ASR Dependent Techniques for Speaker Recognition

by

Alex S. Park

B.Sc., Massachusetts Institute of Technology (2001)

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of

Master of Engineering in Electrical Engineering and Computer Science

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

May 2002

Author ....
Department of Electrical Engineering and Computer Science
May 10, 2002

Certified by ....
Timothy J. Hazen
Research Scientist
Thesis Supervisor

Accepted by ....
Arthur C. Smith
Chairman, Department Committee on Graduate Theses

ASR Dependent Techniques for Speaker Recognition

by

Alex S. Park

Submitted to the Department of Electrical Engineering and Computer Science on May 10, 2002, in partial fulfillment of the requirements for the degree of

Master of Engineering in Electrical Engineering and Computer Science

Abstract

This thesis is concerned with improving the performance of speaker recognition systems in three areas: speaker modeling, verification score computation, and feature extraction in telephone quality speech.

We first seek to improve upon traditional modeling approaches for speaker recognition, which are based on Gaussian Mixture Models (GMMs) trained globally over all speech from a given speaker. We describe methods for performing speaker identification which utilize domain dependent automatic speech recognition (ASR) to provide a phonetic segmentation of the test utterance. When evaluated on YOHO, a standard speaker recognition corpus, several of these approaches were able outperform all previously published results for the closed-set speaker identification task. On a more difficult conversational speech task, we were able to use a combination of classifiers to reduce identification error rates on single test utterances. Over multiple utterances, the ASR dependent approaches performed significantly better than the ASR independent methods. Using an approach we call speaker adaptive modeling for speaker identification, we were able to reduce speaker identification error rates by 39% over a baseline GMM approach when observing five test utterances per speaker.

In the area of score computation for speaker verification, we describe a novel method of combining multiple verification metrics, which is based on confidence scoring techniques for speech recognition. This approach was compared against a baseline which used logarithmic likelihood ratios between target and background scores.

Finally, we describe a set of experiments using TIMIT and NTIMIT which compare the speaker distinguishing capabilities of three different front end feature sets: mel-frequency cepstral coefficients (MFCC), formant locations, and fundamental frequency (F_0). Under matched training and testing conditions, the MFCC feature sets had perfect identification accuracy on TIMIT, but significantly worse performance on NTIMIT. By training and testing under mismatched conditions, we were able to determine the reliability of extraction for individual features on telephone quality speech. Of the alternative features used, we found that F_0 had the highest speaker distinguishability, and was also the most reliably extracted feature under mismatched conditions.

Thesis Supervisor: Timothy J. Hazen
Title: Research Scientist

Acknowledgments

First, I would like to thank my thesis advisor, T.J. Hazen, who has been a patient and knowledgeable mentor throughout the course of this research. Over the last year and a half, his insightful comments and explanations have taught me a great deal about speech and research in general. I would also like to thank my academic advisor, Jim Glass, who has been an invaluable source of advice in matters large and small, from help in finding a research assistantship, to help in finding a dentist.

I am grateful to Victor Zue and the research staff of the Spoken Language Systems group for providing a supportive atmosphere for research and learning. I am also grateful to the students of SLS for many informative and entertaining discussions on and off the topic of speech. Particular thanks go to Han and Jon for their help with many technical issues, and to Ed and Vlad for sharing many painfully fun nights of Tae Kwon Do practice.

I would also like to thank friends and family outside of SLS: Rob and Fred, for their perspectives on academic life; Nori and Jolie, for a fun year at the Biscuit factory; my uncle and his family, for providing a home away from home; my brothers, for always reminding me of our great times together in Canada; and especially Karyn, for always being a steady source of support and encouragement.

Finally, I am most grateful to my parents, whose unwavering love and support have been a source of motivation and inspiration throughout my life.

This research was supported by an industrial consortium supporting the MIT Oxygen Alliance.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The focus of this work is to investigate and improve upon currently used techniques for modeling, scoring, and front-end signal processing in the field of automatic speaker recognition. Speaker recognition is the task of determining a person's identity by his/her voice. This task is also known as voice recognition, a term often confused with speech recognition. Despite this confusion, these two terms refer to complementary problems; while the goal of speech recognition is to determine what words are spoken irrespective of the speaker, the goal of voice recognition is to determine the speaker's identity irrespective of the words spoken.

In mainstream media and science fiction, automatic speaker recognition has been depicted as being as simple as reducing a person's spoken utterance into a characteristic "voiceprint" which can later be used for identification. Despite this simplistic portrayal, determining the unique underlying properties of an individual's voice has proven difficult, and a high-performance speaker recognition system has yet to be built.

1.1 Background

Speaker recognition is specified as one of two related tasks: identification and verification. Speaker identification is the task of identifying a speaker from a set of previously enrolled speakers given an input speech utterance. Potential applications for speaker identification include meeting transcription and indexing, and voice mail summarization. A high level block diagram of a typical speaker identification system is illustrated in Figure 1-1.

A closely related task to speaker identification is speaker verification, which takes a purported identity as well as a speech utterance as inputs. Based on these inputs, the system can either accept the speaker as matching the reference identity, or reject the speaker as an imposter. Speaker verification is one of many biometric devices that are being explored for use in security applications. A high level block diagram of a typical speaker verification system is shown in Figure 1-2.

An additional task distinction is the notion of text dependence. Text dependent systems assume prior knowledge of the linguistic content of the input utterance. This

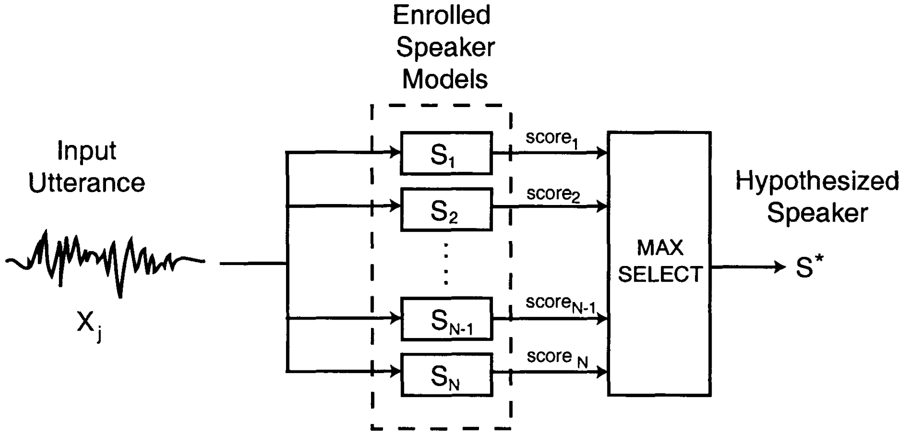

Figure 1-1: Block diagram of typical speaker identification system. The input utterance is scored against enrolled speaker models, and the highest scoring speaker is chosen.

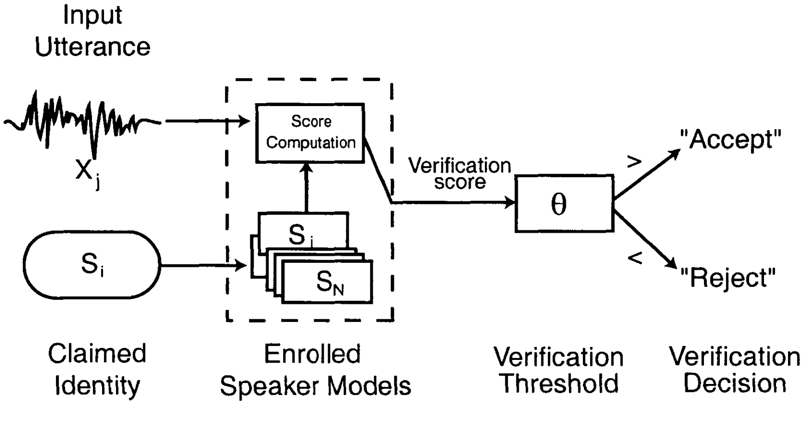

Figure 1-2: Block diagram of typical speaker verification system. A verification score is computed from the input utterance and claimed identity. The verification decision is then made by comparing the score against a predetermined verification threshold.

knowledge is usually obtained by constraining the utterance to a fixed phrase (e.g. digit string). Text independent systems, on the other hand, assume no knowledge of what the speaker is saying, and typically exhibit poorer performance when compared to their text dependent counterparts. In spite of this, text independent systems are considered more useful, as they are more flexible and less dependent on user cooperation.

1.2 Previous Work

There are two main sources of variation among speakers: idiolectal, or high-level, characteristics, and physiological, or low-level, characteristics. Physiological variations arise due to factors such as differences in vocal tract length and shape, and intrinsic differences in glottal movement during speech production. Although humans use these acoustic level cues to some extent when performing speaker recognition, they primarily rely on higher-level cues such as intonation, speaking rate, accent, dialect, and word usage [6]. Doddington [7] and Andrews [1] have recently demonstrated that idiolectal and stylistic differences between speakers can be useful for performing speaker recognition in conversational speech. However, since the usefulness of speech as a biometrically reliable measure is predicated on the difficulty of voluntarily changing factors of speech production, research has generally focused on approaches that are based on low-level acoustic features.

Recent research in the field of speaker recognition can be roughly classified into three areas:

- Feature selection. What is an optimal front-end feature set which provides the greatest separability between speakers? In particular, what feature set remains undistorted and easy to extract in a noisy channel or in otherwise adverse conditions?

- Modeling. How should feature vectors from a particular speaker be used to accurately train a general model for that speaker? This is a difficult issue for text-independent tasks.

- Scoring and Classification. In performing speaker verification, what is an appropriate way of using scores generated by speaker models to reliably determine whether a speaker "closely" matches his/her purported identity?

With a few notable exceptions [17, 20], research in the field of feature extraction and front end processing has been relatively limited. Instead, most current systems use mel-scale cepstral features (MFCCs), and much of the recent research in speaker recognition has focused on modeling and scoring issues.

The use of Gaussian Mixture Models (GMMs) for text-independent speaker modeling was first proposed by Reynolds [24], and has since become one of the most widely used and studied techniques for text-independent speaker recognition. This approach models each speaker using a weighted sum of Gaussian density functions which are trained globally on cepstral feature vectors extracted from all speech for that speaker. Recent variations of this method have included the use of phonetically structured GMMs [10], and the so-called "multi-grained" GMM approach investigated by Chaudhari *et al.* [4]. These techniques are described more fully in Chapter 3.

An alternative approach to global speaker modeling is the use of class-based speaker models. The benefit of using multiple models for each speaker is that the acoustic variability of different phonetic events can be modeled separately. However, this framework also requires a reliable method of assigning features from the test utterance into classes. In [25], for example, Sarma used a segment based speech recognizer to model speakers using broad phonetic manner classes. Similar work was also done by Petrovska-Delacrétaz in [21]. In this work, the use of unsupervised speaker class models was also studied, but results indicated that class-based models overall had worse performance compared to global models due to insufficient training data per class.

For speaker verification, a major obstacle is the determination of a suitable metric for gauging the closeness of claimant speakers to the target speaker. The prevailing approach for most systems is to score the input utterance against both the target speaker model and a universal background model which is a combination of all speaker models. The logarithmic likelihood ratio of these two scores is then used as a verification score which is compared against a pre-determined threshold. An alternative approach to this problem has been presented by Thyes *et al.* [28], who used the concept of "eigenvoices" for verification. In that work, principal components analysis was used on feature vectors from a large set of training speakers to determine a subspace of feature dimensions, or "eigenspace", which captured the most variability between the speakers in the training data. Claimant utterances were projected into the eigenspace and were verified or rejected based on their distance from the target speaker vector, or "eigenvoice". The eigenvoices approach has been shown to perform well in cases where there is a large amount of data to train the eigenspace, and has the benefit that performance is good even for sparse enrollment data.

1.3 Goals and Motivation

Although the goal of text independent speaker recognition has led to an increased focus on global speaker modeling, it is well known that some phones have better speaker distinguishing capabilities than others [9]. Global speaker modeling techniques like the GMM approach are not able to take optimal advantage of the acoustic differences of diverse phonetic events. Conversely, phone level speaker modeling techniques exhibit poor performance due to insufficient training data at the phone level [21]. This work strives to address both of these modeling issues by using automatic speech recognition together with techniques borrowed from speaker adaptation.

A secondary goal of this thesis is to apply a new method of score combination to the task of speaker verification. While many systems attempt to find a single optimal verification metric, a fusion of metrics can potentially give better performance. We explore possible benefits of this approach using confidence scoring techniques.

A final goal of this thesis is to investigate the use of alternative features to MFCCs for noise robustness issues. To this end, we explore the relative speaker distinguishing capabilities of formant locations and fundamental frequency measures in mismatched noise conditions.

1.4 Overview

The rest of the thesis is organized in the following manner. Chapter 2 describes the system, signal processing tools, and corpora used for this work. A detailed description of baseline and ASR dependent approaches for speaker modeling is provided in Chapter 3, and performance and analyses of these systems on closed set speaker identification tasks are presented in Chapter 4. Chapter 5 describes a novel framework for performing speaker verification using confidence scoring methods. Chapter 6 discusses the speaker distinguishing capabilities of alternative features, such as formants and fundamental frequency. Finally, Chapter 7 closes with concluding remarks and provides some directions for future work.

Chapter 2

Corpora and System Components

This chapter describes the corpora used for evaluation, and the system components used for building, testing, and experimenting with various aspects of the speaker recognition system.

2.1 Speaker Recognition Corpora

Over the course of this work, four corpora were used. The YOHO and MERCURY data sets were used to evaluate the performance of modeling approaches for speaker identification. An augmented version of MERCURY was also the primary corpus used for testing verification scoring methods. Finally, for evaluating the identification performance of different features, TIMIT and NTIMIT were used. The following sections give a description of the corpora, and the information is also summarized in Table 2-2.

2.1.1 YOHO

YOHO is a speaker recognition corpus distributed by the Linguistic Data Consortium [3]. As a standard evaluation corpus, many speaker recognition papers publish performance results on YOHO, making it useful for comparing the performance of different systems. The corpus consists of speech recorded from 106 male speakers and 32 females and attempts to simulate a verification scenario taking place in an office environment. Each utterance is a fixed-length combination-lock type phrase (e.g. "twenty-four, thirty-five, eighty-two"). During enrollment, each speaker was prompted with the same set of 96 digit phrases divided into four sessions of 24 utterances each. For verification, each speaker had ten sessions of four utterances each. Unlike enrollment, the verification utterances were not constrained to be the same across all speakers. The speech was recorded at a sampling rate of 8 kHz using a high-quality telephone handset, but was not actually passed through a telephone channel.

2.1.2 Mercury

The Mercury corpus is a speaker-labelled subset of data collected from the Mercury air travel information system [26]. Mercury differs from YOHO in a number of ways. The type of speech is spontaneous, rather than read, and the 2300 word vocabulary includes city names, dates, and function words in addition to digits. Because Mercury is a telephone-based system, the data in the corpus also has additional adverse conditions such as variable handsets and telephone channels, variable length utterances and out of vocabulary words. An example of a Mercury user interaction is shown in Table 2-1.

For the closed set identification task, a set of 38 speakers was used. The training and test sets consisted of approximately 50 and 100 utterances per speaker, respectively. In addition, a development set was created from remaining data left over after training and test sets had been formed. The development set contained between 15 and 200 utterances per speaker.

For the verification task, an additional set of utterances taken from 80 imposter speakers was also created. The test set and development set were then augmented with these utterances by placing data from half of the new speakers in the test set, and the rest in the development set.

2.1.3 TIMIT & NTIMIT

TIMIT is a continuous speech corpus developed as a joint effort between the Massachusetts Institute of Technology, Stanford Research Institute, and Texas Instruments. The speech data in TIMIT was recorded using a high quality microphone in a quiet recording booth. Although the full corpus consists of sentences read by 630 different speakers, the evaluations performed in this thesis used only a 168 speaker subset of the corpus. Each speaker read a set of ten sentences: two standard (SA) sentences, five phonemically rich (SI) sentences, and three unique (SX) sentences. The SA sentences were the same across all speakers, and the SX sentences were unique to each speaker. The SI sentences were drawn from a pool of constructed phrases, and were therefore neither the same across all speakers nor unique to any particular speaker. For the speaker identification task, training was done on each speaker's eight non-SA utterances, and evaluation was performed using the two SA utterances.

NTIMIT is a speech corpus derived from TIMIT, and was developed in order to examine telephone channel effects on speech recognition performance for high quality data. NTIMIT data was obtained by playing TIMIT recordings out of a speaker into a telephone handset. The data was then passed through a land-line telephone channel and recorded. The training and test set partitioning was identical to that for TIMIT.

Mercury: Welcome to MERCURY, the MIT air travel planning system. I can help plan your air travel itinerary. Please tell me your user name. If you do not have a user name, please say new user.
User: New user.
MERCURY: Welcome. You have successfully logged onto MERCURY as a new user. How may I help you?
User: I would like to fly from Boston to London
MERCURY: Flights from Boston to London. What date will you be travelling?
User: Next Friday in the evening
MERCURY: Okay. flights from Boston to London departing between 5:00 pm and midnight on June 25. Can you provide a departure or arrival time? The earliest nonstop flight from Boston to London departing between 5:00 pm and midnight on June 25 leaves at 5:30 p.m. and arrives at 4:55 a.m. on British Airways. You arrive one day later.
User: Is there a United flight?
MERCURY: Okay. United flights from Boston to London departing between 5:00 pm and midnight on June 25. One moment please... I have a nonstop United flight leaving at 6:40 p.m. and arriving at 6:25 a.m.?
User: Book that flight
MERCURY: United flight 998 has been added to your itinerary. You depart Boston at 6:40 pm and arrive in London Heathrow at 6:25 am. You arrive one day later. Shall I price your itinerary?
User: Yes.
MERCURY: The price for your itinerary is $2065.40. Is there anything else I can do for you?
User: No thank you. Good bye.
MERCURY: Thanks for using MERCURY. Have a nice day!

Table 2-1: Example of a conversation with MERCURY.

Property	YOHO	MERCURY	TIMIT/NTIMIT
Number of speakers	138	38	168
Male,Female	106M,32F	18M, 20F	112M, 56F
Type of speech	Read	Spontaneous	Read
Lexical content	Digit strings	Conversational queries	Short sentences
Environment	Quiet office	Variable	Recording booth
Microphone	Telephone handset	Variable handset	Noise cancelling/ Telephone handset
Channel	8 kHz	8 kHz Variable telephone channel	16 kHz/8 kHz
Training Data	96 utterances	$\leq$ 50 utterances	8 sentences

Table 2-2: Summary of corpus properties

2.2 The SUMMIT Speech Recognition System

In this section, we give a brief overview of the components of the SUMMIT speech recognition system. The speaker recognition system built for this thesis used the existing infrastructure of SUMMIT. In addition, the ASR dependent modeling approaches described in the following chapter used word and phone hypotheses generated by SUMMIT. For a more detailed description of the system, see [13].

2.2.1 Segment-based Recognition

Unlike many modern frame-based speech recognition systems, SUMMIT uses variable-length segments as the underlying temporal units for an input waveforms. Frame-based recognizers compute feature vectors (*e.g.*, MFCC's) at regularly spaced time intervals (frames), and then use those frame-level feature vectors to model acoustic events. Although SUMMIT initially computes features at regular time intervals, it then hypothesizes acoustic landmarks at instants where large acoustic differences in the frame-level feature vectors indicate that a boundary or transition may occur. These landmarks then specify a network of possible segmentations for the utterance, each one associated with a different feature vector sequence.

2.2.2 Mathematical Framework

In performing continuous speech recognition, the ultimate goal of any system is to determine the most likely word sequence, $\vec{w}^* = \{w_1, w_2, \ldots, w_M\}$, which may have been produced by a sequence of input feature vectors, $A = \{\vec{a}_1, \vec{a}_2, \ldots, \vec{a}_N\}$. This can be expressed as

$$\vec{w}^* = \arg\max_{\vec{w}} P(\vec{w}|A), \tag{2.1}$$

With a segment-based recognizer, there are multiple segmentation hypotheses, s, each one associated with a different sequence of acoustic feature vectors, A_s. Additionally, most words or word sequences, $\vec{w}$, have multiple realizations as sequences of subword units, $\vec{u}$. To reduce computation, SUMMIT implicitly assumes that there exists an optimal segmentation, s^*, and subword unit sequence, $\vec{u}^*$, for any given word sequence. Equation 2.1 therefore becomes

$$\{\vec{w}^*, \vec{u}^*, s^*\} = \arg\max_{\vec{w},\vec{u},s} P(\vec{w}, \vec{u}, s, |A) \tag{2.2}$$

Applying Bayes' rule, Equation 2.2 becomes

$$\{\vec{w}^*, \vec{u}^*, s^*\} = \arg\max_{\vec{w},\vec{u},s} P(A|\vec{w}, \vec{u}, s)P(s|\vec{u}, \vec{w})P(\vec{u}|\vec{w})P(\vec{w}) \tag{2.3}$$

In Equation 2.3, the estimation of the right-hand components is partitioned as follows

- $P(A|\vec{w}, \vec{u}, s)$ - Acoustic model. This component represents the likelihood of observing a set of acoustic features given a particular class. In practice, SUMMIT assumes that the acoustic features of each landmark are dependent only on the subword unit at that landmark. Thus, this likelihood is calculated as $P(A|\vec{w}, \vec{u}, s) = \prod_{i=1} P(\vec{a}_i|l_i)$, where l_i is the ith landmark. If l_i occurs within a segment, then $P(\vec{a}_i|l_i) = P(\vec{a}_i|u_i)$, and if l_i occurs at a boundary between two segments, then $P(\vec{a}_i|l_i) = P(\vec{a}_i|u_i, u_{i-1})$.

- $P(s|\vec{w}, \vec{u})$ - Duration model. This component represents the likelihood of observing a particular segmentation given a particular word and subword sequence. In the implementation of SUMMIT used in this work, duration models are not incorporated, so this value is constant.

- $P(\vec{u}|\vec{w})$ - Lexical/Pronunciation model. This component represents the likelihood of a particular subword sequence for a given word sequence. SUMMIT uses a base set of pronunciations for each word in the vocabulary [31]. Phonological rules are then applied to these baseforms in order to generate alternate pronunciations that account for common phenomena observed in continuous speech [16].

- $P(\vec{w})$ - Language model. This component represents the *a priori* probability of observing a particular word sequence. SUMMIT uses a smoothed n-gram language model which associates a probability with every n-word sequence. These probabilities are trained on a large corpus of domain-dependent sentences.

2.2.3 Finite State Transducers

In order to find the optimal word sequence for Equation 2.3, SUMMIT models the search space using a weighted finite-state transducer (FST), R, which is specified as the composition of four smaller FSTs

$$R = (S \circ A) \circ (C \circ P \circ L \circ G) \tag{2.4}$$

where:

- $(S \circ A)$ represents a segment graph which maps acoustic features to phonetic boundary labels with some associated probability score;
- C represents a mapping from context-dependent boundary labels to context-independent phone labels;
- P is a transducer which applies phonological rules to map alternate phone realizations to phoneme sequences;
- L represents the phonemic lexicon which maps phoneme sequences to words in the recognizer vocabulary; and
- G represents the language model which maps words to word sequences with an associated probability.

The composition of these four FSTs, therefore, takes acoustic feature vectors as input, and maps them to word sequences with some output probability. The best path search through R is performed using a forward Viterbi beam search [23], followed by a backward A^* beam search [30]. The final result of the search is an N-best list of the most probable word sequences corresponding to the input waveform.

2.3 Signal Processing Tools

For the non-cepstral features examined in Chapter 6, such as fundamental frequency (F_0) and formant locations, values were computed offline using tools from the Entropic Signal Processing System (ESPS). ESPS is a suite of speech signal processing tools available for UNIX. Measurements for F_0 and voicing decisions were obtained using the `get_f0` command, and formant locations were obtained using the `formant` command. More detail about the actual algorithms used in each of these tools is given in Chapter 6.

Chapter 3

Modeling Approaches for Identification

In this chapter, we describe various modeling approaches used for speaker recognition. In particular, several modeling techniques are illustrated for the task of closed-set speaker identification. We distinguish here between traditional text independent approaches which we classify as ASR independent, and ASR dependent approaches which make use of automatic speech recognition during speaker identification. The first section discusses the general theory of Gaussian mixture models (GMM), upon which each the modeling techniques are based. We then describe two ASR independent approaches followed by two ASR dependent approaches. The final section details a two stage implementation strategy which allows combination of multiple classifiers.

3.1 Gaussian Mixture Models

3.1.1 Background

The most widespread paradigm for statistical acoustic modeling in both speech recognition and speaker recognition involves the use of class-conditional Gaussian mixture models. With this approach, the probability density function for a feature vector, $\vec{z}$, is a weighted sum, or *mixture*, of K class-conditional Gaussian distributions. For a given class, c, the probability of observing $\vec{z}$ is given by

$$p(\vec{z}|c) = \sum_{k=1}^{K} w_{c,k} \mathcal{N}(\vec{z}; \vec{\mu}_{c,k}, \Sigma_{c,k}) \tag{3.1}$$

For speech recognition, the class c is usually taken to be a lexical, phonetic, or subphonetic unit. For speaker recognition, each class is usually taken to represent a different speaker. In Equation 3.1, $w_{c,k}$, $\vec{\mu}_{c,k}$, $\Sigma_{c,k}$ are the mixture weight, mean, and covariance matrix, respectively, for the i-th component, which has a Gaussian distribution given by

$$\mathcal{N}(\vec{z};\vec{\mu},\Sigma) = \frac{1}{\sqrt{(2\pi)^n|\Sigma|}} e^{\frac{1}{2}(\vec{z}-\vec{\mu})'\Sigma^{-1}(\vec{z}-\vec{\mu})} \tag{3.2}$$

Although Σ can, in principle, represent a full covariance matrix, in practice, most implementations use diagonal covariance matrices to reduce computation.

In order to determine the parameters for the class conditional densities, a semi-supervised training approach is used. Assignment of training vectors to classes is done with prior knowledge of the correct classes, but the training of the class models themselves is performed in an unsupervised fashion. Given a set of training vectors which are known to be in the class, an initial set of means is estimated using the K-means clustering [23]. The mixture weights, means, and covariances are then iteratively trained using the well-known expectation-maximization (EM) algorithm [5].

3.1.2 Training

Our baseline system was based closely upon Reynolds' GMM approach [24]. The class conditional probability densities for the observed feature vectors was the same as in Equation 3.1, with each class, C, representing a different speaker, S_i.

$$p(\vec{z}|S_i) = \sum_{k=1}^{K} w_{S_i,k}\mathcal{N}(\vec{z};\vec{\mu}_{S_i,k},\Sigma_{S_i,k}), \quad K \leq 64 \tag{3.3}$$

For training, enrollment data was separated into speech and non-speech using a time aligned forced transcription produced by SUMMIT. The feature vectors from the speech segments, $\vec{z}$, were derived from 14-dimension mean normalized MFCC vectors. For each input waveform, 98-dimension vectors were created at 10 millisecond intervals by concatenating averages of MFCCs from six different segments surrounding the current frame. These additional segment level measurements, which are illustrated in Figure 3-1, were used to capture dynamic information surrounding the current frame. Principal components analysis was then used to reduce the dimensionality of these feature vectors to 50 dimensions [13].

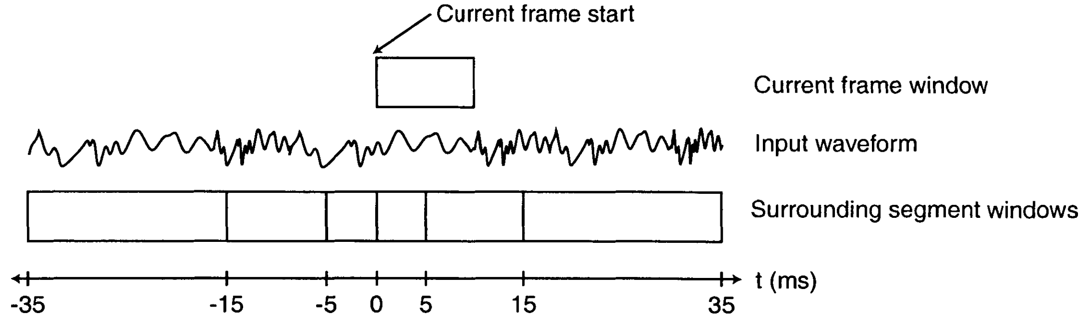

Figure 3-1: Segment windows for computing frame level feature vectors.

3.1.3 Identification

During recognition, we make use of a speaker FST, which constrains the possible state transitions for each input waveform, and also maps a sequence of input frame-level feature vectors to output speaker labels. An example speaker FST is illustrated in Figure 3-2. In Figure 3-2, the input labels "S_i" and "-" refer to speaker and silence models, respectively. The input-output mapping in the speaker FST enforces the constraint that each utterance will only produce input feature vectors that are classified as either speech or non-speech (-), and that all speech samples from a single utterance will be produced by the same speaker. By using this FST together with speaker acoustic models in place of R in Equation 2.4, we are able to use SUMMIT to perform speaker identification by generating a speaker N-best list.

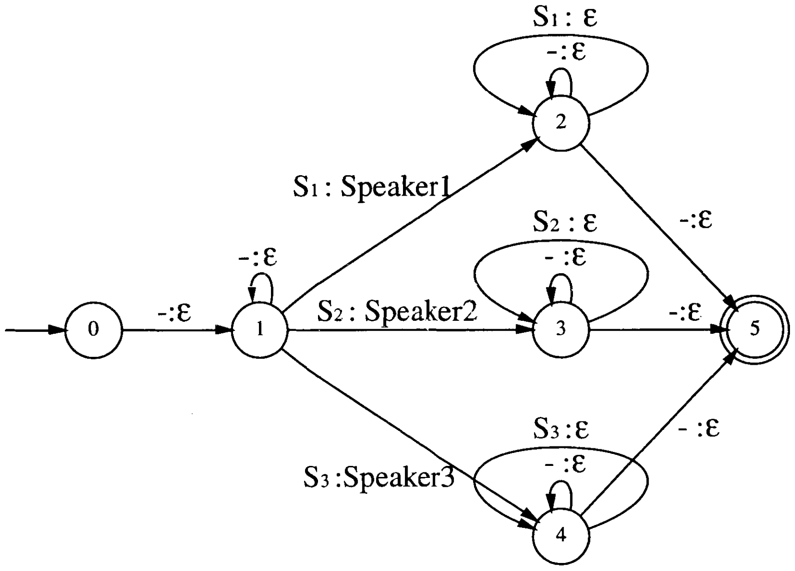

Figure 3-2: Finite-state transducer mapping input frame feature vectors to output speaker labels

3.2 Alternative Modeling Approaches

3.2.1 Phonetically Structured GMMs

A recent variant of the traditional GMM approach is the so-called "phonetically-structured" GMM method which has been proposed by Faltlhauser, *et al.* [10]. During training, forced transcription of the enrollment data is used to separate frame-level features into broad phonetic manner classes. These classes are shown in Table 3-1. For each speaker, eight separate GMMs are then trained, one for each phonetic class. After training, these smaller GMMs are then combined into a single larger model using a globally determined weighting. We can contrast this approach with the baseline by observing the structure of the speaker models

$$p(\vec{z}|S_i) = \sum_{j=1}^{8} a_j \sum_{k=1}^{K_j} w_{S_i,k,C_j}\mathcal{N}(\vec{z};\vec{\mu}_{S_i,k,C_j},\Sigma_{S_i,k,C_j}) \tag{3.4}$$

In Equation 3.4, C_j is the phone class, and a_j and K_j are the weight and number of mixtures for C_j, respectively. The motivation for constructing the global speaker GMM in this fashion is that, this method is less sensitive to phonetic biases present in the enrollment data of individual speakers. During identification, all speech frames from the test utterance are scored against the combined model, as illustrated in Figure 3-3.

Class	Example Phone Label
diphthong	[ɑ^{y}]
vowel	[æ]
closure	[g$^{\square}$]
nasal	[n]
burst	[k^{h}]
strong fricative	[š]
weak fricative	[f]
liquid	[l]

Table 3-1: Phone classes used for phonetically structured GMMs and phonetic classing approaches.

3.2.2 Phonetic Classing

The following two approaches, which we term ASR-dependent, require a speech recognition engine, such as SUMMIT, to generate a hypothesized phonetic segmentation of the test utterance. The generation of this hypothesis is described in Section 3.3.

The use of separate phonetic manner classes for speaker modeling was studied previously by Sarma [25]. This technique is similar to the use of phonetically structured GMMs in that training is identical. Phonetic class GMMs are trained for each speaker, but instead of being combined into a single speaker model, the individual classes are retained. Each speaker is then represented by a set of class models, instead of a global model as in Equation 3.4.

$$p(\vec{z}|S_i) = \begin{cases} p(\vec{z}|S_i, c_1) = \sum_{k=1}^{K_1} w_{S_i,k,c_1}\mathcal{N}(\vec{z};\vec{\mu}_{S_i,k,c_1},\Sigma_{S_i,k,c_1}) & \text{if } \vec{z} \in c_1; \\ \vdots & \\ p(\vec{z}|S_i, c_8) = \sum_{k=1}^{K_8} w_{S_i,k,c_8}\mathcal{N}(\vec{z};\vec{\mu}_{S_i,k,c_8},\Sigma_{S_i,k,c_8}) & \text{if } \vec{z} \in c_8. \end{cases} \tag{3.5}$$

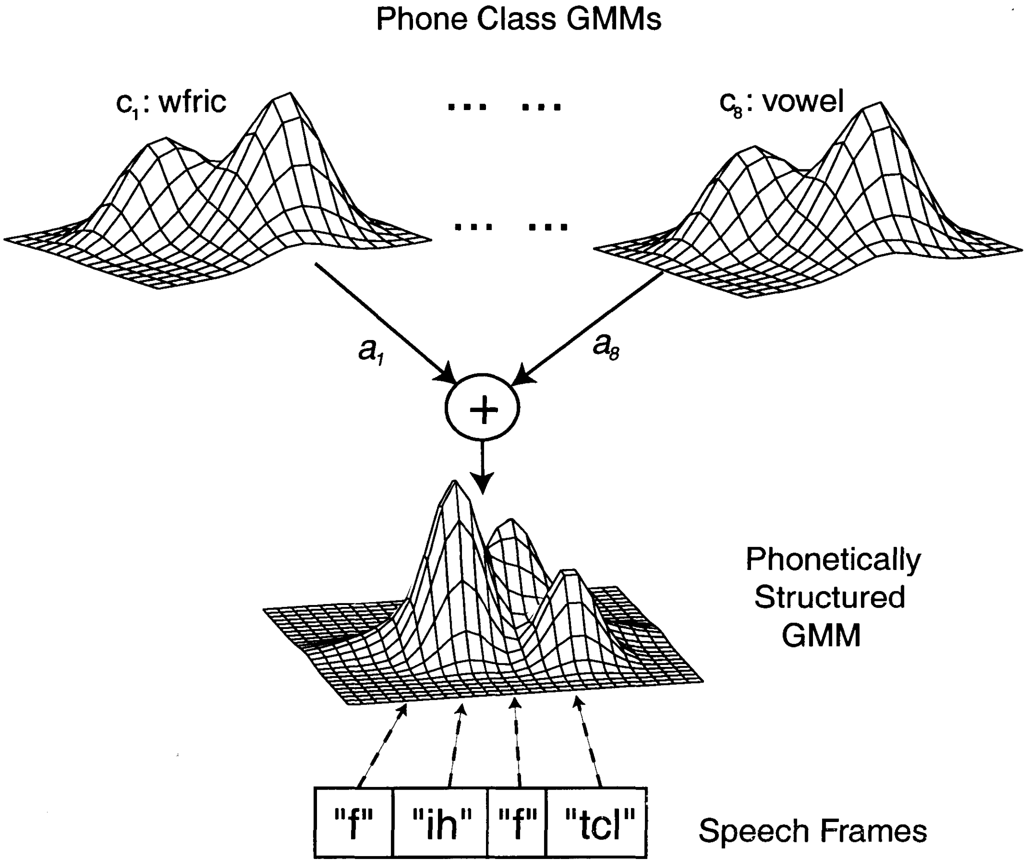

Figure 3-3: Phonetically Structured GMM scoring framework

During identification, each test vector is assigned to a phonetic class using the phonetic segmentation hypothesis provided by the speech recognizer. The appropriate phone class model is then used to score the vector. This scoring procedure is illustrated in Figure 3-4.

Since test vectors are scored directly against the class-level GMMs, this approach is similar to the "multigrained" method proposed by Chaudhari *et al.* [4]. However, by using the phone class assignment provided by the speech recognizer, this approach eliminates the need to score against every model in the speaker's library, as is required by the multigrained method.

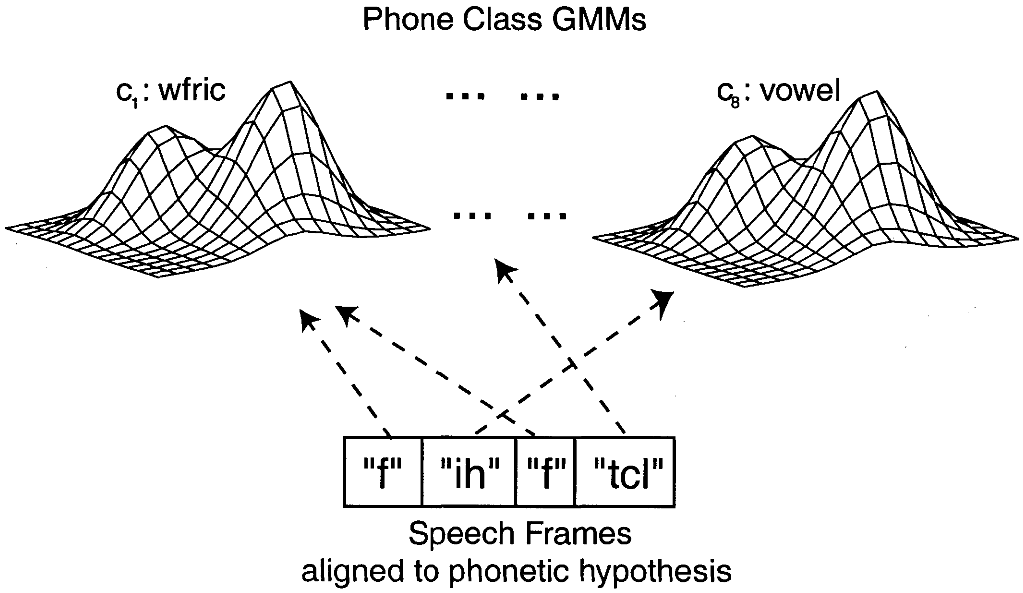

Figure 3-4: Phonetic Class scoring framework

3.2.3 Speaker Adaptive Scoring

The previous two approaches attempt to improve upon the baseline GMM approach by using broad phonetic class models which are more refined than the global GMMs. At a further level of granularity, models can be built for specific phonetic events. Unfortunately, the enrollment data sets for each speaker in typical speaker ID tasks are usually not large enough to build robust speaker dependent phonetic-level models. To compensate for this problem, we can draw upon techniques used in the field of speaker adaptation. This allows us to build models that learn the characteristics of a phone for a given speaker when sufficient training data is available, and rely more on general speaker independent models in instances of sparse training data.

In this approach, speaker dependent segment-based speech recognizers are trained for each speaker. That is, rather than training *class* level speaker models, a separate *phone* level speaker model is trained for each phonetic model in the speaker independent speech recognizer. During identification, the hypothesized phonetic segmentation produced by the speaker independent speech recognizer is used to generate the best path speaker dependent score, which is then interpolated with the recognizer's speaker independent score. This method approximates the MAP strategy for speaker adaptation [12]. Mathematically, if the word recognition hypothesis assigns each test vector $\vec{z}$ to a phone j, then the likelihood score for $\vec{z}$ given speaker S_i is given by

$$p(\vec{z}|S_i) = \lambda_{i,j} p(\vec{z}|M_{i,j}) + (1 - \lambda_{i,j}) p(\vec{z}|M_j) \tag{3.6}$$

where $M_{i,j}$, M_j are the speaker dependent and speaker independent models for phone j, and $\lambda_{i,j}$ is an interpolation factor given by

$$\lambda_{i,j} = \begin{cases} \frac{n_{i,j}}{n_{i,j}+\tau} & \text{if } n_{i,j} > 1; \\ 0 & \text{if } n_{i,j} = 0, 1. \end{cases} \tag{3.7}$$

In Equation 3.7, $n_{i,j}$ is the number of training tokens of phone j for speaker i, and τ is an empirically determined tuning parameter that is the same across all speakers and phones. For instances when $n_{i,j}$ is equal to 0 or 1, the corresponding speaker dependent Gaussian, $M_{i,j}$, cannot be trained, and the score is computed using only the speaker independent Gaussian.

3.3 Two Stage Scoring

In order to make use of speech recognition output during speaker identification, we utilize a two-stage method to calculate speaker scores. This framework is illustrated in Figure 3-5. In the first stage, the test utterance is passed in parallel through a speech recognition module and a GMM speaker ID module, which is implemented using the baseline approach. The speech recognition module produces a time-aligned phonetic hypothesis, while the GMM speaker ID module produces an N-best list of hypothesized speakers. These results are then passed to the next stage, where a second classifier rescores each speaker in the N-best list using one of the refined techniques described above.

This two-stage scoring method is useful in a number of ways. First, by using the GMM speaker ID module for fast-match, we reduce post-recognition latency by limiting the search space of speakers presented to the second stage. Identification performance is not significantly affected since the probability of N-best exclusion of the target speaker by the GMM module can be made arbitrarily low by increasing N. Furthermore, there is little increase in pre-identification latency for the ASR dependent approaches since the GMM scoring proceeds in parallel with word recognition. Another advantage of this framework is that scores from multiple classifiers can be used and combined in the second stage.

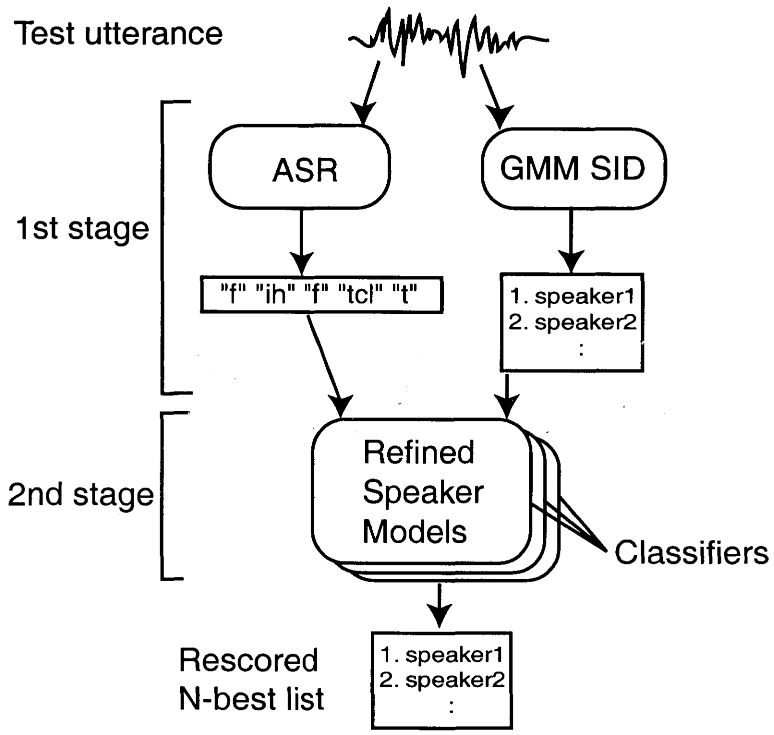

Figure 3-5: Two stage scoring framework allowing simultaneous speech recognition and speaker identification

Chapter 4

Identification Experiments for Modeling

In this chapter, closed set speaker identification results are described for YOHO and MERCURY. We omit evaluation on TIMIT in this chapter because we were able to attain 100% identification accuracy on TIMIT using the baseline GMM approach. Instead, we reserve investigation of TIMIT and NTIMIT for Chapter 6, where we discuss benefits of different front-end features under adverse channel conditions.

4.1 Experimental Conditions

4.1.1 Speech Recognition

For both corpora we used domain dependent implementations of the MIT SUMMIT speech recognizer [13]. On the YOHO data set, the vocabulary and language model were limited to allow only the set of possible numerical combination lock phrases. Figure 4-1 illustrates the domain dependent finite-state acceptor (FSA) used to model the constrained vocabulary allowed in YOHO.

On the MERCURY data set, a recent version of SUMMIT currently deployed for the MERCURY domain was used [26]. This recognizer was considerably more complex than the one used for YOHO, and included a 2300 word vocabulary suited for conversational queries regarding airline travel.

4.1.2 Modeling Parameters

For each of the modeling approaches that we used, performance was dependent on the choice of parameters specified in the Chapter 3. The relevant parameters for each approach were

- Baseline - Number of mixture components, K, from Equation 3.3.
- Phonetically Structured - Number of mixture components per class, K_j, and phone class weights, a_j, from Equation 3.4.

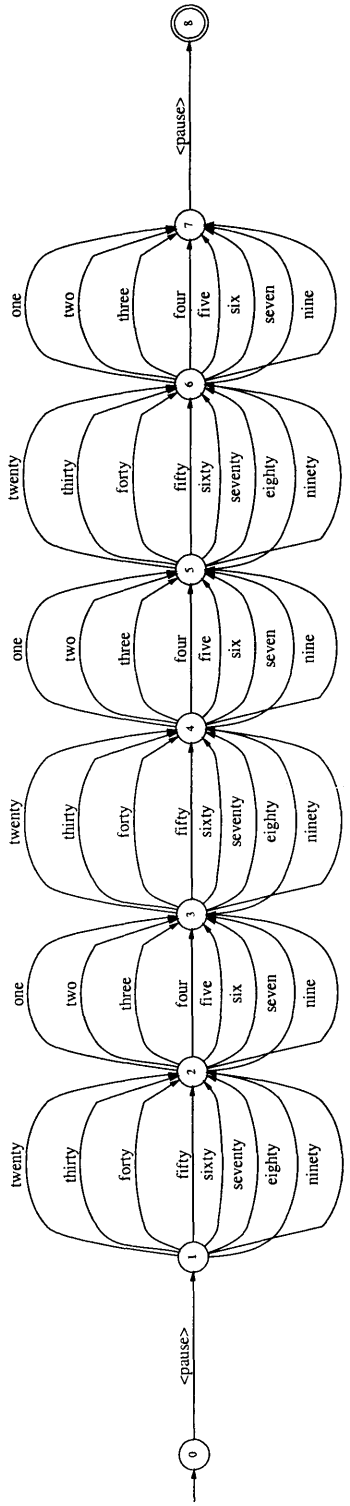

Figure 4-1: Finite state network illustrating constrained grammar for YOHO

- Phonetic Classing - Number of mixture components per class, K_j, from Equation 3.5.
- Speaker Adaptive - Interpolation parameter, τ, from Equation 3.7.

In addition to the model parameters for individual classifiers, weights for combining classifier scores must also be specified.

Determination of Mixture Numbers and Phone Class Weights

We first examined the effect of varying the total number of mixture components used in the global speaker modeling approaches. Initially, our choice of $K = 64$ for the baseline system was motivated by Reynolds' use of 64-mixture GMMs in [24]. We also used $K = 64 \times 8 = 512$ in order to perform a fair comparison of this method against the Phonetically Structured approach with $K_j = 64$. These results are shown in Table 4-1.

	Error Rate (%)	
Method	YOHO	MERCURY
Baseline GMM (K=64)	0.83	22.43
Baseline GMM (K=512)	0.80	22.43

Table 4-1: Comparison of identification error rates for baseline approach with different numbers of mixtures on YOHO and MERCURY data sets

For the Phonetically Structured approach, we also performed similar experiments using $K_j = 8$, and $K_j = 64$. These values corresponded to total mixture counts of $K_{tot} = 64$ and $K_{tot} = 512$, respectively. In addition to varying the number of mixture components, we also used two methods for determining mixture weights

- Unweighted: $a_j = 1/(\# \text{ of classes})$
- Weighted: $a_j = N_j/N_{tot}$
  where N_j and N_{tot} are the number of training tokens for phone class C_j and the total number of training tokens across all speakers, respectively.

From the results in Table 4-1 and Table 4-2, we noted that weighting the phone class GMMs by their prior distributions in the training data resulted in worse performance than the unweighted GMMs. This was likely due to the fact that weighting with priors introduces bias against mixture components from underrepresented phone classes, which negated the main benefits of using the phonetically structured approach. The results also demonstrated that the phonetically structured models outperformed the baseline models when the total number of mixtures was large. The disadvantage of simply increasing K_j is that the size of the global speaker model is increased, therefore requiring more computation to score each test utterance during identification.

		Error Rate (%)	
Method	K_j	YOHO	MERCURY
Weighted	8	0.83	24.17
Unweighted	8	0.74	23.95
Weighted	64	0.33	22.43
Unweighted	64	0.31	21.32

Table 4-2: Comparison of identification error rates for Phonetically Structured approach when varying mixture counts and class weights

For the phonetic classing approach, incorporation of automatic speech recognition reduces the need to "budget" the total number of mixtures among the individual phone class models. During identification, every frame of the test utterance is scored only against the phone class that it is assigned to. Therefore, computational complexity is a function of the number of mixtures per class rather than the total number of mixtures across all the class models for a particular speaker. For the phone classing approach, we set $K_j = 64$.

Determination of Interpolation Parameters for Speaker Adaptive Scoring

For the speaker adaptive approach, we used linear interpolation to combine the speaker dependent and speaker independent scores at the phone level as described in Equations 3.6 and 3.7. In this approach, the interpolation factor for speaker i and phone j was specified by a global parameter, τ, and the number of training tokens from speaker i for phone j, $n_{i,j}$. In order to empirically determine an optimal value for τ, we performed identification experiments on the MERCURY development set. We observed that identification accuracy generally improved as τ was decreased from 100 down to 5. For values between 1 and 5, identification accuracy was not significantly affected. We used these results to set $\tau = 5$ for speaker adaptive scoring on MERCURY and YOHO.

In performing experiments to tune for τ, an interesting result that we noted was that setting $\tau = 0$ yielded high identification error rates on the development set. As described in Section 3.2.3, this τ value corresponded to using only speaker dependent scores for phones with more than one training token, and speaker independent scores otherwise. This result confirmed our earlier suspicion that completely speaker dependent phone-level models would perform poorly due to an insufficient number of training examples for each phone.

Determination of Classifier Combination Weights

A primary benefit of the two-stage scoring framework described in Section 3.3 is the ability to combine scores from multiple classifiers, which can can often mitigate errors made by a single classifier. After an initial speaker N-best list is produced by the

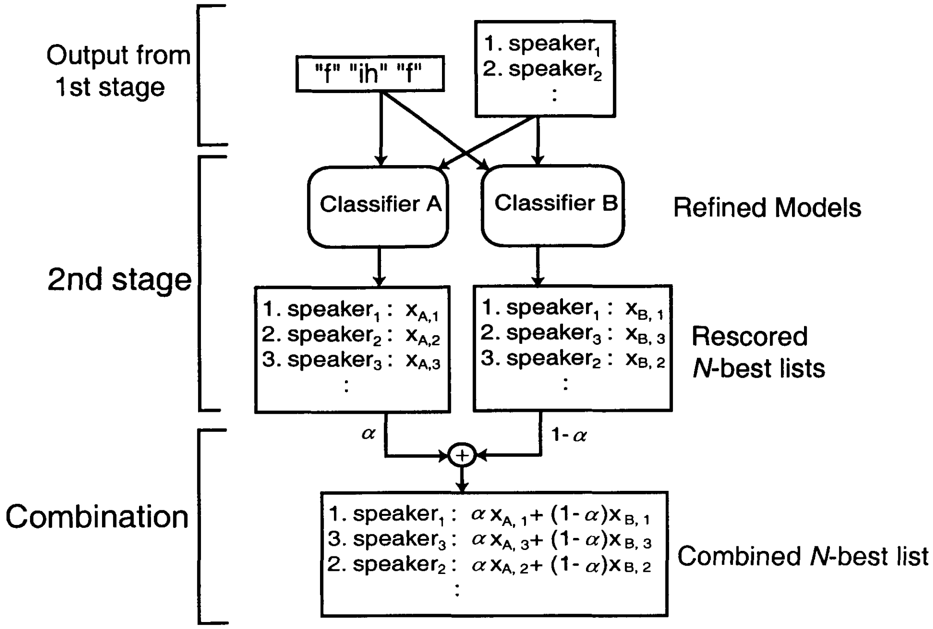
Output from 1st stage
"f" "ih" "f"
1. speaker$_1$
2. speaker$_2$
⋮
2nd stage
Classifier A
Classifier B
Refined Models
1. speaker$_1$: $x_{A,1}$
2. speaker$_2$: $x_{A,2}$
3. speaker$_3$: $x_{A,3}$
1. speaker$_1$: $x_{B,1}$
2. speaker$_3$: $x_{B,3}$
3. speaker$_2$: $x_{B,2}$
Rescored N-best lists
α
1-α
Combination
1. speaker$_1$: $\alpha x_{A,1} + (1-\alpha) x_{B,1}$
3. speaker$_3$: $\alpha x_{A,3} + (1-\alpha) x_{B,3}$
2. speaker$_2$: $\alpha x_{A,2} + (1-\alpha) x_{B,2}$
Combined N-best list

Figure 4-2: Combination of scores from multiple classifiers

GMM classifier in the first stage, a different classifier can be used in the second stage to produce a rescored N-best list using more refined speaker models. If the initial N-best list is passed to multiple classifiers, each one produces its own rescored N-best list. These rescored lists can then be combined and resorted to produce a final N-best list. This process is illustrated in Figure 4-2.

Although it is possible to use any number of classifiers, we only investigated pairwise groupings using linear combination

$$\text{Total score for } S_i = \alpha x_{A,S_i} + (1-\alpha) x_{B,S_i} \qquad (4.1)$$

In Equation 4.1, x_{A,S_i} and x_{B,S_i} are scores for speaker S_i from classifiers A and B, respectively. The combination weights were optimized by varying α between 0 and 1 on the MERCURY development set.

4.2 Comparison of Methods on Single Utterances

In order to investigate the advantages of each of the modeling approaches, we first computed results for the closed set identification task on individual utterances. These results are shown in Table 4-3. When comparing the performance of the different classifiers, we observed that error rates on the YOHO corpus were uniformly low. In particular, we noted that our best results on the YOHO corpus were better than the 0.36% identification error rate obtained by a system developed at Rutgers [3], which is the best reported result that we are aware of for this task. However, with the exception of systems involving the GMM baseline, each of the classifiers produced between 14 and 22 total errors out of 5520 test utterances, making the differences between these approaches statistically insignificant.

On the MERCURY data set, the comparative performance of each system was more evident. Both the phonetically structured GMM system and the phonetic classing system had slight improvements over the baseline, while the speaker adaptive system has a higher error rate than any of the other approaches. Across all systems, we observed that error rates were significantly higher on the MERCURY task than on YOHO, clearly illustrating the increased difficulties associated with spontaneous speech, noise, and variable channel conditions. These factors also led to a higher word error rate for speech recognition on the MERCURY data, which partially explains why the recognition aided systems did not yield improvements over the baseline GMM method as observed with YOHO. However, we saw that by combining the outputs of multiple classifiers, lower overall error rates were achieved on both corpora.

	Parameters				*Error Rate (%)*	
Method	K	K_j	τ	α	YOHO	MERCURY
Baseline GMM	64				0.83	22.4
Phonetically Structured GMM (PS)		64			0.31	21.3
Phone Classing (PC)		64			0.40	21.6
Speaker Adaptive (SA)			5		0.31	27.8
Multiple Classifiers (GMM+SA)	64		5	0.45	0.53	19.0
Multiple Classifiers (PS+SA)		64	5	0.33	0.25	18.3
Multiple Classifiers (PC+SA)		64	5	0.30	0.25	18.5

Table 4-3: Comparison of identification error rates for each approach on YOHO and MERCURY data sets

4.3 Comparison of Methods on Multiple Utterances

Because of the non-uniformity of test utterance lengths on the MERCURY corpus, we performed additional experiments using multiple utterances for speaker identification on MERCURY. Multiple utterance trials were performed by combining the N-best list scores from single utterance trials for a particular speaker. For each M-wise grouping of N-best lists, each list was first normalized by the score of its lowest scoring speaker. The scores for each speaker were then added together over all M lists to create a cumulative N-best list which was then resorted. This method, which is illustrated in Figure 4-3, is equivalent to using cumulative speaker identification scores over several utterances.

Identification error rates over 1, 3, and 5 utterances are shown in Table 4-4. For all methods, scoring over multiple utterances resulted in significant reductions in error rates. We observed that the speaker adaptive approach attained the lowest error rates among the individual classifiers as the number of test utterances was increased (Figure 4-4). Moreover, as the number of utterances was increased past 3, the performance of the combined classifiers exhibited no significant gains over the speaker adaptive approach. When compared to the next best individual classifier, the speaker adaptive approach yielded relative error rate reductions of 28%, 39%, and 53% on 3, 5, and 10 utterances respectively.

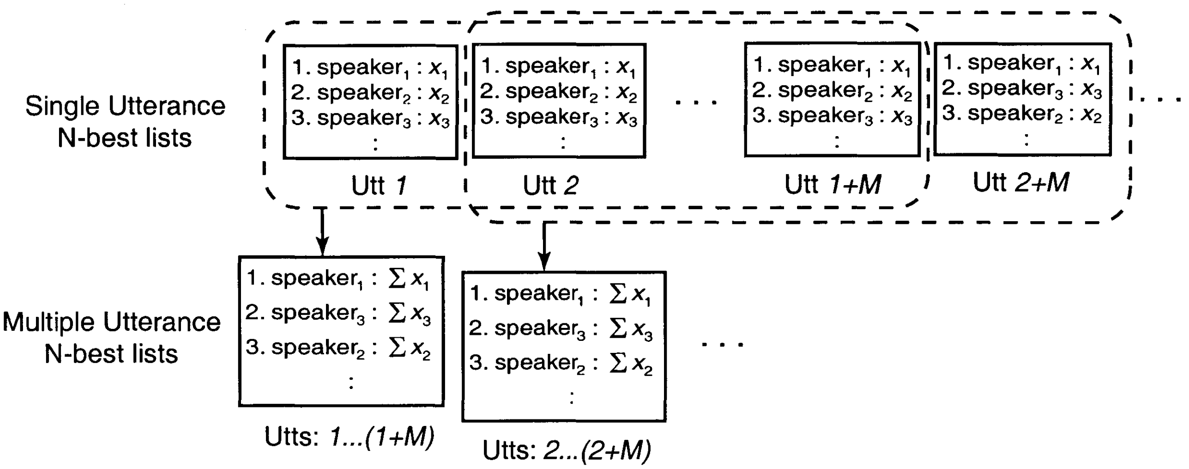

Figure 4-3: Cumulative scoring over M utterances from a particular speaker

	Error Rate (%)		
Method	1 Utt	3 Utt	5 Utt
Baseline (GMM)	22.4	14.3	13.1
PS	21.3	15.6	14.3
PC	21.6	14.9	13.8
SA	27.8	10.3	7.4
GMM+SA	19.0	9.7	7.5
PS+SA	18.3	11.2	8.0

Table 4-4: Identification error rates over 1, 3, and 5 utterances on MERCURY corpus

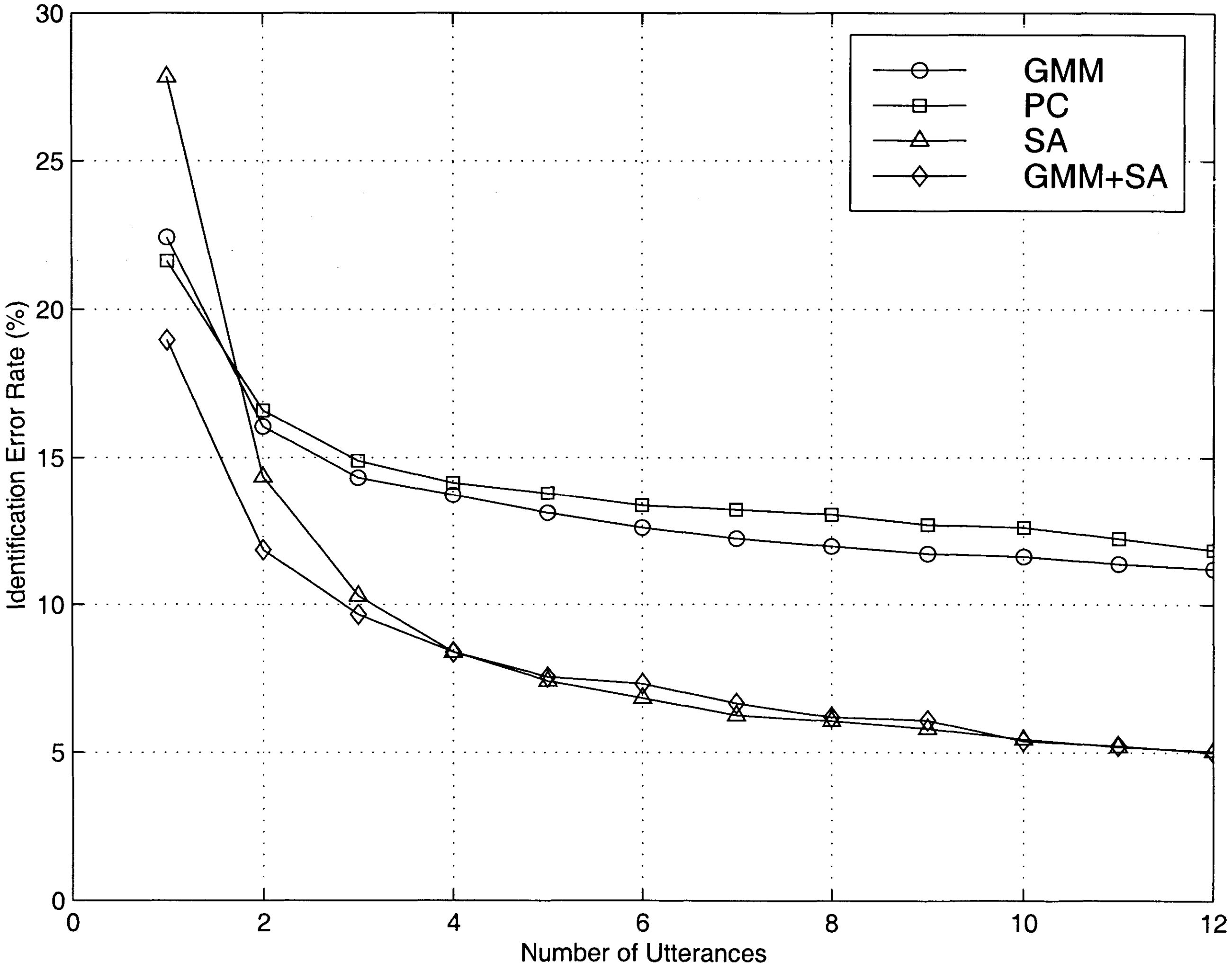

Figure 4-4: Comparison of identification error rates over multiple utterances on MERCURY corpus

Chapter 5

Confidence Scoring Techniques for Speaker Verification

In this chapter, we describe the task of speaker verification and review some currently used techniques. We then outline a novel method for performing speaker verification using confidence scoring techniques and present the results of experiments conducted using this technique.

5.1 Background

For the remainder of this chapter, we will use the notation X_j to indicate that utterance X was produced by speaker S_j. In speaker verification, the inputs to the system are an utterance, X_j, and a claimed speaker identity, S_i. If S_i is in the set of enrolled speakers, $\mathcal{S}$, the goal of the system is to determine whether or not $S_j = S_i$. Phrased as a detection task, the goal is to confirm whether or not X_j was produced by S_i.

Both of these formulations of the verification task differ from the closed-set identification task in two important ways. First, a system performing verification has the advantage of being provided with an appropriate target speaker model to compare against. Second, the verification system has the added difficulty of computing a score for a potentially unenrolled speaker, if S_j is not in $\mathcal{S}$.

5.1.1 Rank-based verification

Rank-based verification is a natural first approach to performing verification if a speaker identification system is already available. Given an input utterance, X, the system simply performs speaker identification to produce an N-best list of most likely candidates. The speaker is then accepted as S_i if S_i appears on the N-best list, and rejected otherwise. The verification threshold can be modified by increasing or decreasing N. This approach was used by Sarma for evaluation of verification performance. However, performance of purely rank-based verification systems are generally not scalable to large speaker sets [6].

5.1.2 Codebook based verification

Codebook based methods for verification essentially segment the feature space into regions for each speaker. One of the first operational speaker verification systems, which was used for building access control at Texas Instruments in the 1980s, used this type of approach with dynamic time warping to build speaker reference templates. Verification decisions were based upon cumulative Euclidean distance measures between input frame features and speaker reference frame features.

Another example of this verification paradigm is the eigenvoices approach described in Chapter 1. More recently, researchers have investigated the use of support vector machines (SVM) for verification [11, 14, 18]. This technique essentially transforms the input feature space into a higher dimensional *kernel* space where speaker regions are divided by separating hyperplanes [2]. Verification scores are then computed as the distance from the nearest hyperplane for the target speaker.

5.1.3 Statistical verification

The predominant approach to speaker verification is based on statistical detection theory. Given an input utterance, X, and a claimed speaker identity, S_i, the verification decision can be made by comparing two probabilities:

- $P(S_i|X)$ - the likelihood that the speaker is S_i given X.
- $P(\overline{S_i}|X)$ - the likelihood that the speaker is *not* S_i given X.

The decision rule based on these likelihoods then becomes

$$\text{Decision} = \begin{cases} \text{Accept} & \text{if } P(S_i|X) > P(\overline{S_i}|X); \\ \text{Reject} & \text{otherwise} \end{cases} \tag{5.1}$$

Using Bayes' rule, the comparison in Equation 5.1 can be rewritten as

$$\frac{P(X|S_i)P(S_i)}{P(X)} \overset{?}{\gtrless} \frac{P(X|\overline{S_i})P(\overline{S_i})}{P(X)} \tag{5.2}$$

$$\frac{P(X|S_i)}{P(X|\overline{S_i})} \overset{?}{\gtrless} \frac{P(\overline{S_i})}{P(S_i)} \tag{5.3}$$

In practice, the left-hand side of Equation 5.3 is computed as a logarithmic likelihood ratio, and the ratio of prior probabilities on the right-hand side is set at some threshold, θ. Equation 5.3 then becomes

$$\log P(X|S_i) - \log P(X|\overline{S_i}) \overset{?}{>} \theta \tag{5.4}$$

The logarithmic likelihood ratio in Equation 5.4 measures how much better the target model scores on the claimant utterance in comparison to an imposter model. Using the same types of modeling approaches described in previous chapters, there are a number of well-defined methods for estimating $P(X|S_i)$. However, estimation of

$P(X|\overline{S_i})$ is not as well-defined. Since $\overline{S_i}$ must essentially model the class of all speakers that are *not* S_i, the approaches for dealing with this problem are necessarily dependent on the application scenario and the level of security desired. In situations with many enrolled users and only moderate security requirements, such as telephone voice mail, the class of potential imposters for a particular speaker is likely to draw from the general population of speakers, with widely varying voice characteristics. This type of "casual" imposter scenario requires a model for $\overline{S_i}$ which captures the voice characteristics of the general speaker population, and is more concerned with computational efficiency and speaker model sizes. In contrast, for high-security applications with limited numbers of enrolled speakers, imposters are likely to be individuals with voice characteristics that are similar to those of the claimed identity. Such "dedicated" imposter scenarios require more specific models for $\overline{S_i}$ and are more concerned with security than convenience or computational complexity. The following sections describe different approaches for modeling $\overline{S_i}$.

Universal Background Models

The use of universal background models (UBM) relies on the assumption that all of the speakers in $\mathcal{S}$ represent an accurate sample of the general population of speakers. The universal background model is constructed by training a speaker independent speaker model on existing training data. The advantage of the universal background model approach for estimating $P(X|\overline{S_i})$ is that the model is not speaker specific. That is, the model for $\overline{S_i}$ is the same for all speakers, making score computation very efficient. Since there is no need to store a separate background model for each speaker, it is also space efficient. The disadvantage is that the system will be particularily vulnerable to falsely accepting dedicated imposters, whose voice characteristics may be similar to S_i. This is because the logarithmic likelihood ratio in Equation 5.4 will be high, since the UBM model does not discriminatively model speakers that are close to S_i.

Cohort Background Models

A *cohort* of a speaker, S_i, is defined as a speaker whose voice characteristics are similar to S_i. In [24], the distance measure used to determine similarity between two speakers, S_i and S_j, is a divergence related metric,

$$d(S_i, S_j) = \log \frac{P(X_i|S_i)}{P(X_i|S_j)} + \log \frac{P(X_j|S_j)}{P(X_j|S_i)} \tag{5.5}$$

In Equation 5.5, X_i and X_j are training utterances for speakers S_i and S_j, respectively.

The use of cohort-set models has been proposed as a way of ameliorating the difficulties experienced with the UBM in dealing with non-casual imposters. In this approach, $P(X|\overline{S_i})$ is estimated using the models for the speakers in the cohort set for S_i, $\mathcal{C}(S_i)$. Mathematically, this is expressed as

$$P(X_j|\overline{S_i}) = \frac{1}{|\mathcal{C}|}\sum_{S_k\in\mathcal{C}} \exp\left(\frac{1}{T}\sum_{t=1}^{T}\log P(x_t|S_k)\right) \tag{5.6}$$

There are a number of disadvantages to using $\mathcal{C}(S_i)$ as the basis for $\overline{S_i}$ in Equation 5.6. First, in order to determine the cohort sets for each speaker, additional development data is required. Second, a separate background model set is required for each speaker, which implies more computation. Finally, cohort-based background models are vulnerable to accepting imposters with very dissimilar voice characteristics to S_i. This is because utterances from dissimilar speakers score poorly on models for both S_i and $\mathcal{C}(S_i)$, which can lead to a high overall logarithmic likelihood ratio.

Speaker Specific Background Models

A solution to the vulnerability problems faced by the previous two background modeling approaches is to use B speakers to create a "balanced" background speaker set, $\mathcal{B}(S_i)$, for each speaker, S_i. This technique was illustrated by Reynolds [24], who used the distance metric in Equation 5.5 to select $B/2$ dissimilar (far) speakers and $B/2$ similar (near) speakers to model $\overline{S_i}$ for each S_i. Using the distance metric described in Equation 5.5, the far background speakers, $\mathcal{B}_f(S_i)$, were chosen using the following algorithm.

1. Set $\mathcal{F}(S_i)$ equal to the N farthest speakers from S_i, where $N \geq B$.

2. Initialize by moving the farthest speaker from $\mathcal{F}(S_i)$ into $\mathcal{B}_f(S_i)$.

3. Move S_f from $\mathcal{F}(S_i)$ into $\mathcal{B}_f(S_i)$, where S_f is given by

$$S_f = \arg\max_{S_f\in\mathcal{F}(S_i)}\left\{\frac{1}{|\mathcal{B}_f(S_i)|}\sum_{S_b\in\mathcal{B}_f(S_i)} d(S_b,S_f)d(S_i,S_f)\right\}$$

4. Repeat Step 3 until $|\mathcal{B}_f(S_i)| = B/2$.

Similarily, the near background speakers, $\mathcal{B}_c(S_i)$, were chosen using the following following algorithm.

1. Set $\mathcal{C}(S_i)$ equal to the N nearest speakers from S_i, where $N \geq B$.

2. Initialize by moving the nearest speaker from $\mathcal{C}(S_i)$ into $\mathcal{B}_c(S_i)$.

3. Move S_c from $\mathcal{C}(S_i)$ into $\mathcal{B}_c(S_i)$, where S_c is given by

$$S_c = \arg\max_{S_c\in\mathcal{C}(S_i)}\left\{\frac{1}{|\mathcal{B}_c(S_i)|}\sum_{S_b\in\mathcal{B}_c(S_i)} \frac{d(S_b,S_c)}{d(S_i,S_c)}\right\}$$

4. Repeat Step 3 until $|\mathcal{B}_c(S_i)| = B/2$.

The rationale for choosing background speakers in this manner is to choose far speakers and near speakers which are maximally spread, in order to provide maximum coverage of their respective sets, and prevent selection of "duplicate" background speakers. The advantage of this type of background modeling is that both casual and dedicated imposters are sufficiently modeled by the far and near speakers in $\mathcal{B}(S_i)$, respectively.

5.2 Evaluation Metrics

The standard method for evaluating the performance of a detection or verification system is the receiver operating characteristic (ROC) curve of the system. As discussed in Section 5.1, verification systems usually return a verification score, which is then compared to some pre-defined threshold in order to decide whether to accept or reject. At different thresholds, or operating points, the system will exhibit different relative levels in the types of classification errors that are made. With lower thresholds, the system is more likely to correctly accept or detect true speakers, but is also more likely to falsely accept imposters. Conversely, with higher thresholds, the system is more likely to correctly reject imposters, but is also more likely to incorrectly reject true speakers. The ROC curve of a system plots the tradeoff between the rate of detection and the rate of false acceptance as the verification threshold is varied. The rate of detection, is given by

$$P(\text{detection}) = P(S_j \text{ accepted as } S_i | S_j = S_i)$$

Similarily, the rate of false acceptance, or false alarm, is given by

$$P(\text{false alarm}) = P(S_j \text{ accepted as } S_i | S_j \neq S_i)$$

An example ROC curve is shown in Figure 5-1.

An alternative method of presenting the operating characteristic of a system is the detection error tradeoff (DET) curve, which is the standard method used by the National Institute of Standards and Technology for presenting performance results of speaker detection systems [19]. The DET curve plots detection error (miss) probability against false alarm probability using logarithmic axes. This type of plot is preferred over the ROC curve in situations where systems have similar performances, or when high accuracies make it difficult to observe operating points on the curve. In [19], Martin *et al.* enumerate three reasons for this preference. First, the logarithmic transform of the ROC curve better distinguishes asymptotic performance characteristics of different systems as error rates go to zero. Second, for the DET curve, error rates are plotted on both axes, in contrast to the ROC curve, which plots error rate on one axis, and detection rate on the other. Finally, plotting the DET curve is advantageous because the slope of the curve indicates whether or not the underlying distribution of scores for the two classes fit a normal distribution.

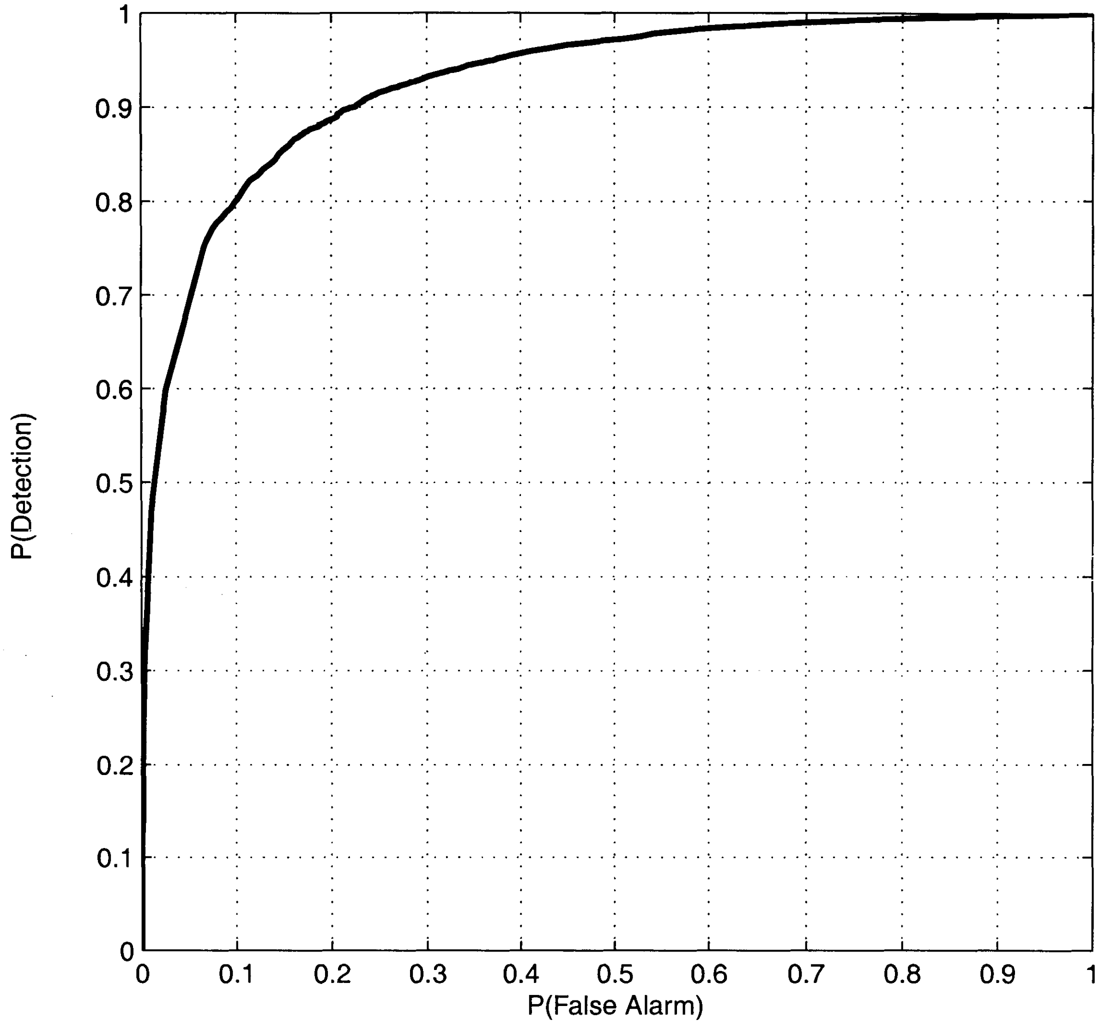

Figure 5-1: Sample ROC curve for a speaker verification system. As the operating point moves to the right and the threshold is increased, the rate of detection approaches 1, as does the rate of false acceptance. As the operating point moves to the left, the false acceptance rate and the detection rate both go to 0.

As an example, speaker verification results for the four different modeling approaches from Chapter 3 are shown for the YOHO data in Figure 5-2 and Figure 5-3. For these results, a simple rank-based verification metric was used. While both figures show the same data, the DET plot in Figure 5-3 better distinguishes performance along the asymptotic regions of the curves, where the error probabilities go to zero.

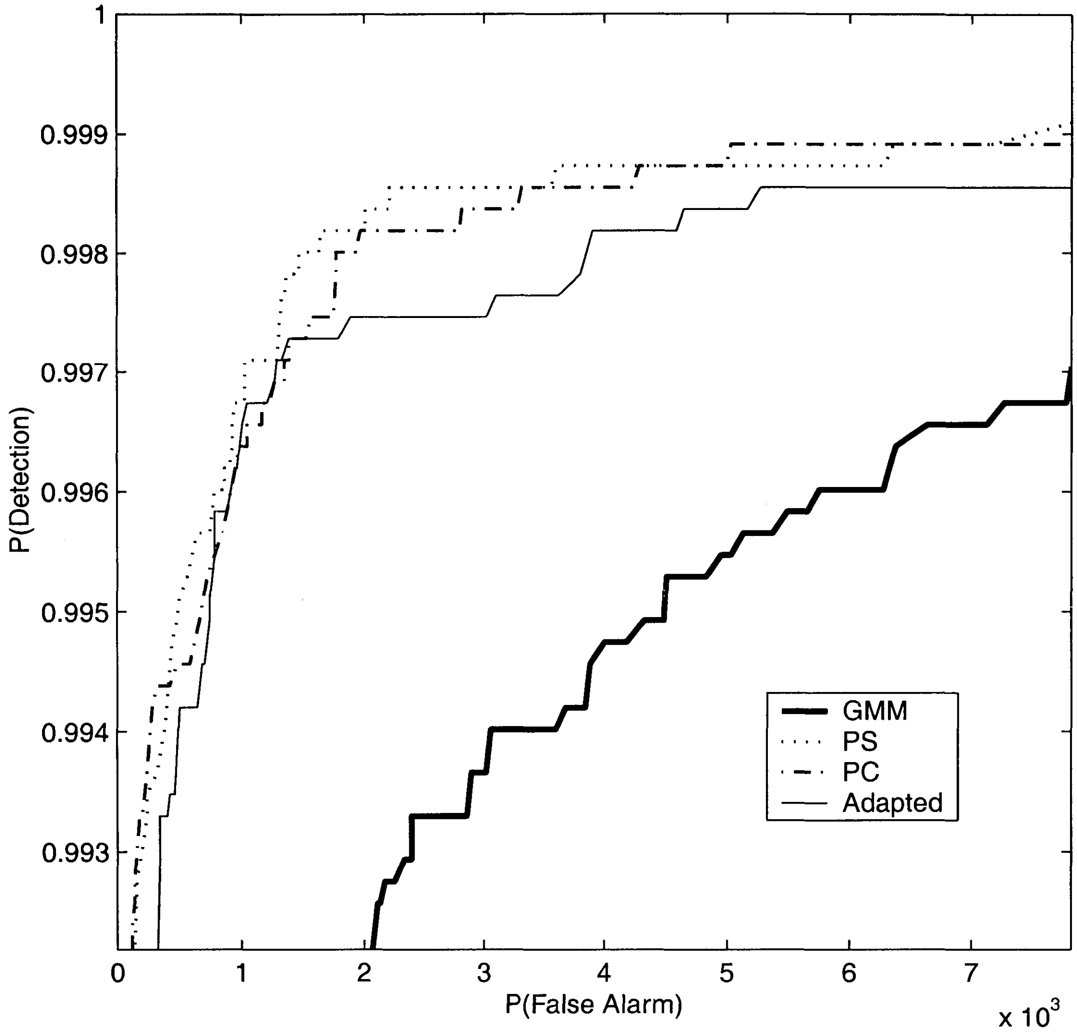

Figure 5-2: Closeup of upper left hand corner of ROC curves on YOHO for verification systems based on the four modeling methods from the previous chapter.

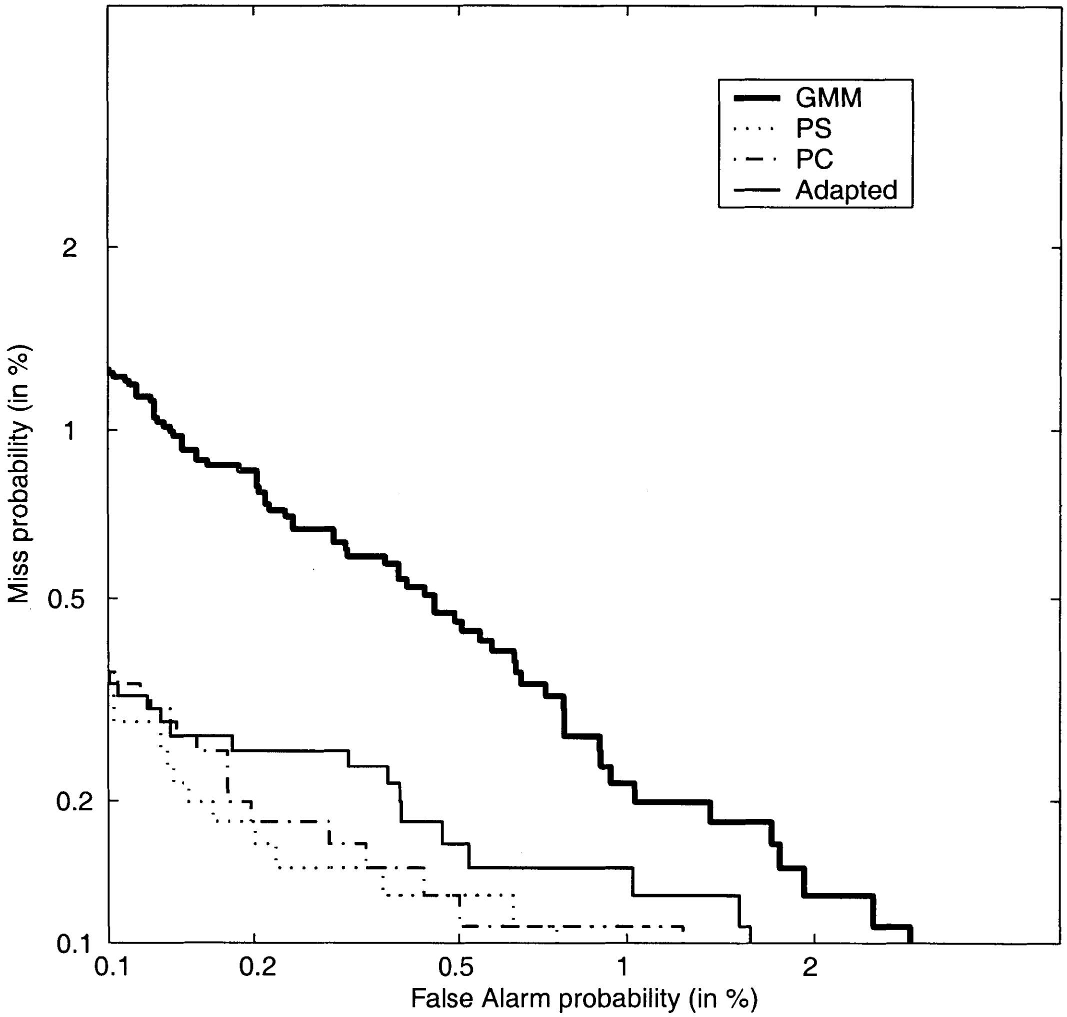

Figure 5-3: Equivalent DET curve to Figure 5-2 for verification results on YOHO.

5.3 Confidence Scoring Approach

The approaches described in the previous section use various individual metrics to produce a verification score. This section describes a method for combining multiple metrics, which has the potential to reduce verification error. The approach we use is similar to a technique used for confidence scoring [15].

First, speaker specific background model sets, $\mathcal{B}(S_i)$, are determined for each speaker, S_i, as described in Section 5.1.3. For a given utterance and target speaker, S_i, log probability scores are computed for S_i as well as for each of the speakers in $\mathcal{B}(S_i)$. From this sorted background speaker list (B-list), a verification feature vector is extracted for speaker S_i. These verification features, which are described in the next section, are then combined using a trained linear discriminant vector to produce a final verification score.

5.3.1 B-list Features for Verification

From the B-list for a particular speaker, S_i, it is possible to extract several features that can be used for verification. As mentioned earlier, some verification systems use metrics such as rank, speaker score, background model score, and cohort model score. In general, there are a variety of methods for choosing a relevant feature set. For example, in [15], a greedy algorithm was used to find the most important set of confidence features to extract from a word and utterance level N-best lists. Since the space of features considered for our approach was relatively small, we used each of the features described below:

1. **Utterance Length:** This is the number of speech observations or input feature vectors, $\{x_t\}$ from the utterance, X.

2. **Target Rank:** This is the position of the target speaker, S_i in the B-list.

3. **Total Target Score:** This is the total log probability score of the target speaker for the utterance

$$T \log P(X_j|S_i) = \sum_{t=1}^{T} \log P(x_t|S_i)$$

4. **Average Target Score:** This is the log probability score for the target speaker, normalized by the number of observations

$$\log P(X_j|S_i) = \frac{1}{T} \sum_{t=1}^{T} \log P(x_t|S_i)$$

5. **Total Background Score:** This is the total log probability score of the background speaker model for the utterance

$$T \log P(X_j|\overline{S_i}) = T \log \left\{ \frac{1}{B} \sum_{S_b \in \mathcal{B}(S_i)} \exp \left(\frac{1}{T} \sum_{t=1}^{T} \log P(x_t|S_b) \right) \right\}$$

6. **Average Background Score (Arithmetic):** This is the average background speaker log probability score taken as an arithmetic average over the background speakers.

$$\log P(X_j|\overline{S_i}) = \log \left\{ \frac{1}{B} \sum_{S_b \in \mathcal{B}(S_i)} \exp \left(\frac{1}{T} \sum_{t=1}^{T} \log P(x_t|S_b) \right) \right\}$$

7. **Average Background Score (Geometric):** This is the average background speaker log probability score taken as an geometric average over the background speakers.

$$\log P(X_j|\overline{S_i}) = \frac{1}{B} \sum_{S_b \in \mathcal{B}(S_i)} \left(\frac{1}{T} \sum_{t=1}^{T} \log P(x_t|S_b) \right)$$

8. **Total "Top M" Score:** This is the total log probability score of the speaker model specified by the top M speakers on the B-list for the utterance X_j.

9. **Average "Top M" Score (Arithmetic):** This is the average log probability score of the top M speakers computed as an arithmetic average over the top M speakers for the utterance X_j.

10. **Average "Top M" Score (Geometric):** This is the average cohort speaker log probability score taken as a geometric average over the top M speakers.

11. **Difference from Best:** This is the difference of the log probability scores for the top-scoring speaker on the B-list and S_i.

$$\max_{S_b \in \mathcal{B}(S_i), S_i} \log P(X_j|S_b) - \log P(X_j|S_i)$$

5.3.2 Linear Discriminant Analysis

Linear discriminant analysis (LDA) is a commonly used technique for performing dimensionality reduction for data classification. Though LDA can be used for problems with an arbitrary number of classes, L, we use only the two class subproblem, with $L = 2$. Given two sets of K-dimensional data vectors, $\mathcal{A} = \{\vec{a}_1, \ldots, \vec{a}_{K1}\}$ and $\mathcal{B} = \{\vec{b}_1, \ldots, \vec{b}_{K2}\}$, the goal of LDA is to find the optimal projection vector, $\vec{p}$, which transforms the vectors from $\mathcal{A}$ and $\mathcal{B}$ into a lower dimensional subspace while maximizing the separability between the two classes.

To perform speaker verification based on a set of verification features, rather than any single metric, two class LDA is used to combine the verification features. In this case, the two classes are the true speakers and the imposters, and the data vectors are the 11 verification features described in the previous section. Since standard verification decision tasks require a numerical verification score, each verification feature vector, $\vec{f}$ must be projected down to a one dimensional verification score. The projection vector, $\vec{p}$, is therefore a (1×11) dimensional weighting vector, where the ith dimension of $\vec{p}$ corresponds to the weight of the ith verification feature.

$$\text{Verification Score} = \vec{p}^{\,T}\vec{f} = \sum_{i=1}^{N} p_i f_i \tag{5.7}$$

To train $\vec{p}$, a data set containing correctly labelled examples of verification feature vectors from both classes is required. Training of $\vec{p}$ is performed in two stages. First, $\vec{p}$ is initialized using Fisher LDA [8]. Next, an iterative hill-climbing algorithm is used to modify the components of $\vec{p}$ to optimize a particular objective function [22]. Because the number of imposter trials is, in general, much greater than the number of true speaker trials, the algorithm was set to minimize the rate of false acceptances. In principle, however, this minimization can be performed on other criteria, such as false rejection rate, equal error rate, or overall classification error.

Once $\vec{p}$ has been trained, testing is performed as follows. Given an input utterance and a target speaker identity, S_i, a B-list of speakers is generated using the scores of the speakers in $\mathcal{B}(S_i) \cup S_i$. The verification feature vector is then extracted from the B-list, and the verification score is computed using the trained projection vector.

5.4 Experimental Conditions

5.4.1 Corpus Creation

To evaluate the technique described in the previous section, we used an extended version of the MERCURY corpus. Although the training portion of the MERCURY corpus contained enrollment data for 38 speakers, an essential component of speaker verification evaluation is observing how well the system is able to reject imposter trials presented by *unseen* speakers. To this end, utterances from 80 imposter speakers (50 male, 30 female) not in the training set were identified from MERCURY call logs. These "supplementary" utterances were partitioned by speaker label into two sets, each containing 40 imposter speakers (25 male, 15 female). This partitioning was done in order to ensure that the imposters in the test set were unseen during the training of the LDA projection vector on the development set. The two imposter sets were then used to augment the MERCURY development and test sets with imposter trials.

For the development data, 1145 supplementary utterances were exhaustively tested against each of the 38 enrolled speakers to give a total of 43,510 imposter trials. Similarly, for the test data, 1136 supplementary utterances were used to generate 43,168 imposter trials. None of the enrolled speakers were used as imposters. The utterances in the existing development and test sets were used to generate 3,071 and 3,157 true speaker trials, respectively. The properties of the augmented Mercury corpus are summarized in Table 5-1.

	Mercury (Original corpus)		Mercury (Augmented corpus)	
Property	True Speakers Development	True Speakers Test	Imposters Development	Imposters Test
Number of Speakers	18M 20F	18M 20F	25M 15F	25M 15F
Number of Utterances	3157	3071	1145	1136
Number of Trials	3157	3071	43510	43168

Table 5-1: Summary of augmented Mercury corpus properties. For the augmented imposter sets, each speaker is present in only one of the two sets.

5.4.2 Training and Testing

First, speaker models were trained using the same methods described in Chapter 3. For the experiments in this chapter, we used only the baseline GMM speaker modeling approach. GMM speaker models were trained for each of the 38 enrolled speakers in the training portion of the Mercury corpus as described in Chapter 2. Speaker specific background sets were then determined for each speaker using the method described in Section 5.1.3, with $B = 10$ and $N = 10$.

Baseline Approach

For the baseline approach, we computed verification scores based on the background speaker model approach proposed by Reynolds in [24]. For each trial utterance, X_j, using speaker S_i as the target speaker, the verification score was computed using the average score of $\mathcal{B}(S_i)$ as an estimate of $P(X_j|\overline{S_i})$. Mathematically, this is expressed as

$$\text{Verification Score} = \log P(X_j|S_i) - \log \left\{ \frac{1}{B} \sum_{S_b \in \mathcal{B}(S_i)} P(X_j|S_b) \right\} \qquad (5.8)$$

Confidence Scoring Approach

In order to combine multiple verification features for the confidence scoring approach, the LDA projection vector, $\vec{p}$, was trained on verification features extracted from development data for the true speakers and imposters. As described in Section 5.3.2,

$\vec{p}$ was first initialized using Fisher LDA. Subsequently, several optimization criteria were used to iteratively train the final LDA projection vector using a hill climbing algorithm. The three criteria examined were:

- Low false alarm rate (LFA): Using this criterion, the LDA algorithm attempts to minimize the detection error rate in the region where the false alarm rate is between 0.0 and 0.1.
- Equal error rate (EER): Using this criterion, the LDA algorithm attempts to minimize the point where false alarm rate equals miss rate.
- Low detection error rate (LDE): Using this criterion, the LDA algorithm attempts to minimize the false alarm rate in the region where the detection error rate is between 0.0 and 0.1.

Once the LDA projection vector was trained, testing was performed by extracting verification features for each imposter and true speaker trial in the augmented MERCURY test set. Verification scores were then computed using Equation 5.7

5.5 Results

The components of the projection vector, $\vec{p}$, trained using LDA are shown in Table 5-2. These components are equivalent to weighting coefficients used for the corresponding components of the verification feature vector. The second column of Table 5-2 illustrates that the baseline verification approach is a special case of the general LDA approach, as it is a linear combination of two verification features: the average target score, and the average background score.

Using the initial Fisher LDA projection vector to compute the verification scores resulted in the DET curve shown in Figure 5-4. In comparison to the set of operating points generated by the baseline approach, scoring using the Fisher LDA projection vector yielded substantially higher detection error rates for each operating point on the curve.

When trained on the development data to optimize for LFA, EER, and LDE, the projection vectors resulted in verification scores which produced the operating characteristics shown in Figure 5-5. Although the optimized Fisher LDA projection vectors showed substantial improvement over the system using the initialized vector, performance for all three systems was similar to the baseline method.

Figures 5-6, 5-7, and 5-8 show closeups of the DET curves from Figure 5-5 in the regions of interest for the three optimization criteria. Of the three LDA-based systems tested, the system using the LFA optimized projection vector had the best performance in all three regions of interest. This is likely due to the high number of imposter trials in comparison to true speaker trials in the testing data.

Verification Feature	Baseline Method	Fisher LDA	Hill Climb (LFA)	Hill Climb (EER)	Hill Climb (LDE)
rank	0.0	0.1544	-0.5856	-0.1606	-0.5606
num_obs	0.0	-0.0163	-0.0213	-0.0213	-0.0213
tot_tgt	0.0	0.0099	0.0099	0.0099	0.0099
avg_tgt	1.0	0.3840	0.5390	0.4240	0.0389
tot_back	0.0	-0.0322	-0.0322	-0.0322	-0.0322
avg_back	-1.0	-12.8324	-12.8324	-12.8324	-12.8324
geo_back	0.0	0.0070	0.0069	0.0069	0.0069
tot_topM ($M = 5$)	0.0	0.0163	0.0163	0.0163	0.0163
avg_topM ($M = 5$)	0.0	12.5032	12.5032	12.5032	12.5032
geo_topM ($M = 5$)	0.0	-0.0010	-0.0010	-0.0010	-0.0010
diff_top	0.0	0.0099	0.0099	0.0099	0.0099

Table 5-2: Weighting coefficients of each verification feature for the baseline method and three linear discriminant analysis results. The Fisher LDA vector is the initialized projection vector used for the hill climbing methods. The three hill climbing LDA variations were optimized on low false alarm rate (LFA), equal error rate (EER) and low detection error rate (LDE).

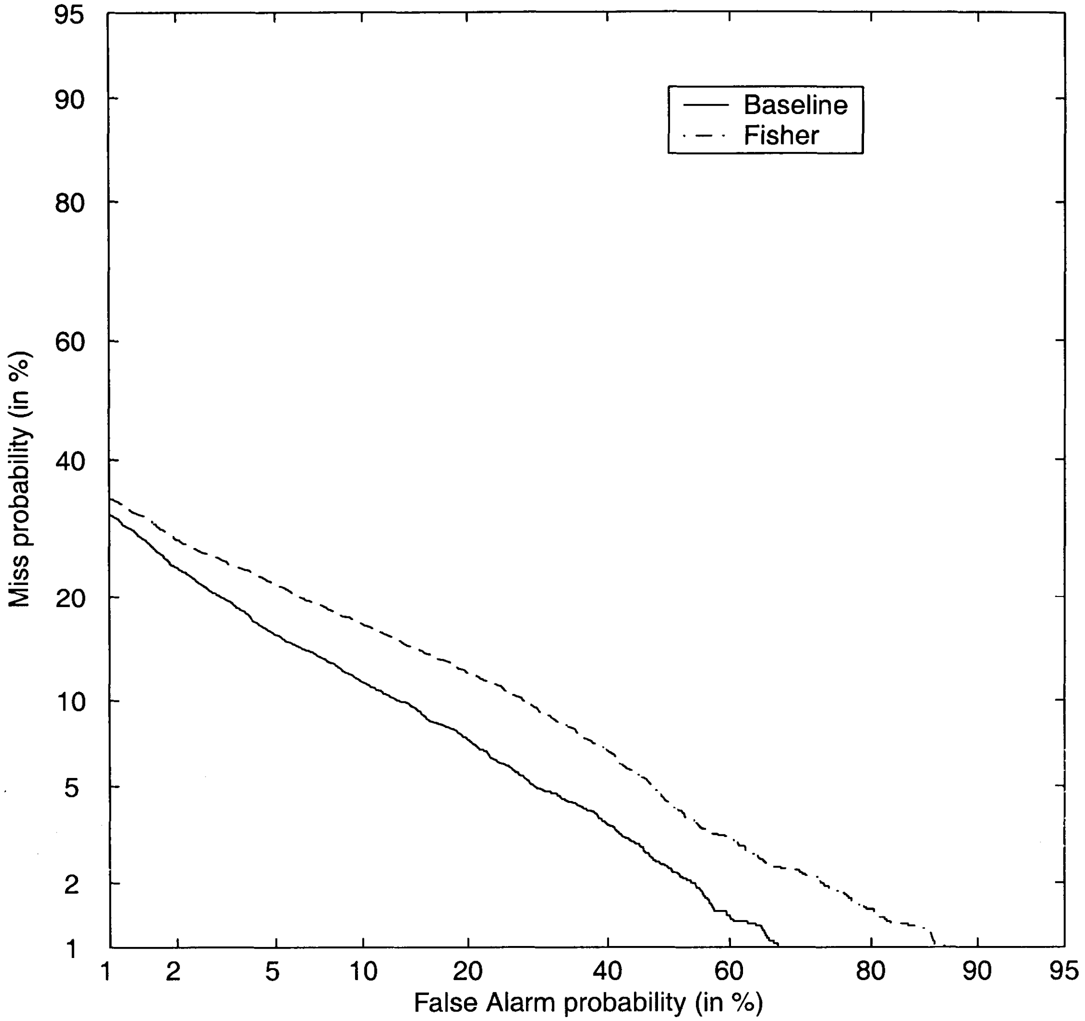

Figure 5-4: Detection error tradeoff curve for baseline and Fisher LDA based verification methods

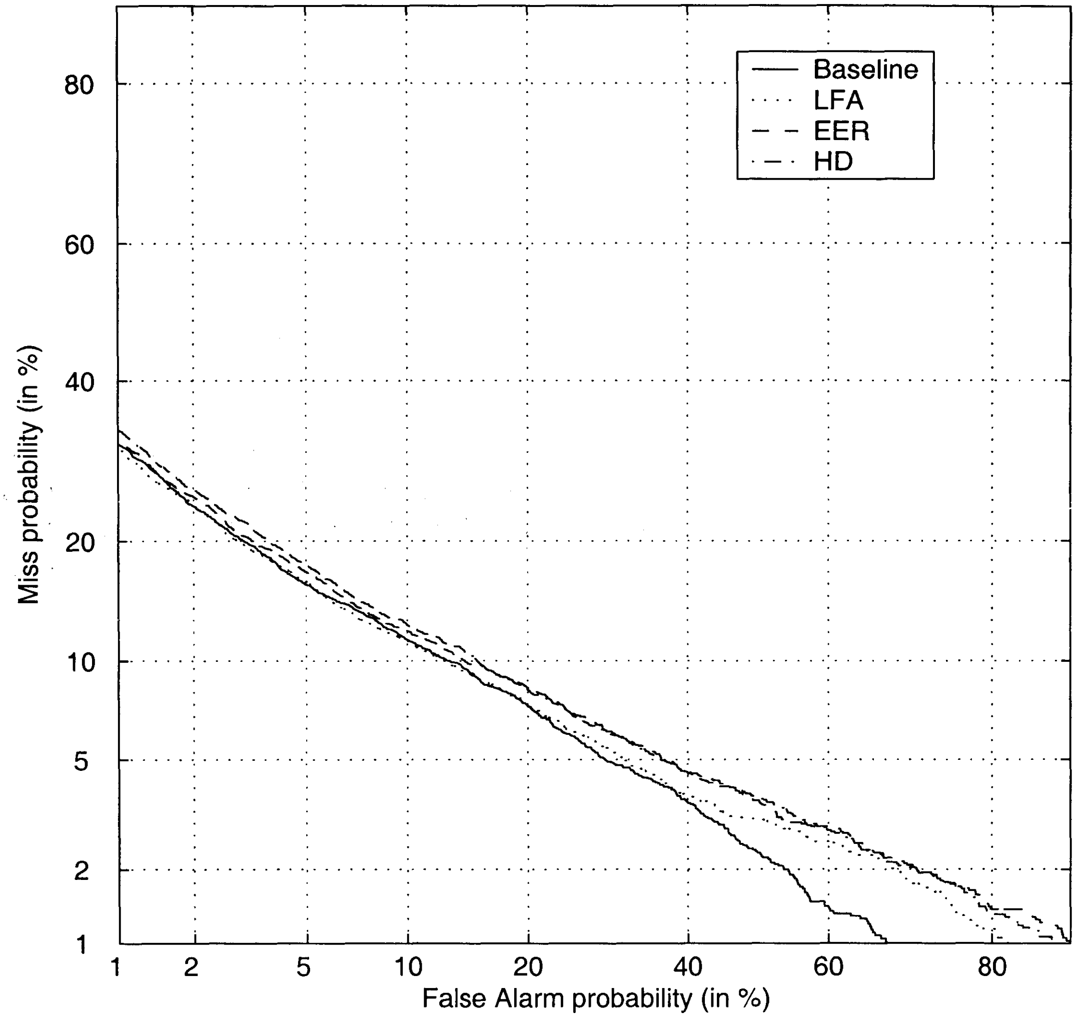

Figure 5-5: Detection error tradeoff curve for baseline and LDA based verification methods

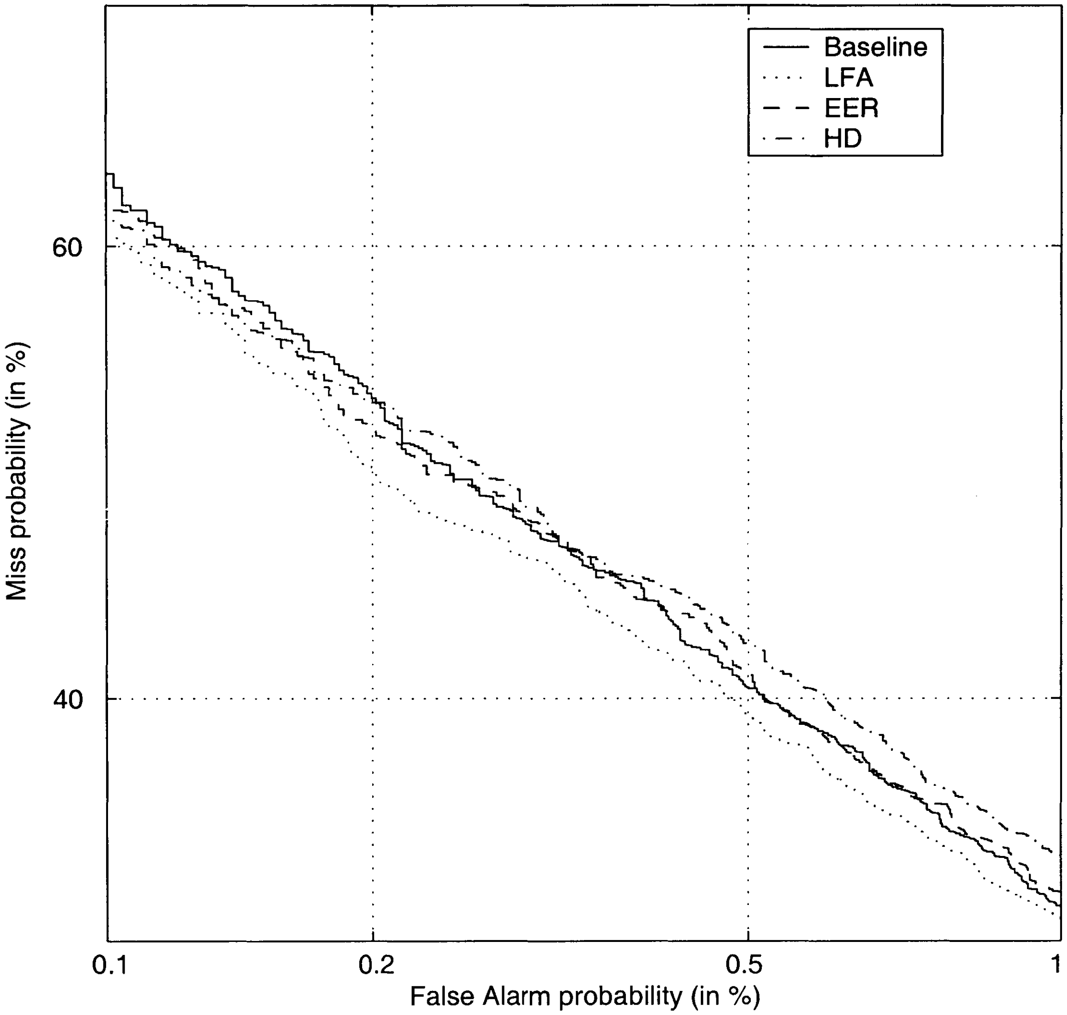

Figure 5-6: Closeup of DET curve from Figure 5-5 showing low false alarm (LFA) region

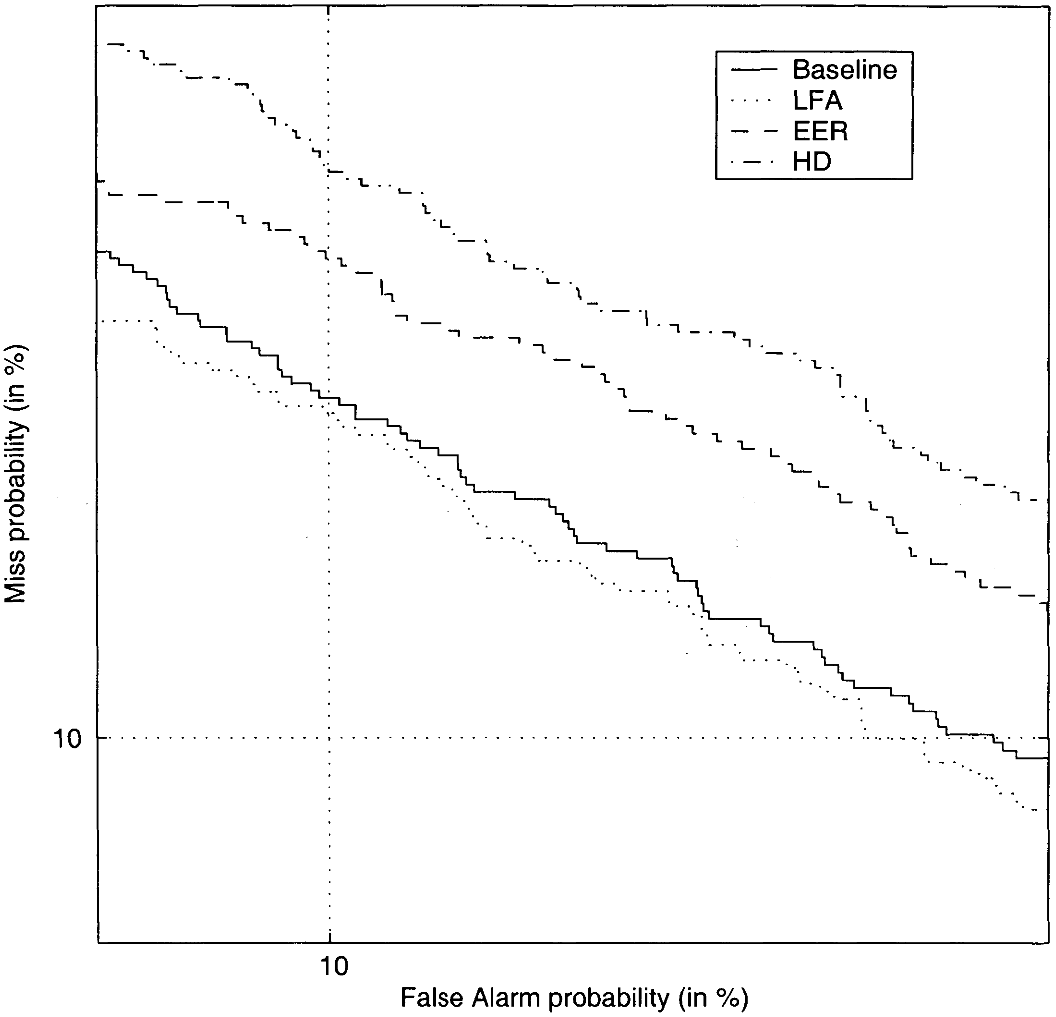

Figure 5-7: Closeup of DET curve from Figure 5-5 showing equal error rate (EER) region

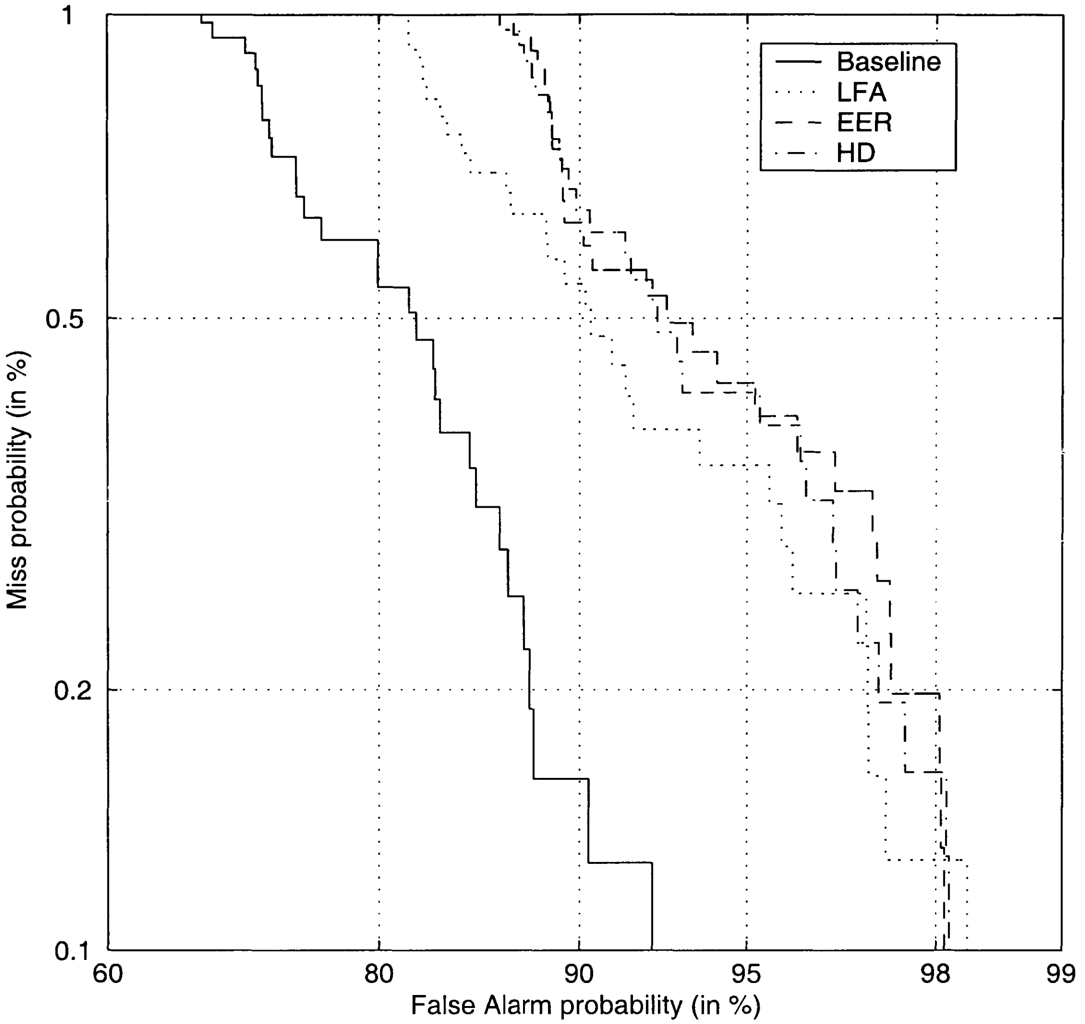

Figure 5-8: Closeup of DET curve from Figure 5-5 showing low detection error (LDE) region

Overall, we observed no noticeable difference in performance between the optimized LDA-based systems and the baseline system. Although we noted that the baseline system is a special case of the LDA approach, the optimized training procedures resulted in projection vectors which were very different from the baseline projection vector, but very similar to the initial Fisher LDA vector. Since the hill-climbing algorithm was able to significantly improve upon the performance of the Fisher LDA system, this suggested that the performance of the baseline could be improved upon by initializing the training algorithm with the baseline vector weights from Table 5-2.

In order to test this hypothesis, we performed an additional experiment where an LDA projection vector was trained using the initial baseline vector and the LFA optimization criterion. Figure 5-9 illustrates the comparative performance of the baseline system and the LFA optimized system initialized with the baseline projection vector. Performance for both systems was very similar, indicating that the baseline vector may be a local optimum for the optimization criteria used in the training algorithm.

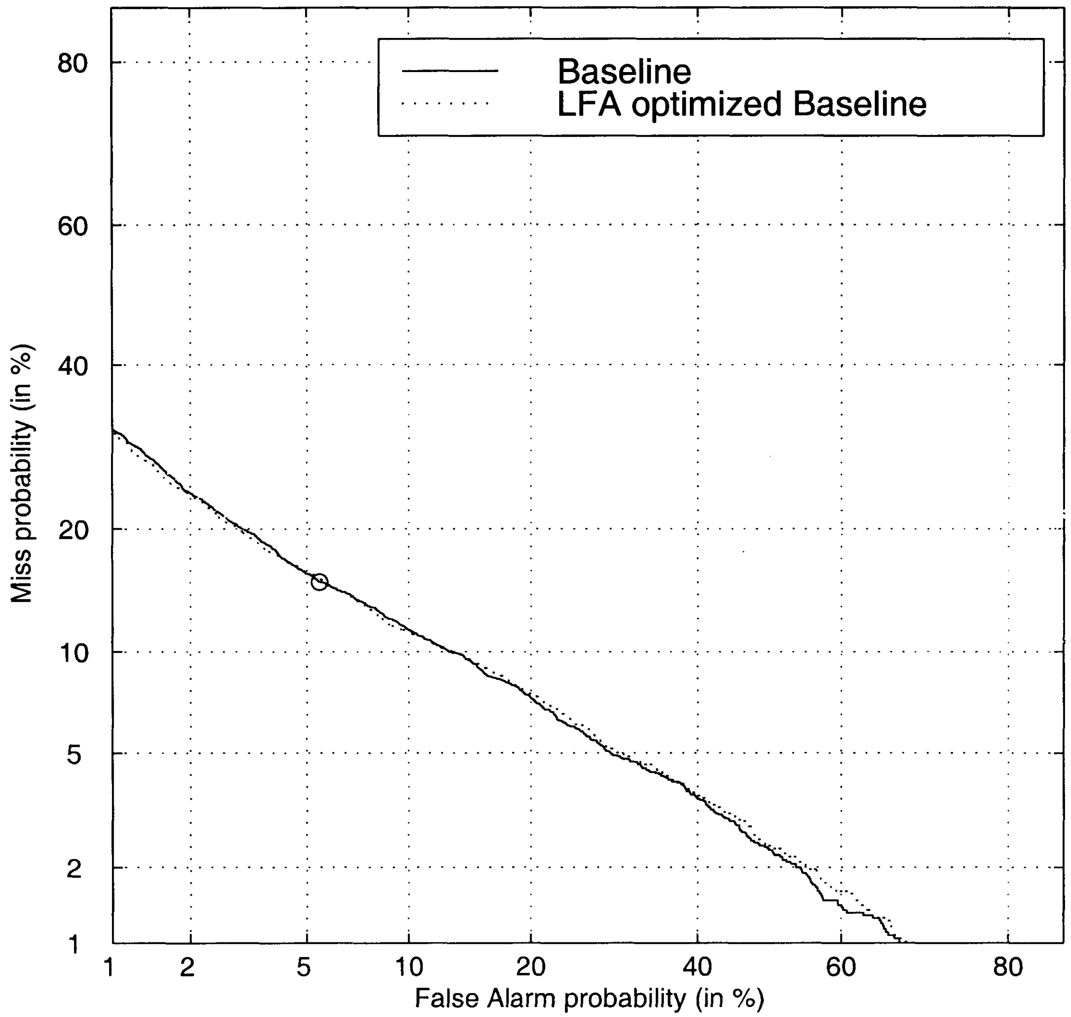

Figure 5-9: DET curve comparing performance of baseline system and LFA optimized baseline system.

Chapter 6

Alternative Features for Speaker Recognition in Telephone Quality Speech

In this chapter, we describe some alternative acoustic features to Mel-frequency cepstral coefficients (MFCCs) that can be used for speaker identification. Our goal in this chapter is to investigate features that are more robust to additive and convolutional noise than traditional MFCCs. First, we give an overview of the speaker distinguishing properties of MFCCs, formant frequencies, and fundamental frequency. Next, we describe the tools and algorithms used for feature extraction. Finally, we give the results of some feature set experiments comparing results on TIMIT and NTIMIT.

6.1 Background

The approaches used for speaker modeling used in the thesis until this point have relied on the same set of template-based spectral front-end features. Specifically, the features used are MFCCs, computed at frame level landmarks. Unfortunately, MFCCs typically exhibit severe degradations in identification accuracy in noisy and bandlimited environments. In this section, we briefly describe the motivation for using MFCCs and background information regarding formant locations and fundamental frequencies.

6.1.1 Mel-frequency Cepstral Coefficients

Many modern speech recognition systems use cepstral coefficients in some form as their front-end features [23]. In general, the cepstrum is a non-linear transformation of the frequency domain representation of waveform. Cepstral features are computed by taking the inverse Fourier transform of the logarithm of the short-time Fourier transform of a time domain waveform.

The Mel frequency scale is a non-linear frequency domain scaling which applies a filter bank to the input signal. The center frequencies increase linearly below 1 kHz and logarithmically thereafter. This scaling is motivated by perceptual studies of human auditory processing. The scaled bank of filters implementing this process is shown in Figure 6-1. The non-linear Mel-scale spectral coefficients obtained after filtering are then converted into the cepstral domain by taking the inverse Fourier transform of the logarithm of the spectral coefficients.

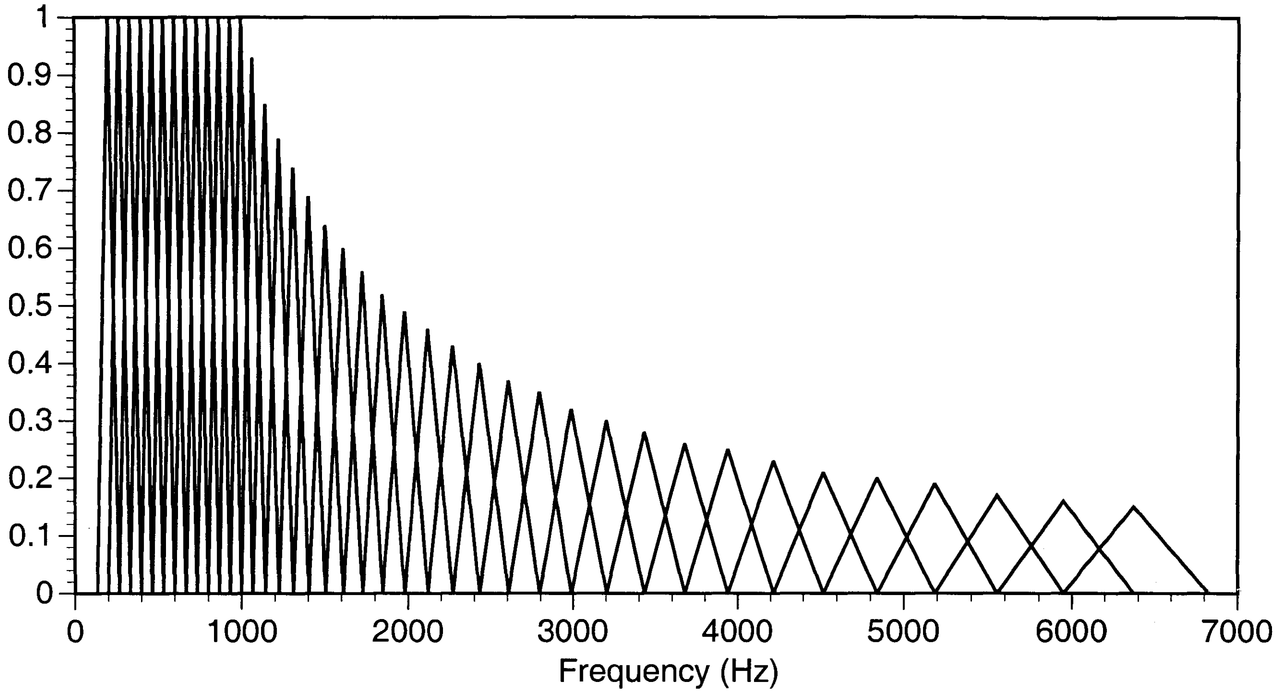

Figure 6-1: Bank of filters illustrating the Mel frequency scale.

A major advantage of using cepstral features is the ability to perform blind channel equalization using cepstral mean subtraction. This technique allows a system to efficiently deconvolve time-invariant channel effects from the signal. The disadvantage of cepstral features is that reliable feature extraction is dependent on absolute spectral magnitudes, which are easily distorted by additive noise.

6.1.2 Formant Frequencies

In this section, we are primarily concerned with the use of formants for voiced phones, such as vowels and semivowels. During vowel production, the overall received speech signal, $p_r(t)$, can be represented as the output of a periodic glottal source, $s(t)$, passed through a cascade of filters, $T(f)$, $R(f)$.

$$P_r(f) = S(f)T(f)R(f) \tag{6.1}$$

In Equation 6.1, $P_r(f)$ is the spectrum of the received speech signal. The terms on the right hand side: $S(f)$, $T(f)$, and $R(f)$ are the glottal source spectrum, the vocal tract transfer function, and the radiation characteristic, respectively. Typical representations of these spectra are shown in Figure 6-2. Of the transfer functions

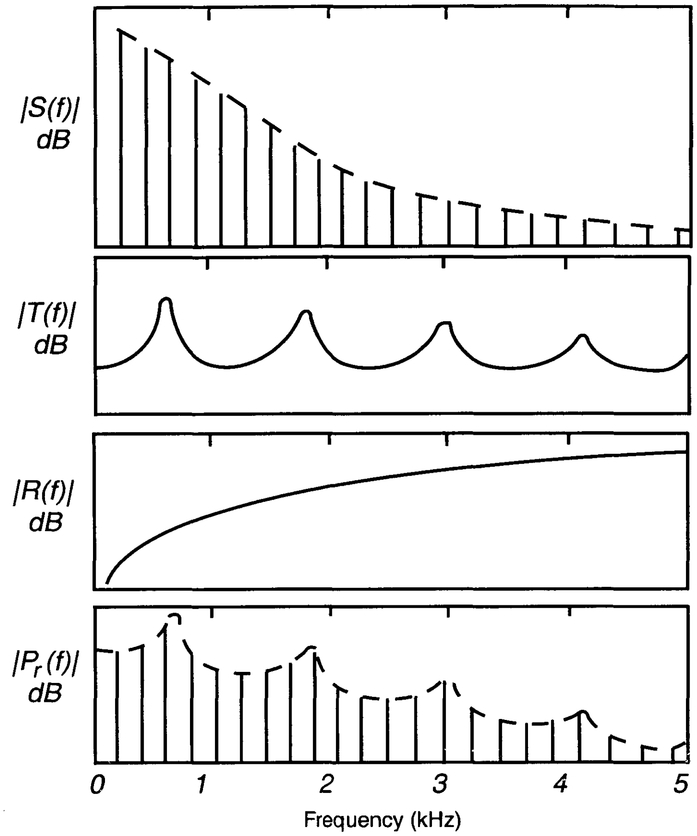

Figure 6-2: Representative spectra for transfer functions in Equation 6.1.

shown in the figure, formant frequencies are directly related to the vocal tract transfer function, $T(f)$.

In the acoustic theory of speech production, the vocal tract is modeled as a concatenation of tubes of varying widths and lengths. The size and shape of these tubes are dependent on speaker physiology and the vocal tract configuration used to produce a particular sound. When producing a vowel, the acoustic tube model for the vocal tract has characteristic resonant frequencies at the poles of the vocal tract transfer function. These resonant frequencies, or pole locations, are known as formant frequencies. Although there are an infinite number of formant frequencies in general, for the study of speech, we are usually only concerned with the first four formants (F_1, F_2,F_3,F_4), which lie in the range of 0-5 kHz.

While the positions of formants in relation to each other is usually consistent for specific vowels across speakers, absolute formant locations can vary widely across different speakers. These differences in formant measurements are potential speaker specific cues which can be used for performing speaker recognition. One motivation for using formants over spectral template-based features such as MFCCs is that formant locations are indicative of relative maxima in the speech. Therefore, in the absence of frequency modulation or frequency specific channel attenuation, estimation of formant locations may be more robust to additive and convolutional noise than MFCC measurements.

6.1.3 Fundamental Frequency

For voiced sounds, the fundamental frequency of the speech signal, is related directly to $s(t)$, the glottal source signal, in Equation 6.1. The glottal source spectrum can be represented as the product of a frequency domain impulse train, $X(f)$, and glottal source characteristic, $G(f)$.

$$S(f) = G(f)X(f) = G(f) \sum_{k=-\infty}^{\infty} \delta(f - kF_0) \tag{6.2}$$

In Equation 6.2, F_0 is the spacing between impulses in the frequency domain representation of $X(f)$, and is indicative of the rate of vibration for the vocal folds. The physiological basis for inter-speaker differences in F_0 measurements is mainly attributed to variation in vocal cord length, which directly affects the rate at which the vocal folds can vibrate. However, F_0 measurements also exhibit a wide degree of variation within speakers, as voice pitch can be easily altered by tensing and slackening the vocal folds. In conversational speech, F_0 varies throughout an utterance, depending on the type of phone being produced and the prosodic constraints of the utterance.

The motivations for using F_0 as a feature for speaker recognition are twofold. First, as with formant locations, F_0 is not dependent on absolute spectral values. Instead, it can be recovered from a received speech signal by measuring the minimum interval between peaks in the frequency domain. As such, the signal characteristics which are required to calculate F_0 are more robust to noise than for MFCCs. The second reason for including F_0 as a front-end feature is that F_0 trajectories could be used to model the degree and rate of pitch variation in a speaker's voice, which is an important higher-level cue that humans use for speaker recognition [6].

6.2 Feature Set Experiments

6.2.1 Feature Extraction Algorithms

In order to extract formant and F_0 data from the input waveform, these features were computed off-line using the Entropic Signal Processing Suite (ESPS). The two relevant commands used were `formant` and `get_f0`. In ESPS, the `formant` command computes formant locations at a specified frame interval. For each frame, linear predictive coding (LPC) coefficients are used to estimate the spectrum of the windowed signal using an all-pole model. The pole locations produced by the all-pole model are used as initial formant estimates. Final formant values are then produced by imposing frequency continuity constraints using a dynamic programming algorithm over a sequence of frames.

The pitch tracking algorithm used by `get_f0` is described in [27]. The algorithm uses cross-correlation over the analysis window of the current frame to estimate the frequency of voicing. As with the formant tracking algorithm, the F_0 estimate is then modified to fit surrounding values using a dynamic programming step. For each frame, the output of the `get_f0` command is a final estimate of F_0 for that frame, and a secondary value which indicates the probability of voicing for that frame. The probability of voicing returned is either 0 or 1.

6.2.2 Performance of MFCC Feature Set

In this section, we describe results of experiments we performed in order to determine speaker identification performance using the standard MFCC feature set in matched and mismatched training and testing conditions. Speaker models were trained using the baseline GMM speaker modeling approach from Chapter 3. First, closed set identification experiments were performed using matched conditions, with TIMIT speaker models trained on TIMIT training data, and NTIMIT speaker models trained on NTIMIT training data. Next, the same experiment was performed with NTIMIT speaker models trained on TIMIT training data, and vice versa. The results of these identification experiments are shown in Table 6-1.

On the TIMIT data, 100% identification accuracy was obtained using models trained in matched conditions, while on the NTIMIT data, only 53.7% identification accuracy was obtained. These results indicate that MFCC features have high speaker distinguishing capability in clean speech, but that these features can not be reliably extracted in noisy and bandlimited speech. The accuracy rates for the same experiments in mismatched condition underscore this observation. For speaker models trained on TIMIT data, testing on the NTIMIT corpus resulted in an accuracy rate of 0.6%. This experiment illustrated that MFCC features extracted for the noisy speech data were severely distorted from the features extracted from the clean version of the same data.

	Identification Accuracy (%)	
Corpus	TIMIT (Test)	NTIMIT (Test)
TIMIT (Train)	100.0	1.2
NTIMIT (Train)	0.6	53.7
Chance	0.6	0.6

Table 6-1: Identification accuracy of MFCC features on TIMIT and NTIMIT. The first row indicates performance using models trained on TIMIT data. The second row indicates performance using models trained on NTIMIT data. The third row indicates expected performance of a random classifier.

6.2.3 Performance of Formant and Pitch Feature Sets

In this section, we describe results of experiments we performed in order to determine speaker identification performance using formant locations and F_0 as front end features.

Training

For these experiments, ESPS was used to compute formant and pitch estimates at one millisecond intervals. Frame level feature vectors were then formed by averaging these estimates over ten millisecond frames, then concatenating them with the averages from the preceding and following frames. Concatenation of features from surrounding frames was done in order to model formant and pitch trajectories. Since the formant and pitch estimates were valid only during voiced phones, the probability of voicing produced by `get_f0` was used to determine which frames to keep or discard during training and testing. GMM speaker models were trained using TIMIT training data, and tested on both TIMIT and NTIMIT test sets.

Formant Feature Set Experiments

We first evaluated speaker identification performance in mismatched conditions using two sets of formant frequencies as front-end features. The first set included all four formants, and the second set included only F_1 and F_2. The results from this experiment are summarized in Table 6-2. Although accuracy rates were high for the four formant set when tested on TIMIT, performance was considerably lower when tested on NTIMIT. Surprisingly, on NTIMIT, the feature set using only the first two formants resulted in higher identification accuracy than when using all four formants. This outcome indicated that F_3 and F_4 were less reliably extracted from the NTIMIT data than F_1 and F_2, which can be explained by the bandwidth limitation imposed on the NTIMIT data. As discussed in Chapter 2, NTIMIT utterances are produced by passing TIMIT utterances throu a telephone line, which imposes a cutoff frequency at 4 kHz. This lower cutoff frequency eliminates many resonant peaks corresponding to F_4 and can severely attenuate high frequency instances of F_3. A comparison of speech segments from the two corpora illustrating the effect of the cutoff frequency is shown in Figure 6-3

	Identification Accuracy (%)	
Feature Set	TIMIT	NTIMIT
F_1,F_2,F_3,F_4	64.6	4.8
F_1,F_2	24.7	9.9

Table 6-2: Identification accuracy using feature vectors derived from formant sets using GMM speaker models trained on TIMIT data

As a followup to the previous experiment, we also performed identification experiments using each individual formant as separate feature set. These results, which are summarized in Figure 6-4 and Table 6-3, show that the degradation in identification accuracy when going from TIMIT to NTIMIT is more severe for those systems using F_3 and F_4 input features. This fact supports the idea that extraction of the higher frequency formants is less robust in band-limited and noisy channel conditions.

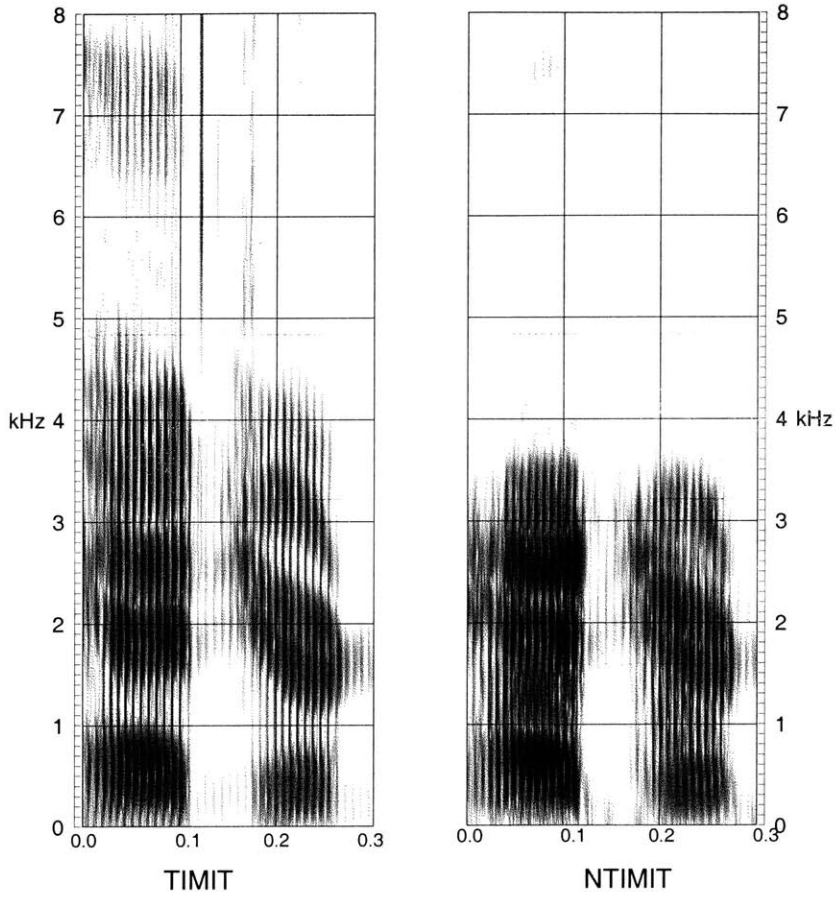

Figure 6-3: Spectrogram comparison of identical speech segments taken from TIMIT and NTIMIT. The NTIMIT utterance has added noise and is bandlimited to 4 kHz.

When training and testing on TIMIT, we noted that the F_3 and F_4 feature sets resulted in signficantly higher accuracy rates when compared to F_1 and F_2. This outcome is not surprising because F_3 and F_4 typically exhibit less intraspeaker variability than the first two formants, which tend to be more dependent on the type of phone being produced. As a consequence, F_3 and F_4 are generally more reliable speaker specific features for speaker recognition than F_1 and F_2.

	Identification Accuracy (%)		Average Rank of Correct Speaker	
Formant	TIMIT	NTIMIT	TIMIT	NTIMIT
F_1	10.7	9.0	9.5	11.7
F_2	9.2	4.8	10.2	13.4
F_3	18.8	6.0	6.5	14.0
F_4	26.8	3.0	4.9	15.0

Table 6-3: Identification accuracy and rank error metric of individual formants using GMM speaker models trained on TIMIT data

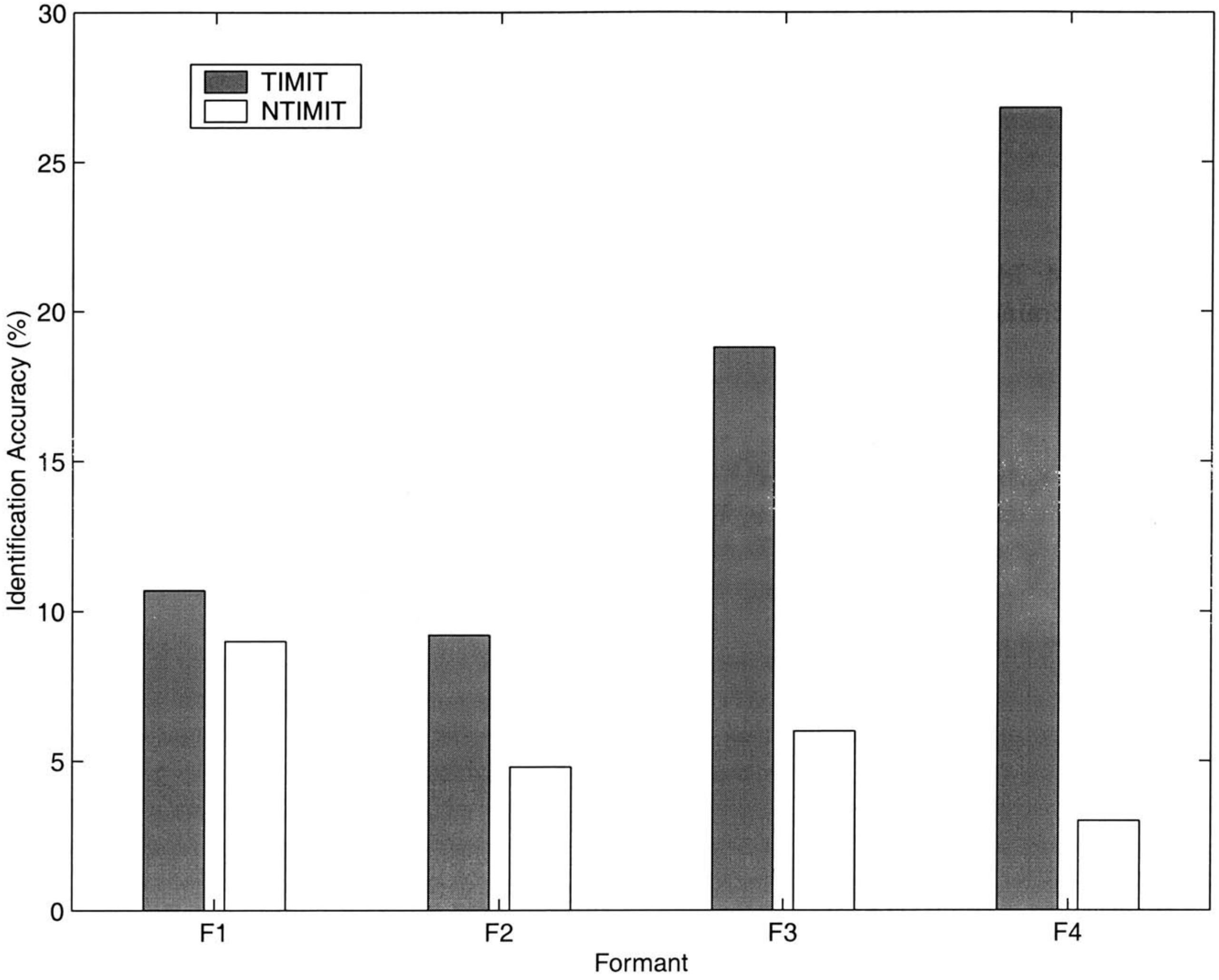

Figure 6-4: Identification accuracy of individual formants using GMM speaker models trained on TIMIT data.

Pitch Feature Set Experiments

In the final set of experiments we performed, we used input feature vectors composed of only F_0 measurements. These results are shown in Table 6-4. We observed that the F_0 based feature set produced lower accuracy rates than the full formant set when tested on the TIMIT data. However, in contrast to the formant set, the F_0 based feature set also exhibited a much smaller performance degradation when tested on NTIMIT under the mismatched condition. This result indicates that estimation of F_0 may be more robust to noise and band-limited channel conditions.

	Identification Accuracy (%)	
Feature Set	TIMIT	NTIMIT
F_0	45.5	39.1

Table 6-4: Identification accuracy using F_0 derived feature vector using GMM speaker models trained on TIMIT data

Chapter 7

Conclusion

7.1 Summary

The work in this thesis has explored three areas for improvement in the field of speaker recognition: speaker modeling, score computation for verification, and front end feature extraction.

In the area of modeling, we compared two baseline text-independent global speaker modeling approaches against two ASR-dependent approaches which make use of speech recognition during identification. The ASR-dependent methods included a novel speaker adaptive approach for scoring with phone-level speaker models. We evaluated the performance of each method on a closed set speaker identification task for two characteristically different corpora, YOHO and MERCURY.

By comparing results on these data sets, we observed that there are a number of factors which can severely degrade identification accuracy. Although the channel bandwidth was the same for each corpus, the MERCURY data also consisted of variable channel and background noise conditions. These acoustic conditions partially explain the large differences in error rates achieved by the various modeling approaches on the two corpora. Aside from differences at the acoustic level, the lexical properties of the two data sets also contributed to the performance gap. While the utterances from the YOHO corpus consisted of read speech from a small set of possible words, the MERCURY utterances consisted of spontaneous speech from a medium-size vocabulary. These lexical differences led to higher word error rates for speech recognition on MERCURY when compared to YOHO, which in turn contributed to higher identification error rates for the ASR dependent approaches, which rely on an accurate phonetic hypothesis for the utterance.

When comparing the different approaches on single utterance experiments, we observed little difference in performance between the baseline systems and the more refined ASR-dependent approaches. However, we demonstrated that classifier combination was effective for reducing identification error rates on single utterances. On the YOHO corpus, we used this technique to achieve a lower error rate than the best reported result for the closed set identification task.

Additional experiments on MERCURY showed that the speaker adaptive approach significantly outperformed the other individual methods when tested using multiple utterances. From this set of experiments, we concluded that phone-level speaker models are better able to capture speaker specific characteristics than global models, but are also more susceptible to identification error when there is only a small amount of test data upon which to base a decision. This result indicates that speaker adaptive scoring has the potential to outperform traditional ASR independent approaches, and is most effectively utilized when performing identification on longer test utterances, or sessions comprised of several utterances.

In the area of speaker verification, we introduced a new technique for computing verification scores which incorporated multiple verification features from the list of scores for a target speaker's background speaker set. Using global GMM speaker models, this approach was compared to the baseline logarithmic likelihood ratio verification score, and exhibited no significant improvement in verification performance. One possible reason for this result is that the baseline scoring technique may produce a near optimal combination of the investigated verification features. Another, more likely reason for the lack of improvement, is that linear discriminant analysis is not a powerful enough technique for feature combination. Further work is necessary to determine whether or not the combination of multiple verification features can actually improve verification performance.

Finally, in the area of front end feature selection, we used the TIMIT and NTIMIT data sets to compare the reliability of feature extraction for MFCCs, formants, and fundamental frequency in bandlimited, telephone quality speech. When trained and tested on clean, wideband speech, the system using the MFCC feature set obtained 100% identification accuracy. Of individual alternative features, F_0, F_3, and F_4 exhibited the greatest speaker distinguishing capability, with accuracy rates of 45.5%, 18.8%, and 26.8%, respectively. Under mismatched conditions, where training was performed on TIMIT and testing was done on NTIMIT, the system using the MFCC feature set showed the greatest performance degradation, achieving 0.6% identification accuracy. Under the same training and testing conditions, accuracy rates for the systems using F_3 and F_4 as features degraded to 6.0% and 3.0%, respectively. The performance of the F_0 based classifier, on the other hand, degraded only slightly, resulting in an accuracy rate of 39.1% on NTIMIT.

In Chapter 6, we hypothesized that the large performance degradation of the F_3 and F_4-based systems was primarily due to the proximity of F_3 and F_4 to the 4 kHz cutoff frequency present in the NTIMIT data. This band-limited condition prevented these formants from being reliably estimated because the higher frequency components, including the resonant peaks corresponding to these formant locations, were attenuated by the low-pass channel filter. The speech signal characteristics required to estimate F_0, on the other hand, were not as adversely affected by the differences between TIMIT and NTIMIT. These alternative feature set experiments demonstrated that F_0 has some degree of speaker distinguishing capability, and can be more reliably extracted amidst additive noise and channel variation than MFCC features, which are currently used in the majority of speaker recognition systems.

7.2 Future Work

In Chapter 5, linear classification techniques were used to combine different sources of information derived from background speaker score lists. Although these techniques showed no significant improvement over the conventional baseline system, which uses a logarithmic likelihood ratio as the verification score, we believe that combination of verification features has the potential to improve overall verification performance given a better feature set and better classification techniques. Some possible non-linear classifiers include multi-layer perceptrons and support vector machines. In addition, we plan to compare the verification performance of systems using modeling approaches other than the baseline GMM method, which was the basis for the experiments in Chapter 5.

Due to time constraints, we were unable to fully investigate novel ways of modeling the feature sets discussed in Chapter 6. Because formant locations, especially F_1 and F_2, are dependent on vowel type, identification accuracy could be improved by using one of the non-global ASR-dependent modeling techniques described in Chapter 3. In particular, with the limited amount of training data present in TIMIT and NTIMIT, the speaker adaptive approach might be an ideal way to model the speaker variability of formant positions for specific vowels.

Besides formants and fundamental frequency, we would also like to perform experiments which test the identification performance of additional features such as duration. One possible way of doing this would be to integrate speaker-dependent duration models into the speaker adaptive scoring framework.

Finally, in addition to performing experiments to determine the viability of using different noise robust features, a number of implementation details for the current feature set also need to be addressed. Since formant tracking and pitch tracking are both performed off-line, a reliable way of integrating these algorithms into the real-time system is desirable. Though robust on-line formant tracking is still a difficult research problem, an accurate on-line pitch tracking algorithm has been developed which can be integrated into SUMMIT [29]. In the future, we would like to incorporate this pitch tracker into the current speaker recognition system.

Bibliography

[1] W.D. Andrews, M.A. Kohler, and J.P. Campbell. Phonetic speaker recognition. In *Proc. Eurospeech*, pages 2517–2520, Aalborg, September 2001.

[2] Christopher J. C. Burges. A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery*, 2(2):121–167, June 1998.

[3] J.P. Campbell. Testing with the YOHO CD-ROM voice verification corpus. In *Proc. ICASSP*, pages 341–344, Detroit, May 1995.

[4] U. Chaudhari, J. Navrátil, and S. Maes. Transformation enhanced multi-grained modeling for text independent speaker recognition. In *Proc. ICSLP*, volume 2, pages 298–301, Beijing, October 2000.

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

[6] G.R. Doddington. Speaker recognition - Identifying people by their voices. *Procedings of the IEEE*, 73(11):1651–1663, November 1985.

[7] G.R. Doddington. Speaker recognition based on idolectal differences between speakers. In *Proc. Eurospeech*, pages 2521–2524, Aalborg, September 2001.

[8] R. O. Duda and P. E. Hart. *Pattern Classification and Scene Analysis.* John Wiley & Sons, New York, NY, 1973.

[9] J.P. Eatock and J.S. Mason. A quantitative assessment of the relative speaker discriminant properties of phonemes. In *Proc. ICASSP*, volume 1, pages 133–136, Adelaide, April 1994.

[10] R. Faltlhauser and G. Ruske. Improving speaker recognition performance using phonetically structured Gaussian mixture models. In *Proc. Eurospeech*, pages 751–754, Aalborg, September 2001.

[11] S. Fine, J. Navratil, and R. Gopinath. Enhancing GMM scores using SVM hints. In *Proc. Eurospeech*, pages 1767–1760, Aalborg, September 2001.

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

[13] J. Glass, J. Chang, and M. McCandless. A probabilistic framework for feature-based speech recognition. In *Proc. ICSLP*, pages 2277–2280, Philadelphia, October 1996.

[14] Y. Gu and T. Thomas. A text-independent speaker verification system using support vector machines classifier. In *Proc. Eurospeech*, pages 1765–1768, Aalborg, September 2001.

[15] T.J. Hazen, S. Seneff, and J. Polifroni. Recognition confidence scoring and its use in speech understanding systems. *Computer Speech and Language*, 16(1):49–67, January 2002.

[16] I.L. Hetherington. An efficient implementation of phonological rules using finite-state transducers. In *Proc. Eurospeech*, pages 1599–1602, Aalborg, September 2001.

[17] C.R. Janowski Jr., T.F. Quatieri, and D.A. Reynolds. Measuring fine structure in speech: Application to speaker identification. In *Proc. ICASSP*, pages 325–328, Detroit, May 1995.

[18] J. Kharroubi, D. Petrovska-Delacrétaz, and G. Chollet. Combining GMM's with support vector machines for text-independent speaker verification. In *Proc. Eurospeech*, pages 1761–1764, Aalborg, September 2001.

[19] A. Martin, G. Doddington, T. Kamm, M. Ordowski, and M. Przybocki. The DET curve in assessment of detection task performance. In *Proc. Eurospeech '97*, pages 1895–1898, Rhodes, Greece, September 1997.

[20] H.A. Murthy, F. Beaufays, L.P. Heck, and M. Weintraub. Robust text-independent speaker identification over telephone channels. *IEEE Trans. on Speech and Audio Processing*, 7(5):554–568, September 1999.

[21] D. Petrovska-Delacrétaz, J. Černocký, J. Hennebert, and G. Chollet. Segmental approaches for automatic speaker verification. *Digital Signal Processing*, 10:198–212, January 2000.

[22] M. Powell. An efficient method of finding the minimum of a function of several variables without calculating derivatives. *The Computer Journal*, 7:155–162, 1964.

[23] L. Rabiner and B.-H. Juang. *Fundamentals of Speech Recognition.* Prentice-Hall, Englewood Cliffs, NJ, 1993.

[24] D.A. Reynolds. Speaker identification and verification using Gaussian mixture speaker models. *Speech Communications*, 17:91–108, 1995.

[25] S.V. Sarma and V. Zue. A segment-based speaker verification system using SUMMIT. In *Proc. Eurospeech*, pages 843–846, Rhodes, September 1997.

[26] S. Seneff and J. Polifroni. Formal and natural language generation in the MERCURY conversational system. In *Proc. ICSLP*, volume 2, pages 767–770, Beijing, October 2000.

[27] D. Talkin. A robust algorithm for pitch tracking (RAPT). In W. B. Kleijn and K. K. Paliwal, editors, *Speech Coding and Synthesis*, pages 111–121. Elsevier, New York, 1995.

[28] O. Thyes, R. Kuhn, P. Nguyen, and J.-C. Junqua. Speaker identification and verification using Eigenvoices. In *Proc. ICSLP*, volume 2, pages 242–245, Beijing, October 2000.

[29] C. Wang and S. Seneff. Robust pitch tracking for prosodic modeling in telephone speech. In *Proc. ICASSP*, pages 887–890, Istanbul, Turkey, June 2000.

[30] P.H. Winston. *Artificial Intelligence.* Addison-Wesley, Reading, MA, third edition, 1992.

[31] V. Zue, J. Glass, D. Goodine, M. Phillips, and S. Seneff. The SUMMIT speech recognition system: Phonological modeling and lexical access. In *Proc. ICASSP '90*, pages 49–52, Alburquerque, NM, April 1990.