani, B. M. A Markov random field approach to Bayesian speaker adaptation. *IEEE Transactions on Speech and Audio Processing, 5* (1997), 183–191.

[37] S.J.Young, N.H.Russell, and J.H.S.Thornton. Token passing: A simple conceptual model for connected speech recognition systems. Tech. Rep. CUED/F-INFENG/TR.38, Cambridge University Engineering Department, July 1989.

[38] Spitzer, F. Markov random fields and Gibbs ensembles. *American Mathematical Monthly, 78* (1971), 142–154.

[39] Wegmann, S., McAllaster, D., Orloff, J., and Peskin, B. Speaker normalization on conversational telephone speech. In *Proceedings of the 1996 International Conference on Acoustics, Speech, and Signal Processing* (1996), vol. 1, pp. 339–341.

[40] Wiley, J., and Sons. *Bayesian Statistics: Principles, Models, and Applications.* John Wiley & Sons, Inc, 1989.

[41] Wolfe, D. A. *Non-parametric Statistical Methods.* John Wiley & Sons Inc, 1973.

[42] Woodland, P. C., Leggetter, C. J., Odell, J. J., Valtchev, V., and Yong, S. J. The 1994 HTK large vocabulary speech recognition system. In *Proceedings of the 1995 International Conference on Acoustics, Speech, and Signal Processing* (1995), vol. 2, pp. 73–76.

[43] Wu, X., Liu, C., Yan, Y., Kim, D., Cameron, S., and Parr, R. The 1998 OGI-FONIX broadcast news transcription system. In *Proceedings, Broadcast News Transcription and Understanding Workshop* (1999). n.p.

[44] Yang, S. J., Odell, J. J., and Woodland, P. C. Tree based state tying for high accuracy acoustic modeling. In *Proceedings of ARPA Workshop on Human Language Technology* (1994), pp. 286–291.

[45] ZAVALIAGKOS, G. *Maximum A Posteriori Adaptation Techniques for Speech Recognition.* PhD thesis, Northeastern University, October 1995.

Appendix A

Notations

The following notations have been used in the thesis:

D	the feature dimension
T	the time period of interest
K_j	the number of mixture components within state j
N	the number of states in the model
M	the number of mixture components in the model
R	the number of speakers in the training set
W	the MLLR or MRFLR transformation matrix
W_i	the ith row vector of the MLLR or MRFLR transformation matrix
λ	the acoustic model
$\hat{\lambda}$	the adapted acoustic model
o_t	the speech observation at time t
q_t	the HMM state at time t
ω_{jk}	the mixture weight of mixture component k of state j
ω_m	the mixture weight of mixture component m
μ_{jk}	the mean vector of mixture component k of state j
μ_m	the mean vector of mixture component m
$\tilde{\mu}_m$	the augmented mean vector of mixture component m, $\tilde{\mu}_m = \left[\mu_m^T \; 1\right]^T$
$\hat{\mu}_m$	the adapted mean vector of mixture component m
Σ_{jk}	the covariance matrix of mixture component k of state j
Σ_m	the covariance matrix of mixture component m
$b_j(o_t)$	the output probability density of state j given observation o_t

a_{q_{t-1},q_t}	the transition probability from state q_{t-1} to state q_t
$\alpha_t(i)$	the forward probability
$\beta_t(i)$	the backward probability
$\gamma_t(j,k)$	the probability of observation o_t at mixture component k of state j
$\gamma_t(m)$	the probability of observation o_t at mixture component m
$N(o_t;\mu_{jk},\Sigma_{jk})$	the Gaussian density function with mean μ_{jk} and covariance matrix Σ_{jk}
$G(i)$	an auxiliary matrix for the MLLR or MRFLR deductions, $(D+1)\times(D+1)$
Z	an auxiliary matrix for the MLLR or MRFLR deductions, $D\times(D+1)$
$V(m)$	an auxiliary matrix for the MLLR deductions, $D\times D$ diagonal
$V^{(1)}(m)$	an auxiliary matrix for the MRFLR deductions, $D\times D$ diagonal
$V^{(2)}(c)$	an auxiliary matrix for the MRFLR deductions, $D\times D$ diagonal
$D(m)$	an auxiliary matrix for the MLLR deductions, $(D+1)\times(D+1)$
$D^{(1)}(m)$	an auxiliary matrix for the MRFLR deductions, $(D+1)\times(D+1)$
$D^{(2)}(c)$	an auxiliary matrix for the MRFLR deductions, $(D+1)\times(D+1)$
Q	the Markov random field
z	the normalizing constant in the joint distribution of MRF Q
β	the scaling constant in the joint distribution of MRF Q
x_c, y_c	the two points in clique c
a_c, b_c	the coefficients in the linear fitting $y_c = a_c + b_c x_c$
a, b	the simplified notation of a_c and b_c
λ_{min}	the normalizing constant in the linear fitting
Ω_s	the area of points that show dependency to point s
$V_c(Q)$	the potential function of clique c
$\chi^2(a_c, b_c)$	the χ^2 merit function
$x_c^{(r)}, y_c^{(r)}$	the rth sample of variable x_c or y_c
x, y	the simplified notation of x_c and y_c
$\hat{x}, \hat{y}$	the simplified notation of x_c and y_c in the adapted field
$\sigma_{x_c}, \sigma_{y_c}$	the standard deviation of point x_c or y_c
ω_c	the notation of $1/\left(\sigma_{y_c}^2 + b_c^2\sigma_{x_c}^2\right)$
ω	the simplified notation of ω_c

μ_x, μ_y	the corresponding column vector (mean vector) of point x or y
$\tilde{\mu}_x, \tilde{\mu}_y$	the corresponding augmented column vector (mean vector) of point x or y
$\hat{\mu}_x, \hat{\mu}_y$	the corresponding adapted column vector (mean vector) of point x or y
$E^{(d)}$	the auxiliary matrix, $D \times D$, $E_{ij}^{(d)} = \begin{cases} 1 & i = j = d \\ 0 & otherwise \end{cases}$
$E_d^{(d)}$	the dth column vector of $E^{(d)}$
ξ	the correlation measure

Appendix B

Tables of Experimental Results

B.1 MRFLR vs MLLR with supervised data

Figure 6.2 corresponds to scores in Table B.1.

Table B.1: MRFLR vs MLLR with supervised sparse data

data (s)	MLLR (WER%)	MRFLR (WER%)
BASELINE	12.6	12.6
1	75.8	12.1
2	72.5	12.3
3	24.4	11.4
4	16.5	11.4
5	13.4	11.2
6	12.6	11.0
7	12.5	11.2
8	12.2	11.3
9	12.0	11.2
10	11.4	10.8
11	11.0	10.6
12	11.3	11.0
13	11.0	11.1
14	11.2	11.1
15	11.2	11.2
16	10.9	11.0
17	11.0	10.9
18	10.9	11.0
19	11.0	11.1
20	11.0	11.1

B.2 MRFLR vs MLLR with unsupervised data

Figure 6.3 corresponds to scores in Table B.2.

Table B.2: MRFLR vs MLLR with unsupervised sparse data

data (s)	MLLR (WER%)	MRFLR (WER%)
BASELINE	12.6	12.6
1	95.8	12.7
2	43.9	11.7
3	30.1	11.6
4	15.0	11.2
5	13.4	11.2
6	12.9	11.3
7	12.7	11.4
8	12.3	11.3
9	11.7	11.3
10	11.4	11.2
11	11.0	11.2
12	10.8	11.1
13	11.4	11.3
14	11.2	11.3
15	11.4	11.3
16	11.2	11.4
17	11.2	11.4
18	11.1	11.3
19	11.1	11.3
20	11.0	11.2

B.3 MRFLR vs MLLR with multiple matrices

Figure 6.5 corresponds to scores in Table B.3.

B.4 Scaling Factor of MRFLR

Figure 6.4 corresponds to scores in Table B.4.

Table B.3: MRFLR vs MLLR with multiple matrices

threshold	MLLR (WER%)	MRFLR (WER%)
300	11.3	11.2
500	11.0	11.0
700	11.0	11.0
900	11.2	10.9
1100	10.8	10.7
1300	10.7	10.7
1500	10.5	10.5
1700	10.6	10.6
1900	10.7	10.7
2100	10.6	10.6

Table B.4: Scaling factors of MRFLR

data (s)	MLLR	MRFLR 3e-5	MRFLR 3e-4	MRFLR 1e-4
1	95.8	16.1	12.2	12.7
2	43.9	14.6	11.8	11.7
3	30.1	13.8	11.4	11.6
4	15.0	12.1	10.9	11.2
5	13.4	11.7	11.1	11.2
6	12.9	11.6	11.2	11.3
7	12.7	11.6	11.3	11.4
8	12.3	11.4	11.4	11.3
9	11.7	11.3	11.3	11.3
10	11.4	11.1	11.1	11.2
11	11.0	11.0	11.0	11.2
12	10.8	10.7	10.8	11.1
13	11.4	11.4	11.2	11.3
14	11.2	11.1	11.1	11.3
15	11.4	11.1	11.2	11.3
16	11.2	11.4	11.4	11.4
17	11.2	11.2	11.3	11.4
18	11.1	11.1	11.3	11.3
19	11.1	11.0	11.2	11.3
20	11.0	11.1	11.2	11.2

Biographical Note

Xintian Wu was born in Shanghai China on March 23, 1970. He attended Tsinghua University, Beijing, China from 1988 to 1996, and finished his Bachelor Degree (93) and Master Degree (96) both in Electronic Engineering. His major was *Speech and Communication.*

From 1996 to 2000, Xintian Wu pursued his Ph.D. degree in Computer Science at Oregon Graduate Institute of Science and Technology. His major was large vocabulary continuous speech recognition focusing on speaker adaptation techniques.

Starting from 1997, Xintian Wu actively participated in the development of the OGI large vocabulary speech recognition system, which later became the 1998 OGI-FONIX Broadcast News Transcription system. The system competed in the NIST Broadcast News benchmark tests in 1997 and 1998. His involvement during the period was as follows:

Implementation of standard training and decoding algorithms including the MFCC (and MFPLP) feature extraction algorithm, embedded EM training algorithm, decision-tree algorithm, tree/graph Viterbi search algorithm, word-graph generation and expansion algorithm and the MLLR adaptation algorithm

Improvement of various algorithms for performance and efficiency. These improvements included the tree-structure re-entry search, the word-graph expansion and re-merging, the phonetic decision-tree based regression tree generation, and the parallelization of the training and decoding algorithms.

Research on speaker adaptation included the development of the MRFLR adaptation algorithm.

The following areas were of his interest: applications of speech recognition techniques, researches on speech recognition algorithms, or other related speech language applications and researches.

List of Publications:

- Wu X., and Yan Y. Linear regression under maximum a posteriori criterion with Markov random field prior. In *Proceedings of the 2000 International Conference on Acoustics, Speech, and Signal Processing* (2000).
- Wu X., and Yan Y. Development of the 1998 OGI-FONIX broadcast news transcription system. In *EUROSPEECH'99* (1999), vol. 2, pp. 683-686.
- Wu X., Liu C., Yan Y., Kim D., Cameron S., and Parr R. The 1998 OGI-FONIX broadcast news transcription system. In *Proceedings, Broadcast News Transcription and Understanding Workshop* (1999), n.p.
- Liu C., Wu X., and Yan Y. High accuracy acoustic modeling using two-level decision-tree based state-tying. In *EUROSPEECH'99* (1999), vol. 2, pp. 1703-1706.
- Yan Y., Wu X., Schalkwyk J., and Cole R. Development of CSLU LVCSR: the 1997 DARPA HUB4 evaluation system. In *Proceedings, Broadcast News Transcription and Understanding Workshop* (1998), n.p.