This is to certify that the

dissertation entitled

LANGUAGE IDENTIFICATION USING
GAUSSIAN MIXTURE MODELS

presented by

Pedro A. Torres-Carrasquillo

has been accepted towards fulfillment
of the requirements for

PH.D. degree in Elec. & Comp. Engineering

John R. Deller Jr.
Major professor

Date 28 June 2002

Language Identification Using Gaussian Mixture Models

By

Pedro A. Torres-Carrasquillo

A DISSERTATION

Submitted to

Michigan State University

in partial fulfillment of the requirements

for the degree of

DOCTOR OF PHILOSOPHY

Department of Electrical Engineering

2002

ABSTRACT

Language Identification Using Gaussian Mixture Models

By

Pedro A. Torres-Carrasquillo

With the increasing globalization of speech information access and interfaces, applications of automatic language identification (LID), identifying the language of a spoken utterance, are increasing. These applications include routing of callers to operators who speak their language and selecting language-dependent speech recognizers. Current state of the art systems rely on phoneme recognition followed by n-gram language modeling for performing the identification task and have high recognition accuracy. However, phoneme recognition can be computationally burdensome and difficult to rapidly adapt for some applications. In this work, we examine a more general and computationally efficient alternative using Gaussian Mixture Models (GMM) for capturing both acoustic and phonetic structure information. In the system developed, the state sequence tokenization of the speech features through the GMM is used to replace the phoneme recognition tokenization. Additionally, the acoustic match score of the speech features to the GMM, which comes at no additional computational cost, is used with the state sequence tokens to further improve performance. Performance is obtained for the CallFriend corpus resulting in a 12-way closed error rate of 18.6% compared to 21.5% for the state of the art phoneme recognition system.

ACKNOWLEDGEMENTS

First of all, thanks go to my advisor Dr. John R. Deller for his support. He provided the initial drive to pursue this area and guided me in the best path possible. Thanks also for keeping up with my constant questions and drafts while I work remotely.

I am deeply grateful to my supervisor at MIT Lincoln Laboratory, Dr. Doug Reynolds. His comments were invaluable through the development of this work. Thanks also for always been available.

To the people in the SPC3 program for giving me support for the last four years. In particular, Dr. Percy Pierre and Dr. Barbara O'Kelly for developing a wonderful environment within the EE Dept to ease the struggles of graduate school.

My thanks also go to the people in Group 62 at MIT Lincoln Laboratory for their support and guidance. Special thanks to Elliot Singer and Marc Zissman for their comments on how to constantly improve this work.

To my family and Beba for their understanding, and to my grandfather, Pedro, for providing the inspiration that kept me going for all these years, thanks!

Chapter 1 Introduction 1

1.1 Introduction and Motivation 1

Chapter 2 Previous Work 6

2.1 Acoustic Processing and Prosody Based Techniques 6

2.2 Phonetic Techniques 8

2.3 Large Vocabulary Continuous Speech Recognition Based Language Identification 14

2.4 Discussion of Previous Work 15

Chapter 3 Mathematical Methods and Algorithms 18

3.1 General Description of the Language Identification System 18

3.2 Front-end Processing 20

3.3 Gaussian Mixture Model 21

3.4 Language Modeling 25

3.5 Backend Classifier 27

3.6 Summary 29

Chapter 4 Long-Term Approaches to Speech Tokenization 30

4.1 Motivation 30

4.2 Temporal Encoding 31

4.3 Multigram Analysis 32

4.4 Vector Quantization 36

4.5 Shifted-Delta-Cepstra (SDC) Features 40

4.5.1 SDC Parameterization 43

Chapter 5 Experiments and Results 47

5.1 Corpus Description 47

5.2 Experimental Framework 49

5.3 Baseline GMM System 51

5.4 GMM Systems with Long-Term Acoustics 54

5.4.1 Temporal Encoding 55

5.4.2 Multigrams 57

5.4.3 Vector Quantization 60

5.5 GMM System Revisited 62

5.5.1 Backend Classifier 62

5.5.2 Parallel GMM System 66

5.5.3. Parallel GMM System with Acoustic Scores Fusion 70

5.6 SDC Parameterization Experiments 72

5.7 SDC-based Parallel GMM System 75

5.8 Evaluation of the SDC-based Parallel GMM System with the OGI Corpus 77

5.9 Analysis of the Confusion Matrix for the CallFriend evaluation set 83

5.10 Miscellaneous Experiments 85

Chapter 6 Summary, Conclusions and Future Work 87

6.1 System Goals and Achievements 87

6.2 Advantages 88

6.3 Disadvantages 89

6.4 Future Work 90

6.5 Contributions 91

Appendix A 93

Appendix B 96

Appendix C 98

Appendix D 102

LIST OF FIGURES

FIGURE 1-1. Typical architecture of a language identification system................................ 3

FIGURE 2-1. Block diagram of Zissman's Phoneme Recognition followed by Language Modeling (PRLM) single tokenizer system... 11

FIGURE 3-1. Block diagram of the new LID system based on GMM tokenization. 19

FIGURE 3-2. Front-end speech processing system.. 20

FIGURE 3-3. Mel-scale filtering operation.. 21

FIGURE 3-4. Single language GMM tokenization process. 23

Figure 4-1. LID systems with the inclusion of long-term processing techniques............. 31

FIGURE 4-2 Block diagram for the computation of the shifted-delta cepstra coefficients. 43

FIGURE 5-1. Baseline single tokenizer GMM system. 52

FIGURE 5-2. Single tokenizer GMM system with temporal encoding. 55

FIGURE 5-3. Error rates for the single tokenizer GMM system with VQ processing for the CallFriend evaluation set. Results are shown for various GMM orders. 61

FIGURE 5-4. Average error rates for the single tokenizer GMM system with a backend classifier and different mixture orders for the CallFriend evaluation set. 65

FIGURE 5-5. Parallel implementation of the GMM system using multiple single language tokenizers.. 67

FIGURE 5-6. Parallel implementation of the GMM system with acoustic scores fusion... 70

FIGURE 5-7. Error rates comparison for the parallel system implementation, including systems without and with the fusion of acoustic scores. 71

FIGURE 5-8. Error rates comparison for the SDC-based parallel GMM system implementation, including a system without the fusion of acoustic scores and with fusion of the acoustic scores. 75

FIGURE 5-9. Error rates comparison between the parallel GMM system implementation using SDC features and cepstral features. 76

FIGURE 5-10. Error rates for the OGI testing set using the SDC-based parallel GMM system. 79

FIGURE 5-11. Comparison of the performance of the system evaluation for the 45-second set of the OGI corpus under different classifier training alternatives. 80

FIGURE 5-12. Comparison of the performance of the system evaluation for the 10-second set of the OGI corpus under different classifier training alternatives. 80

FIGURE 5-13. Error rates comparison for the parallel system using the OGI 45-second set with leave-one-out evaluation and higher mixture acoustic scores fusion. 82

FIGURE 5-14. Error rates comparison for the parallel system using the OGI 10-second set with leave-one-out evaluation and higher mixture acoustic scores fusion. 82

FIGURE A-1. SDC-based parallel GMM system using 256 mixtures. 93

FIGURE A-2. SDC-based parallel GMM system using 128 mixtures. 94

FIGURE A-3. SDC-based parallel GMM system using 64 mixtures. 94

FIGURE A-4. SDC-based parallel GMM system performance as a function of the GMM order. 95

FIGURE B-1. SDC-based parallel GMM system performance of the system on the 3-second segments of the CallFriend evaluation set. 96

FIGURE B-2. SDC-based parallel GMM system performance of the system on the 10-second segments of the CallFriend evaluation set. 97

FIGURE C-1. SDC-based parallel GMM system performance of the system on the 30-second segments of the CallFriend corpus using OGI training data for the GMM tokenizer 98

FIGURE C-2. SDC-based parallel GMM system performance of the system on the 30-second segments of the CallFriend corpus using CallHome training data for the GMM tokenizer 99

FIGURE C-3. SDC-based parallel GMM system performance of the system on the OGI corpus test set using CallFriend training data for the GMM tokenizer 100

FIGURE C-4. SDC-based parallel GMM system performance of the system on the OGI corpus test set using CallHome training data for the GMM tokenizer. 101

FIGURE D-1. SDC-based parallel GMM system performance of the system as different combinations of tokenizers are used. The plot shows the system best, average and worst combination of tokenizers results. 102

LIST OF TABLES

TABLE 2-1. Comparison of the information requirements for different approaches to LID. 17

TABLE 4-1. Description of the multigram training algorithm. 35

TABLE 4-2. Steps of the general vector quantization algorithm. 37

TABLE 4-3. Description of the modified vector quantization algorithm. 39

TABLE 4-4. Phoneme durations for the labeled languages in the OGI corpus. 45

TABLE 5-1. Error rates for the single tokenizer GMM system for the CallFriend evaluation set as a function of the tokenizing language and the mixture order. A not available (N/A) note is used for results that can be determined given that phonemic transcriptions are not available in those languages. 53

TABLE 5-2. Error rates for the single tokenizer GMM system with temporal encoding for the CallFriend evaluation 56

TABLE 5-3. Error rates for the single tokenizer GMM system with different multigram lengths and mixture order for the CallFriend evaluation set. 59

TABLE 5-4. Error rates for the single tokenizer GMM system with and without a Gaussian backend classifier for CallFriend evaluation set 64

TABLE 5-5. Error rates for the parallel GMM system with conventional cepstral features for the CallFriend evaluation set. 69

TABLE 5-6. Error rates for the CallFriend evaluation set using SDC parameter $d = 1,2$ and 4. 73

TABLE 5-7. Error rates for the CallFriend evaluation set for SDC parameter k= 2, 3, 4, 5 and 6. 73

TABLE 5-8. Error rates for the CallFriend evaluation set for SDC parameter N=6, 8 and 10. 74

TABLE 5-9. Confusion matrix for the SDC-based parallel system using the CallFriend evaluation set. 84

Chapter 1

Introduction

1.1 Introduction and Motivation

Automatic language identification (LID) is the process of using a computer to identify the language of a spoken utterance. Over the last decade, LID has received attention from the research community as economies become more global and speech recognition systems become viable in commercial applications.

Practical applications of LID systems are emerging in areas such as banking, telephony, transportation, information services, and as part of more intelligent speech systems. In the area of telephony, in particular emergency systems like the 9-1-1 line[1], an application is being used in the United States to efficiently route phone calls to operators who speak the preferred language of a caller [1].

Similar applications are anticipated in other areas. For example, an international bank could have a centralized operator system to route client calls to a human operator who speaks the identified language. Airport information services could be automated by having computerized systems to give information to clients in a desired language. A LID

[1] The 9-1-1 telephone number is widely used in the continental United States to report emergencies.

system could also be a vital component of self-contained automatic spoken language translation systems, in which identifying the input and output languages could be considered a first step.

Other applications of interest occur in military tasks. A system than can correctly identify a spoken language can be used to obtain information on the adversary by identifying his language and using this identification to dispatch a translator to gather vital information. As more interaction takes place among the countries of the world, other applications for LID systems are expected to emerge.

Perceptual studies in the LID area have shown humans to be adept to the task [1]. Studies in the area of linguistics have evaluated the use of rare segments for discrimination among the world languages [2]. The study on rare speech segments emphasized the tradeoff between discriminatory capabilities of different language features and the ability to estimate these features robustly. The techniques of choice for implementation for computer-based systems have focused on three major areas. These areas include: acoustics and prosody, phonetics and phonotactics, and vocabulary.

The acoustic and prosody approach have focused on techniques for evaluating spectral similarities and elements such as intonation and rhythm. The phonetic and phonotactic approaches rely on the phonemic inventories and phoneme sequences of the languages while the vocabulary-based techniques deal with large-vocabulary continuous speech recognition.

Most existing LID techniques have focused on acoustic techniques and phoneme-based techniques. A block diagram of the general approach to LID is shown in Fig. 1-1. The acoustic technique approach relies on capturing the frame-by-frame acoustics and computing likelihoods of an utterance for each model. The phoneme-based techniques segment the test speech into a sequence of phonemes, then construct language models out of the decoded sequences of phonemes. The obtained language models typically consist of the statistics observed in the decoded sequence of phonemes.

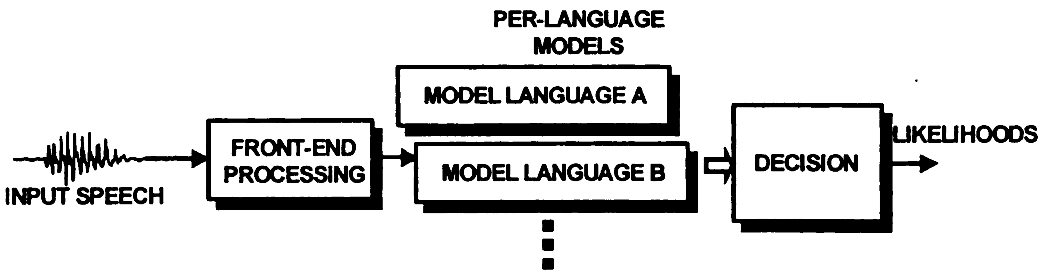

FIGURE 1-1. Typical architecture of a language identification system.

The state of the art systems employing phonotactics, phoneme inventories and sequencing, are limited to applications that can afford the high computational cost required for the identification task. The techniques presented in this work are aimed at reducing the computational complexity of the LID system while obtaining performance competitive with state of the art systems. An additional advantage of the system introduced in this work is the elimination of the need for phonetic or phonemic transcriptions of the speech.

The research presented in this dissertation introduces a system that combines the tokenization approach, where tokenization refers to the process of converting a feature vector into one of a set of discrete elements, used in phoneme-recognition systems with the acoustic modeling of the purely acoustic systems. Rather than relying on a phonemic set for tokenization, the systems studied here use information derived directly from the speech waveform to model the speech acoustics. The presented system also incorporates temporal information about the speech by constructing language models from the tokenized speech, similarly to the phonemic-recognition based systems.

As shown in Chapter 5, the Gaussian Mixture Model (GMM) tokenization system presented in this work outperforms the current state-of-the-art phonotactic technique on the CallFriend corpus while reducing computations by 66%. Results also indicate the further improvement in accuracy by combining multiple sources of information from the speech signal.

The outline of this dissertation is as follows: Chapter 2 presents a description of previous work and discusses some of the advantages and disadvantages of the major techniques. Chapter 3 presents the proposed system with the mathematical description of the techniques used for modeling the speech acoustics. Chapter 4 presents a description of a series of methods aimed at analyzing the long-term characteristics of the speech to derive units that can be used for the tokenization of the speech and eventually to characterize the languages. Chapter 5 presents the experimental framework used to evaluate the LID system. Chapter 6 presents the conclusions highlighting the advantages and disadvantages of the LID system proposed and a comparison between the results obtained and state of the art systems.

An Appendix presents additional experimental results. These results include the performance of the shifted-delta-cepstra (SDC) based parallel GMM[2] system under different conditions and the performance of this system for speech utterances of different durations.

[2] The SDC features and the Gaussian Mixture Model are important concepts that are central to this research. These terms are carefully defined in following discussions.

Chapter 2

Previous Work

Research on LID has been based principally on three approaches: acoustic and prosody based techniques, phonemic approaches which use phoneme occurrences and phoneme sequences to discriminate across languages, and more sophisticated systems which use sub-word models and speech recognition systems. The discussion below examines techniques from each of these approaches, noting advantages and disadvantages of each.

2.1 Acoustic Processing and Prosody Based Techniques

Research by Sugiyama [3] uses acoustic features in a vector quantization codebook approach. A set of cepstral features [4] is obtained for utterances of training speech from each language, and then a language specific codebook is created. The language specific codebook consists of a collection of cepstral feature vectors, obtained from the training data, which characterizes each of the languages. The identified language is chosen, from among 20 candidates, as the language closest to the candidate codebooks. In this case, a distortion metric is employed to determine the distance between the test utterance and the candidate languages. A technique by Li [5] uses syllabic spectral features and nearest neighbor distances to discriminate among five languages. The study shows how certain syllable-based information can yield language-identifying information. Li's technique is later enhanced by adding prosody features such as normalized pitch, amplitude and timing [6].

Another technique due to Itahashi and Du [7] uses fundamental frequency information and speech energy to classify languages. In this case, statistics of these measures such as standard deviation, and skewness and kurtosis, are used. The space of statistical features is studied using discriminant analysis, and then the eigen-features are used for classification. The initial study is later expanded by using ergodic hidden Markov models (HMM) [8] to model acoustic event sequences in combination with the fundamental frequency analysis [9].

A study by Savic [10] combines the prosodic approach with an ergodic HMM [4]. In this case, pitch contours are analyzed by characterizing their distribution over frequency and their rates of change. A five-state ergodic HMM is used to model each language to be identified. The results are then combined by a voting classifier, and a language is chosen based on this final classifier.

Hutchins [11] employs other prosody-based features, including pitch and amplitude information from a syllable-by-syllable analysis, in a pairwise classification approach. Cummins [12] generalizes the study of Hutchins by using information extracted from the fundamental frequency and the first difference in the amplitude envelope. In this case, rather than evaluating different features in a statistical discrimination framework, the two features are used as input to a recurrent neural network [13]. The neural network processes this information during training, and hypothesizes a language when subsequent test data are presented.

In a study on discrimination of Arabic dialects, Barkat uses prosody as the discrimination feature in perceptual studies [14]. Barkat's listeners had access only to characteristics from the speech signal, which included fundamental frequency, amplitude, and rhythm features.

An approach due to Pellegrino [15] uses vowel system modeling on a five-language discrimination task. In this study, a GMM is trained using an unsupervised language-independent technique to model the vowels [16]. For each language of interest, such a model is used for maximum likelihood identification.

The systems presented in this first section represent one class of approaches used for LID. In general, the systems in this class do not perform as well as systems to be described below. The systems in this section provide lower correct classification rates than other systems, but do so without using labeled data which are required by the next class of systems.

2.2 Phonetic Techniques

Another class of techniques used for LID relies on the use of phoneme statistics as the basis for decisions. In this approach, phonemic occurrences and phonemic sequencing, together known as *phonotactic structure*, are used to discriminate languages. In most instances of this class of systems, a phonemic segmentation is employed and a language model (LM) is constructed based on the recognized phonemes. The hypothesized language is the one with the highest LM probability.

One of the first studies in phonotactics was presented by House and Neuburg [17]. In this study, languages are identified based on their phonemic inventories and sequencing. The study is the first to describe the use of phoneme sequencing as a tool for language discrimination, and includes results for the technique on artificially labeled data.

A study by Lamel [18] represents one of the first initiatives to employ phonotactics for LID using field collected data. In Lamel's experiment, a two-language discrimination task is based on an ergodic HMM for each language. Each ergodic HMM is built from smaller left-to-right phonetic HMMs. The hypothesized language for a test utterance is based on the highest acoustic likelihood.

Other approaches base classification schemes on the differences and similarities of phonemes across languages. In Berkling's study [19], for example, phonemes are classified according to acoustic similarities across the languages. Phonemes whose acoustics are similar across languages are labeled as *poly-phonemes* while phonemes that are acoustically unique for a language are labeled as *mono-phonemes*. A pairwise classification task is studied using features derived from both classes of phonemes and the contribution of the different features is assessed.

A hybrid technique due to Hazen [20, 21] combines information from phonotactics, prosody and acoustics. In Hazen's work, a phonotactic model, an acoustic model and a prosodic model are based on segmenting the training data into broad phonetic classes. The phonotactic model classifies segments into broad phonetic classes and creates a LM using bigram statistics [22, 23]. The prosodic model is constructed from statistics of segment durations and fundamental frequency contours. The acoustic model is derived from a probability density function deduced from segments of each class. The model scores are integrated by combining the individual likelihoods.

Zissman [24] proposes a system that also segments speech into phonemic units which are then used in a LM similar to Hazen's. Zissman's system, named *phoneme recognition followed by language modeling* (PRLM), is shown in Fig. 2.1. PRLM segments the phonemic units in finer detail than those of Hazen. A phoneme recognizer is trained using phonemically labeled speech. Each training utterance is parsed based on the phonemic recognizer and a bigram-based LM is created. During testing, each test utterance is parsed using the phoneme recognizer and scored against a LM for each of the candidate languages.

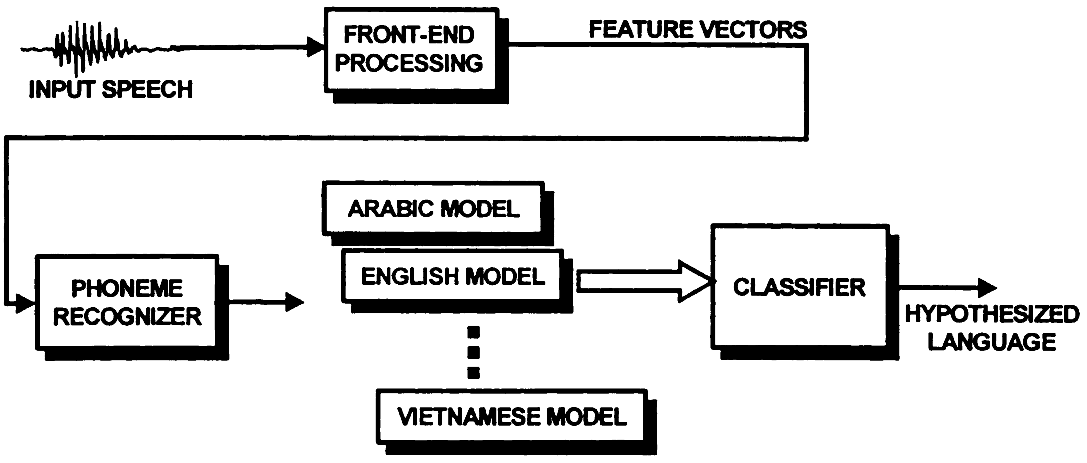

FIGURE 2-1. Block diagram of Zissman's Phoneme Recognition followed by Language Modeling (PRLM) single tokenizer system.

The original PRLM technique has been enhanced by the use of multiple PRLM systems in parallel and by incorporating gender information and phoneme-duration information [25]. In the enhanced system, a single tokenizer PRLM system is constructed for each of the languages for which a phonemic transcription is available. The LM scores resulting from each tokenizer system are fed into an output classifier for final classification. The duration information is incorporated into the system by classifying each recognized phoneme as either short or long when compared to the average phoneme duration across all languages. The parallel system, known as PPRLM, consists of six tokenizer systems in parallel, each producing 12 LM scores for a total of 72 scores. This approach represents the current state of the art, yielding a 21.5% error rate on the CallFriend corpus [26]. The CallFriend corpus will be described in Chapter 5.

Yan [27] presents a technique that is similar to PRLM but which incorporates a backward bigram model, in addition to the forward bigram model used by Zissman. Yan also incorporates duration analysis in the language modeling. Yan's work is extended by adding an optimization technique for modeling the language and increasing the number of languages in the original study to 11 languages [28]. In a third study [29], Yan employs the technique on a different database of languages comprised of informal telephone conversations. The first two studies by Yan are conducted using the Oregon Graduate Institute (OGI) corpus [30] which consists mainly of answers to a series of questions in each collected language. The OGI corpus will be described in more detail in Chapter 5.

Other techniques with PRLM as a core component are presented by Kwan and Hirose [31], Andersen and Dalsgaard [32], and Navratil and Zuhlke [33]. Kwan and Hirose study the problem by building a "mixed phoneme" recognizer rather than a language-dependent recognizer. The mixed recognizer is constructed out of a union of language-specific phonemes and language-independent phonemes. This study supplements the LM by building LMs for not only the decoded phoneme sequences but also deriving an additional model from the hand marked transcriptions in the training data.

Andersen and Dalsgaard [32] also use a phoneme merging technique for the LID problem. In this case, an additional step is performed after the LM scores are computed and final classification occurs. Linear discrimination analysis is performed on the LM scores to study relations on these scores that might be useful for discrimination. In subsequent studies, linear discrimination analysis is replaced by the k-nearest neighbor algorithm, Mahalanobis distance measures, and decision trees [34, 35].

Navratil and Zuhlke [33] study language modeling alternatives to bigrams, including a modified bigram technique, and the use of decision trees. The modified bigram technique uses prior information based on phoneme pairs rather than a single phoneme. Typically, the bigram LM is implemented by counting the frequency of occurrence of phoneme B after observing phoneme A. In the modified bigram technique, pairs of phonemes that occur frequently are considered as a single entity and the next observed phoneme is incorporated into the model. This technique is a computationally efficient alternative to trigrams [23]. In theory, trigrams should provide better discrimination than bigrams but require an extensive amount of training data and more computational effort. The modified bigram and the tree technique offer alternatives to trigrams at a lower computational cost. The tree technique is based on an entropy measure and provides better results than the conventional bigram model.

Navratil and Zuhlke [36] introduce yet another alternative for language modeling based on the so-called "skip-grams." The method is similar to that of Kwan in using phonotactic information from both the original labeled speech and the decoded phoneme sequence. Like Navratil's earlier method, skip-grams are suggested as an alternative to trigrams. The skip-grams are constructed by building relations between present phonemes and phonemes two time slots before, rather than the ones in the immediate past. Navratil and Zuhlke [37] present a system that moves toward the use of multiple information sources by combining the phonotactic approach with acoustic based models.

Phonotactic techniques provide the best compromise between performance and computational expense. These systems provide some degree of adaptability but require labeled speech on at least some of the candidate languages, which is difficult to obtain for some applications.

2.3 Large Vocabulary Continuous Speech Recognition Based Language Identification

The third general class of LID systems is based on techniques drawn directly from large vocabulary continuous speech recognition (LVCSR) systems. These systems exhibit performance that is usually higher than that of acoustic or phonotactic-based systems, but at a higher computational cost along with word-level and phonemic transcriptions. Some of the techniques in this class of systems are described below.

The LVCSR approaches to LID rely on using the information obtained by the phonotactic approaches and on incorporating a lexical component [4]. For example, Schultz [38] describes research on LID systems with increasing amounts of lexical information. Schultz reports results on three lexical models. The experiments include word recognition on each language, word bigram language modeling, and a two-stage process combining both methods.

Kadambe and Hieronymus [39] report results on a four-language recognition task. In this work, a stochastic trigram-based LM is constructed to analyze phoneme sequences, and a lexical matching technique is implemented at the output of the LM. Results from all of the parallel recognizers are then classified by Bayesian likelihood analysis. In later work [40, 41], this study is expanded to include a fifth language.

A slightly different technique is employed by Matrouf [42]. For a four-language recognition task, a phoneme recognizer is augmented by the *N* most frequent words. Experiments are performed to determine the effects of lexical coverage. Results are reported on two databases, the first consists of telephone speech, and the second consists of task-oriented read and elicited speech.

LVCSR systems use the largest amount of past information, phonetic and word level information, among the three general classes of systems. The LVCSR systems are computationally burdensome, resulting in the slowest systems of the three classes of systems described. For these systems, phonetically labeled speech and lexical information are both necessary, making it difficult to design a system that can readily adapt to new conditions.

2.4 Discussion of Previous Work

Sections 2.1 through 2.3 presented the main techniques currently employed in LID research. The performance within each class of methods improves as higher levels of phone and lexical information are included into the discrimination process. The performance improvements obtained with higher levels of information also carry higher computational cost associated with the phonemic recognition and word recognition tasks. For each class of systems described above, a general description of advantages and disadvantages is given below.

The class of systems described in Section 2.1 provides the highest degree of flexibility among those presented. These systems do not require *a priori* information for the LID task, but result in the lowest performance among the studied systems. Typically the systems in this class result in correct identification rates around 50%. One of the possible problems with the acoustic and prosody-based approaches is its limited use of "long-term" acoustic information as contrasted with the phonotactic approaches. Acoustic systems rely on information derived from frame-level information instead of combinations of frames.

The second class of techniques, based on phonotactics, is the most popular method among existing LID systems. The phonotactic techniques have yielded the best tradeoff between the need for *a priori* information and the amount of required computation among the studied approaches. Performance levels for the 12-language LID task have resulted in correct classification as high as 75% for the phonotactic approaches. The main drawback is the need for labeled data. Reliance on phonemic labeling results in limited adaptability to new conditions.

The third class of systems based on LVCSR, augments the phonotactic approaches with the use of lexical information. LVCSR has proved the best performer for discrimination problems such as the five-language classification task, resulting in systems yielding correct classification rates as high as 95%. This technique shares the adaptation problems with phonotactic approaches, while requiring more training data.

Class of systems	Required information for training the system
Acoustic / Prosody based	Language identity
Phonetic / Phonotactic	Language identity, phonetic or phonemic transcriptions
LVCSR	Language identity, phonetic or phonemic transcriptions, and word transcriptions

TABLE 2-1. Comparison of the information requirements for different approaches to LID.

Table 2-1 summarizes the techniques currently used for the LID task. The work presented in this dissertation combines the performance advantages of the phonetic and phonotactic approach with the flexibility of the acoustic approaches. As described in Chapter 3, the flexibility of the acoustic approach allows the system to be trained to match the acoustic conditions of the operating environment. This flexibility is not available for the phonotactic system given the requirement for phonetic or phonemic transcriptions.

The chapters that follow present a new technique that attempts to overcome the problems of the phonotactic-based approach while obtaining similar performance.

Chapter 3

Mathematical Methods and Algorithms

This chapter presents a description of the core mathematical methods and algorithms employed in this work. An overview of the LID system is provided, including the basics of each component of the LID system.

3.1 General Description of the Language Identification System

The system developed in this work is motivated by systems relying on phoneme-recognition, particularly the PRLM system [24] described in Chapter 2. The PRLM system is based on decoding an incoming speech stream into a set of phonemes and building a statistical LM of phoneme occurrences and phoneme sequences for each language. The work presented in this chapter and Chapter 4 is directed at alternatives for replacing the phoneme-recognition unit as the tokenization element by a faster and more flexible component.

The LID system studied in this research is shown in Fig. 3.1. The system is similar to that of Zissman [43] which is described in Chapter 2. The system researched here employs data-driven techniques to tokenize the incoming speech. The use of data-driven techniques enables the system to be highly adaptable, allowing for training the system for new acoustic conditions. In addition, these techniques eliminate the need for transcriptions. In contrast, a system such as PRLM will need phonemic transcriptions to be available before retraining is an option.

The core component in the new system is the GMM used for modeling speech acoustics. The main purpose of the GMM is to serve as a speech tokenizer. In turn, the tokenizer is used to obtain a segmentation of the acoustic space into classes with similar intraclass acoustics. The analysis of the *output* of the GMM tokenizer, known as long-term processing and presented in Chapter 4, allows for the incorporation of long-term acoustic information into the LID process. The use of these long-term processing techniques is motivated by the performance obtained by phonotactic systems.

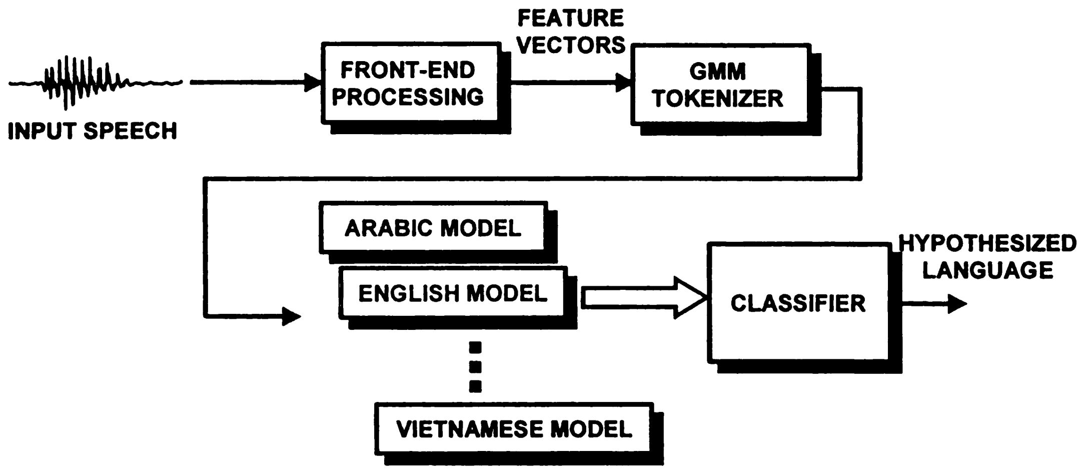

FIGURE 3-1. Block diagram of the new LID system based on GMM tokenization.

The general LID system consists of a pre-processing and feature extraction (front-end) stage, the GMM tokenizer, a language modeling stage, and a backend classifier. A description of each of these stages is given below.

3.2 Front-end Processing

The front-end processing is used to extract features from the incoming speech. This feature extraction process is typically employed in pattern recognition problems to derive an efficient parametric representation of the patterns of interest. A description of the general process is shown in Fig. 3-2. In this research, a set of mel-warped cepstral coefficients [4] is computed at a frame rate of 20 ms with a 10 ms overlap between frames. Speech activity detection is performed to eliminate frames of speech that are considered to contain silence. A Rasta filter [44] processes the coefficients to remove channel effects and short-term mean effects. The final feature set is then composed of the appropriate number of mel-warped cepstral coefficients and delta coefficients [4]. The delta-cepstra coefficients are typically computed by the subtraction of the cepstral coefficients over a fixed interval, usually one or two frames from the frame at time *t*. The use of this set of coefficients for LID is consistent with the methods used in similar research.

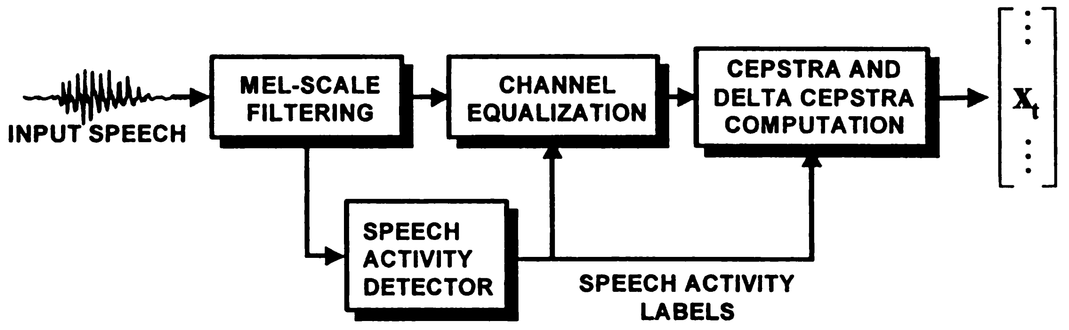

FIGURE 3-2. Front-end speech processing system.

A block diagram of the mel-scale coefficient computation is shown in Fig. 3-3. The mel-scale technique is employed to capture lower-frequency spectral information in greater detail while the higher frequencies are more coarsely represented. Typically the lower frequencies are passed through filters centered at a linearly-spaced set of frequencies, while the frequencies above 1000 Hz are filtered with filters distributed on a logarithmic scale [45].

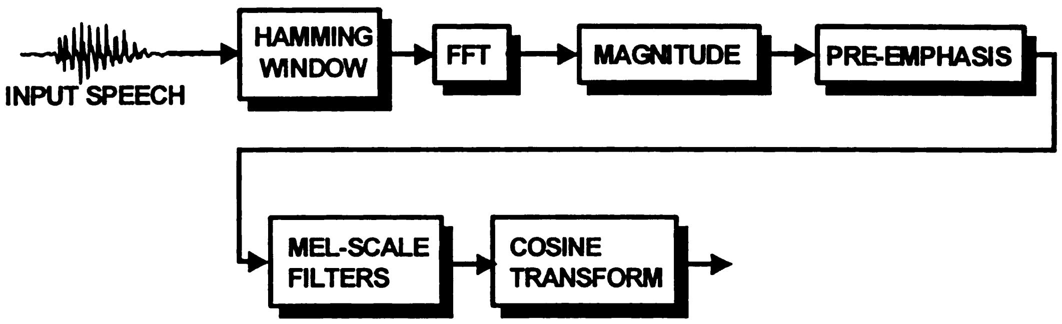

FIGURE 3-3. Mel-scale filtering operation.

3.3 Gaussian Mixture Model

The use of a GMM as a tokenizer is one of the novel contributions of this work. The GMM technique is generally used to model multimodal distributions and it has been used successfully for other speech applications such as speaker identification [46, 47]. In this work the GMM is used to characterize the short-term acoustics of the languages of interest, motivated by the hypothesis that a set of short-term acoustic elements will capture fine structure information that characterizes the languages. This short-term modeling concept contrasts with the PRLM technique which focuses on longer acoustic events. In the case of PRLM, features characterize phonemes whose durations average about 80-100 ms. The acoustics modeled by the GMM system comprise events in the 10 ms range.

The GMM is a multimodal probability density function (pdf) model consisting of a weighted sum of Gaussian pdfs [48]. The feature vectors extracted during front-end processing are used by the GMM to model the acoustic space. During training, the incoming mel-warped cepstral feature vectors are used to deduce these component densities.

Let the *M*-order probability distribution, i.e. GMM, for a set of training feature vectors $\mathbf{X} = \{\mathbf{x}_t\}$, from a language *L* be given by

$$P_L(\mathbf{x} \mid \lambda) = \sum_{j=1}^{M} b(\mathbf{x} \mid \mathbf{\Theta}_j) w_j \tag{3.1}$$

b denotes a single multivariate normal distribution over the feature vectors. There are *M* such pdfs, or "modes," in the overall GMM, the *j*th mode weighted by w_j, and parameterized by mean vector $\boldsymbol{\mu}_j$ and covariance matrix $\mathbf{\Sigma}_j$. For convenience, we package the *j*th Gaussian parameters as an augmented matrix $\mathbf{\Theta}_j = [\boldsymbol{\mu}_j \ \mathbf{\Sigma}_j]$ and $\lambda = [w_j \ \mathbf{\Theta}_j]$. In these terms, the *j*th modal pdf in the mixture model is formally given by

$$b(\mathbf{x} \mid \mathbf{\Theta}_j) = \frac{1}{(2\pi)^{\frac{D}{2}} \left|\mathbf{\Sigma}_j\right|^{\frac{1}{2}}} \exp\left\{-\frac{1}{2}\left(\mathbf{x} - \boldsymbol{\mu}_j\right)^T \mathbf{\Sigma}_j^{-1} \left(\mathbf{x} - \boldsymbol{\mu}_j\right)\right\} \tag{3.2}$$

where D is the total number of cepstral and delta-cepstral coefficients obtained during the feature extraction process.

During the model training the Expectation-Maximization (EM) algorithm [49] is used to obtain the conditional probability parameter set λ. The value of M is determined prior to training. For the purpose of the present research, M is varied to determine the value that yields the best LID results. For other applications using acoustic modeling of speech, this value typically is in the range of 128-512 acoustic units [50].

The testing phase for the mixture model includes tokenization scoring and acoustic likelihood scoring. The tokenization scoring is the main focus of the system developed here, but the acoustic likelihood scoring will be introduced in Chapter 5 as an element for system enhancements in the experimental results. A description of the tokenization process is shown in Fig. 3-4.

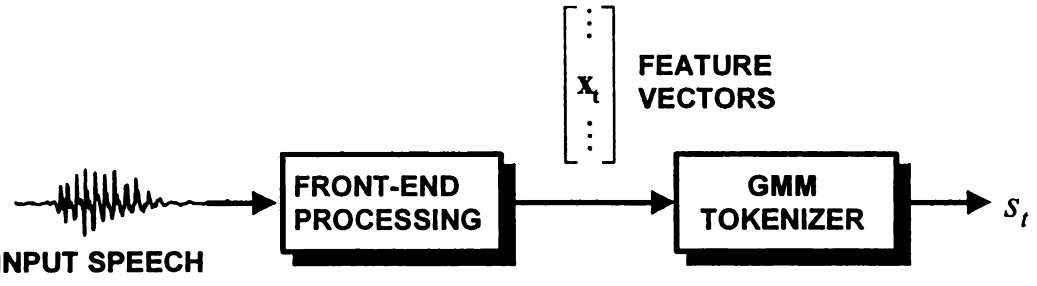

FIGURE 3-4. Single language GMM tokenization process.

The tokenization scoring is used to partition the incoming speech into a stream of tokens corresponding to the acoustics modeled by the GMM. The incoming speech is preprocessed by the front-end system and every vector of cepstral and delta-cepstral features is assigned to one of M acoustic classes, represented by the M mixtures in the GMM determined during training. This assignment is based on the likelihood of each mixture given the observation. Let $\mathbf{x}_t$ be the vector of extracted speech features at time t and P_L the probability distribution associated with the GMM for language L. Formally, the tokenization process associates the feature vector $\mathbf{x}_t$ with the modal index j and speech token s_t by maximizing the posterior likelihood as

$$s_t = \underset{1 \le j < M}{\arg\max} \left\{ b(\mathbf{x}_t \mid \Theta_j) w_j \right\} \tag{3.3}$$

An acoustic likelihood score, the likelihood that the observed speech stream was produced by model P_L, can also be computed from the GMM. Given a sequence of feature vectors, $\mathbf{Y} = \{\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_N\}$ obtained from the incoming speech, and a set of N_L possible languages, each drawing tokens from mixture model P_L, the acoustic likelihood score is obtained by,

$$l(\mathbf{Y}) = P(\lambda^L \mid \mathbf{Y}) = \prod_{i=1}^{N} P(\mathbf{y}_i \mid \lambda^L) \tag{3.4}$$

Note that the superscript λ^L in (3.4) indicates the parameter set for language L, whereas the subscript in λ_j in equation (3.3) refers to the parameters of mixture component j within a particular model.

Although not the case for the system presented here, the acoustic likelihood scores obtained from (3.4) could be used in a stand-alone system for LID. In the stand-alone case, the hypothesized language correspond to the highest likelihood from a set of N_L languages as,

$$\hat{l}(\mathbf{Y}) = \underset{1 \le l \le N_L}{\arg\max} \; P(\lambda^l \mid \mathbf{Y}) \tag{3.5}$$

The general assumption used for the acoustic modeling of speech using a GMM is that a language can be modeled as a set of short-term acoustic events and that each of these events can be modeled by a multivariate Gaussian density. Previous approaches, as those described in Chapter 2, use the mixture description to model broad phonetic classes while the GMM is used in the present approach to model acoustic events.

3.4 Language Modeling

To complete the implementation of the general LID system, a LM is used to incorporate knowledge about the temporal relationships among sequences of the tokenized acoustic events. These events can be obtained from short-term analysis out by the GMM tokenizer, s_t, or from the tokenized segments, v_t, obtained from the long-term processing techniques to be described in Chapter 4. Similarly to PRLM, the LM is based on an interpolated bigram model [23]. A typical bigram model probability is usually estimated using

$$\hat{P}(v_t \mid v_{t-1}) = \frac{\#\text{ of ocurrences of } v_t \text{ after observing } v_{t-1}}{\#\text{ of ocurrences of } v_{t-1}} \tag{3.6}$$

This probability is estimated by the number of occurrences of this bigram in the training data. This type of LM presents a problem for cases when the testing sequence includes symbol sequences not observed during training.

An alternative implementation to the bigram model is the interpolated bigram model [23]. The interpolated bigram model uses a probability estimation method that receives contributions from all estimated probabilities up to order two. The use of the interpolated model allows for better generalization during testing by assigning small nonzero probabilities to events that were not observed during training but arise during testing. The probability computation using the interpolated bigram model is given by

$$\tilde{P}(v_t \mid v_{t-1}) = \beta_0 + \beta_1 \hat{P}(v_t) + \beta_2 \hat{P}(v_t \mid v_{t-1}) \tag{3.7}$$

In this case, the weights β_i represent the confidence in each of the probabilities of interest. This model guarantees that the estimated probabilities be set to a minimum β_0, which resolves the missing observation issues of the basic bigram model [23]. The probabilities $\hat{P}(v_t)$ and $\hat{P}(v_t \mid v_{t-1})$ relate to the occurrence of token v_t in the training data or the number of occurrences of the bigram sequence (v_t, v_{t-1}) as computed in (3.6).

During testing, a sequence of tokenized acoustic events, $V=\{v_1,v_2,...,v_R\}$ is scored against each language model M_L and a likelihood is obtained using

$$P(M_L \mid V) = \prod_{i=1}^{R} P(v_i \mid v_{i-1}, M_L). \tag{3.8}$$

The hypothesized language corresponds to the highest likelihood obtained among the candidate models.

3.5 Backend Classifier

Backend classification is included in most reported phoneme-based LID systems [43]. The purpose of the backend classifier is to extract discriminatory patterns in the LM scores from the previous stage, to enhance the overall classification performance. Additionally, the classifier is used to combine information from different sources of information about the incoming speech to enhance the system performance. A description of this fusion process will be given in Chapter 5.

The classifier used in the proposed system is a linear Gaussian classifier [51]. The LM scores from the language modeling stage are used as the classifier inputs. The classifier input is preprocessed by linear discrimination analysis (LDA) [51]. There are two main reasons for using LDA. First, the LDA process reduces the dimension of the input feature vector to N_c-1, where N_c is the number of classes, i.e. languages. This dimension reduction results in a more robust and computationally simpler classifier. Second, the

LDA process produces a linear combination of the original features by projecting the original space into a lower dimensional subspace that is more efficient for discrimination. The Gaussian classifier is then built in this lower dimensional subspace. A derivation of the LDA process is found in [51].

The Gaussian classifier is obtained by training a Gaussian pdf for each class. The Gaussian densities are trained using a grand (one for all classes) diagonal covariance matrix. This choice is necessary because of the amount of data available.

The final classification is then completed by the evaluation of the likelihoods of each Gaussian density for an incoming vector of LM scores. The likelihood for each class is given by

$$g_i(\mathbf{z}) = -\frac{1}{2}(\mathbf{z}-\boldsymbol{\mu}_i)^T \boldsymbol{\Sigma}^{-1}(\mathbf{z}-\boldsymbol{\mu}_i) - \frac{d}{2}\ln(2\pi) - \frac{1}{2}\ln|\boldsymbol{\Sigma}_i| + \ln(P(c_i)) \qquad (3.9)$$

where $\mathbf{z}$ is the incoming vector of LM scores, $\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i$ are the parameters of the Gaussian pdf for language *i*, *d* is the dimension of the feature vectors, and $P(c_i)$ is the *a priori* probability of class c_i. The final hypothesized language i^* is then given by,

$$i^* = \arg\max_i \; (g_i(\mathbf{z})) \qquad (3.10)$$

3.6 Summary

This chapter presented a general overview of the LID system reviewing the techniques used to capture information about the short-term acoustics. The following chapter presents a description of the approaches for studying these events using long-term processing techniques. A technique is presented that focuses on capturing the long-term information in the sequence of tokens generated by the GMM tokenizer. Another technique is used to study the temporal relationships in the GMM training and testing data rather than in the output stream from the GMM tokenizer.

Chapter 4

Long-Term Approaches to Speech Tokenization

This chapter presents time-domain techniques to study the LID information inherent in long-term speech acoustics. The techniques to be described here analyze the long-term acoustics in the context of the tokenization process. There are two classes of techniques described for studying the long-term acoustics. The first class of techniques builds the long-term events from the *output* of the GMM tokenizer. These techniques include temporal encoding, multigrams and vector quantization. A second class of techniques is used to capture temporal information about the speech in the acoustic domain. The second class of techniques, based on the *shifted-delta-cepstra* (SDC) features [52], replaces the set of conventional cepstral features in the front-end processing with a new set of features used for training the GMM tokenizer.

4.1 Motivation

The performance obtained by phoneme-based LID systems provides the motivation for studying algorithms to incorporate long-term information about the speech acoustics. These algorithms include a variety of techniques directed at extending the time frame of information beyond the 20 ms modeled by the GMM in the foregoing developments. In this section, the elementary acoustic description of a language provided by the GMM is studied to capture this long-term speech information. The modeling of these long-term events is based on the hypothesis that longer events like phonemes can be constructed as sequences of the acoustic events modeled by the GMM. The inclusion of these longer events might capture phoneme-like information previously used by the phonotactic systems for LID. The first class of techniques is used to derive the long-term acoustic events from the output of the GMM tokenizer. A block diagram of the LID system that incorporates these long-term processing algorithms is shown in Fig. 4-1. The long-term processing algorithms studied include temporal encoding, multigram analysis, and vector quantization (VQ). Each of these algorithms is described below.

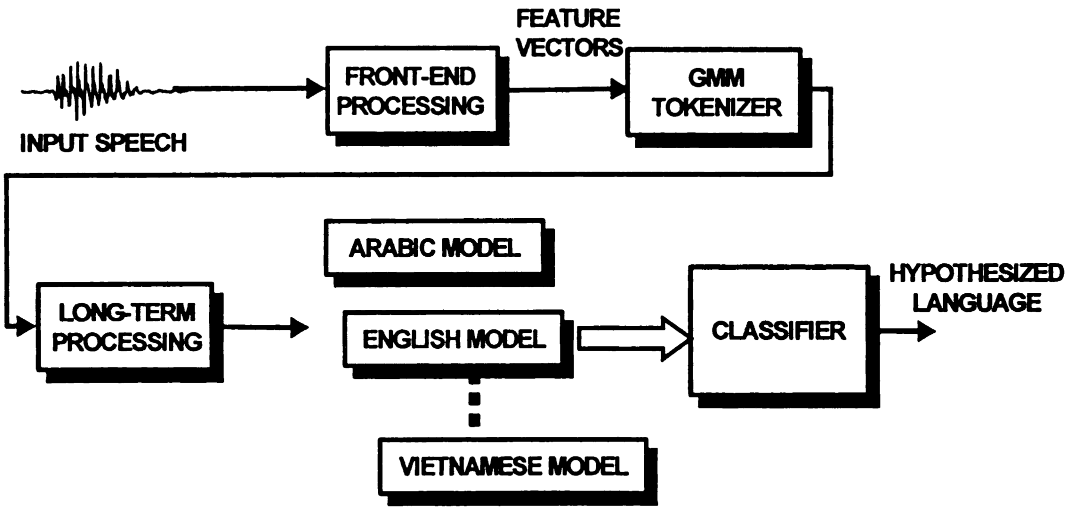

Figure 4-1. LID systems with the inclusion of long-term processing techniques.

4.2 Temporal Encoding

Typically, the stream of indices produced by the GMM tokenizer consists of subsequences representing transition and stationary acoustic regions. The stationary region is expected to be represented by strings of tokens of the same values, $s_t = s_{t+1} = \cdots = s_{t+T}$, in the observed output stream. The idea of temporal encoding is to emphasize information about the change in acoustic events in the tokenized stream while discarding information about the acoustic event duration. This process would emphasize information related to the phoneme sequence transitions. The GMM tokenizer and temporal encoder could be used in place of the phoneme recognizer in phonemic recognition systems provided that a consistent mapping is obtained.

Given a sequence of indices observed at the GMM tokenizer output, $S = \{s_t\}_{t=1}^{N_0}$, the output of the temporal encoder follows the relation

$$v_t = \begin{cases} s_1 & t = 1 \\ s_t & s_t \neq s_{t-1}, t > 1 \\ skip & s_t = s_{t-1}, t > 1 \end{cases} \qquad (4.1)$$

The sequence $\{v_t\}$ is reindexed so that "skip" decisions are no longer elements of the final sequence.

4.3 Multigram Analysis

The purpose of multigram analysis is to obtain the set of sequences that best describes the training data in a statistical sense to be described below. The multigram concept has been proposed by Bimbot [53] and was applied to speech processing by

Deligne [54-56]. Given a maximum sequence length N_{MAX}, a sequence of observations can be described by parsing the sequence into sub-sequences of variable length using a maximum likelihood criterion. For the current work, the multigram analysis is used to determine whether a consistent mapping to phonetic and sub-phonetic units can be obtained.

The multigram analysis, based on the information theoretic concept of minimum description length [57], is best understood with reference to an observation-emitting model. A source is assumed to produce a finite string of independent units, $u_1, u_2, \ldots, u_t \in \{U\}$. Each u_i is composed of a variable number of elementary acoustic events drawn from the set $O = \{o_i\}_{i=1}^{N_O}$. The observed output of the source is the string $u_1, u_2, \ldots, u_t \in \{U\}$. In this framework, the purpose of the multigram technique is to obtain the set U which represents a maximum likelihood segmentation, R, of the observed string. The parsing is represented symbolically as

$$\begin{array}{llllllll}
U: & u_1 & & u_2 & & u_3 & & \cdots \\
 & \Uparrow & & \Uparrow & & \Uparrow & & \\
R: & [\mathrm{o}_1 \;\; \mathrm{o}_2 \;\; \mathrm{o}_3] & \oplus & [\mathrm{o}_4] & \oplus & [\mathrm{o}_5 \;\; \mathrm{o}_6] & & \cdots \\
 & & & & & & & \\
O: & & \mathrm{o}_1\ \mathrm{o}_2\ \mathrm{o}_3\ \mathrm{o}_4\ \mathrm{o}_5\ \mathrm{o}_6 & & & & & \cdots
\end{array} \qquad (4.2)$$

In this work, the observations, o_i, are given by the output of the GMM tokenizer and the hypothesis that set of underlying units are represented by phones. The idea is then to determine whether the phonetic structure known to exist in speech can be modeled by the multigram approach.

Let O denote a string of acoustic observations and R denote a segmentation of O into q variable length sequences of the acoustic events. The T-multigram model computes the joint likelihood $\Lambda(O, R)$ of O and the segmentation R as the product of successive independent sub-sequences, each with maximum length T,

$$\Lambda(O,R) = \prod_{t=1}^{q} P(r_t) \tag{4.3}$$

where r_t represents a set of independent sequences comprising the segmentation R and $P(r_t)$ represent the probability of subsequence r_t.

Defining Q to be the set of all possible segmentations of O with a maximum length of T, the most likely segmentation of O is given by:

$$\Lambda^*(O,R) = \max_{R \in \{Q\}} \Lambda(O,R) \tag{4.4}$$

For example, a string of four acoustic symbols and with a maximum length T=3 can be parsed as:

$$\Lambda^{*}(O,R)=\Lambda(o_1\, o_2\, o_3\, o_4,R)$$

$$= \max_{R\in\{Q\}} \begin{Bmatrix} P\left((o_1o_2o_3)\right) P(o_4), \\ P(o_1o_2)\ P(o_3o_4), \\ P(o_1o_2)\,P(o_3)\,P(o_4), \\ P(o_1)\,P(o_2o_3o_4), \\ P(o_1)\,P(o_2o_3)\,P(o_4), \\ P(o_1)\,P(o_2)\,P(o_3o_4), \\ P(o_1)\,P(o_2)\,P(o_3)\,P(o_4) \end{Bmatrix} . \tag{4.5}$$

The algorithm used to deduce the multigram model for a training corpus, proposed by Bimbot [53], is provided in Table 4-1.

I. Initial estimation of probabilities.

$$P_{initial}(b_t)=\frac{\#of\ \ ocurrences\ \ of\ \ each\, multigram\ \ i}{Total\ \ number\ \ of\ \ ocurrences\ \ of\ \ all\ \ multigrams}$$

b_t is the set of all possible sequences observed in the training data

II. Segmentation into most likely sequences using Viterbi procedure, with best parsing of the string O maximizing the likelihood at this step of the iteration,

$$\Lambda^{*}(o_1,\ldots,o_{k+1},R)=\max_{1\leq i\leq n}\{P([o_{k-i+2},\ldots,o_{k+1}])\Lambda(o_1,\ldots,o_{k-1+1},R)\}.$$

III.Re-estimation of probabilities based on the updated segmentation from step II.

$$P(b_t)=\frac{\#of\ \ ocurrences\ \ of\ \ multigram\ \ i\ \ after\ \ segmentation\, in\, step\,\mathrm{II.}}{Total\ \ number\ \ of\ \ ocurrences\ \ of\ \ all\ \ multigrams\ \ after\ \ segmentation}$$

IV.Repeat steps II and III until convergence of all probabilities or a fixed number of iterations is completed.

TABLE 4-1. Description of the multigram training algorithm.

During testing, the output token stream from the GMM is segmented in a similar way to that used in training. A feature vector is extracted from the speech and tokenized into the acoustic space defined by the GMM. The tokenized stream is then segmented using the multigram codebook (set of sequences) obtained during training. Given a tokenized stream of symbols out of the GMM tokenizer, $S=\{s_0,s_1,\ldots,s_N\}$, and the codebook Λ^*, that maximizes the likelihood of the training tokens, the tokenized stream is segmented according to,

$$R^* = \arg\max_{R \in \Lambda^*} \Lambda(S, R) \tag{4.6}$$

which results in the incoming set of tokenized symbols becoming segmented into a new sequence $R=\{r_0,r_1,\ldots,r_W\}$. The new sequence generated is composed of elements of variable size based on the composition of the codebook Λ^*. The number of elements obtained in sequence R is unknown, but it is at most the number of tokens in the stream out of the GMM tokenizer.

4.4 Vector Quantization

The VQ technique is studied as an alternative approach to the multigram technique. Similarly to the multigram technique, the VQ technique [58] is used to derive subsequences of an utterance's feature stream that provide discriminatory information about the languages. The VQ approach attempts to identify similar events to those targeted by the multigram approach but can potentially do so at a lower computational cost. There are two main differences between the multigram technique and VQ. First, the

VQ method uses a fixed size for the segments (vectors) used to build the VQ codebook or dictionary rather than the variable size for the segments studied in the multigram technique. Second, VQ uses representative samples as its codebook while the multigram technique uses all samples obtained from the training set to build its dictionary. The VQ technique is usually implemented by the algorithm in Table 4-2 [58].

I. Codebook initialization

   Choose an initial set of samples at random as the initial codebook.

II. Nearest neighbor search

   For each vector in the training sample set, search the codebook and assign each vector to the closest element in the codebook.

III. Codebook update

   Update the distributions using the new assignment of the elements to the classes in step II.

IV. Repeat steps II and III until termination criterion is met.

TABLE 4-2. Steps of the general vector quantization algorithm.

In the current implementation, vectors of length L_{VQ} are created from the output stream of the GMM tokenizer. The training algorithm is a modified version of Table 4-2. In the conventional implementation of the VQ algorithm the codebook is refined by iterating over an initial codebook. In the implementation used in this research, the codebook initially contains only one element, chosen at random, and the codebook size is increased by one element, following the criterion in Table 4-3, on each iteration until the desired number of elements *P* is obtained. The number of elements, *P*, is varied from 16-256 to obtain the value resulting in the best performance. This customized version of the VQ algorithm allows for a more flexible implementation and conforms better to the set of parameterizations of interest. A description of the customized algorithm is shown in Table 4-3.

I. Codebook initialization

Partition the incoming tokens set of tokens, $S=\{s_0, s_1, \ldots, s_N\}$, into a set of vectors of length L_{VQ}, G.

$$\cdots \mathbf{g_n} = \begin{bmatrix} s_{(n)M_{VQ}} \\ \vdots \\ s_{(n+1)M_{VQ}-1} \end{bmatrix}, \; \mathbf{g_{n+1}} = \begin{bmatrix} s_{(n+1)M_{VQ}} \\ \vdots \\ s_{(n+2)M_{VQ}-1} \end{bmatrix}, \cdots$$

Choose one vector at random from the set of training vectors as the initial codebook.

II. Compute the distance from training vector $\mathbf{g_n}$ to each element in the codebook, where the Euclidean distance along all the L_{VQ} elements in each vector

$$\left\| d_{n,k} \right\|^2 = \sum_{i=1}^{M_{VQ}} \boldsymbol{\mu}_{\mathbf{s_{i,n}}}^{\mathbf{T}} \boldsymbol{\mu}_{\mathbf{s_{i,k}}}$$

where $\boldsymbol{\mu}_{\mathbf{s_{i,n}}}^{\mathbf{T}}$, $\boldsymbol{\mu}_{\mathbf{s_{i,k}}}$ represent the mean vector of the Gaussian Mixture Model component associated with the i^{th} elements of vectors $\mathbf{g_n}$ and $\mathbf{g_k}$ respectively.

III. Add vector $\mathbf{g_n}$ to the codebook that has the maximum distance to the elements already in the codebook, increasing the codebook size by one element.

IV. Repeat steps II and III until the desired size for the codebook, P, is obtained.

TABLE 4-3. Description of the modified vector quantization algorithm.

The Euclidean distance metric is used for the development of the VQ codebook and its subsequently used in testing. The computation is performed by relating the tokenized symbols out of the GMM tokenizer to the original distribution in the model. This approach could introduce an undesirable additional level of quantization in the process but it allows for a more uniform system for comparison of the approaches used for including temporal information within the tokenization process. For example, at time *t*, the tokenized segment has been estimated as s_t from (3.3), the index *j* of the most likely mixture in the model. The index *j* obtained can then be related to a Gaussian distribution, whose mean is used to compute the distance in (4.7).

During testing, an incoming feature vector is assigned the code of the closest vector in the codebook following the same distance metric as during training. For a testing vector, $\mathbf{g}_n$, the closest vector in the codebook is assigned as the output token, v_{k^*}, following,

$$\begin{aligned} &g_n \rightarrow v_{k^*}, \textit{ where} \\ &k^* = \arg \min_{1 \le k \le P} \left\| d_{n,k} \right\| \end{aligned} \tag{4.7}$$

4.5 Shifted-Delta-Cepstra (SDC) Features

The algorithms described in Section 4.1 rely on analyzing the output of the GMM tokenizer to discover temporal relations that could be exploited in the tokenization process. This section describes a different approach in which the temporal information is incorporated into the tokenization process directly in the acoustic domain using SDC features.

A previous study in LID has employed SDC features for use with acoustic likelihood systems [52]. In this system, the acoustic similarity between a language model represented by a GMM and an input utterance is computed and the highest scoring model is hypothesized as the language of the utterance. This system incorporates SDC features to model longer temporal content in the feature domain rather than studying the temporal characteristics generated by tokenizing the acoustic events.

Bielefeld [52] empirically studied the use of different parameterizations of SDC features for the discrimination of English from another languages. The best parameterizations obtained from Bielefeld's work were then employed by Singer [59] in the study of language discrimination using higher order GMMs than previous studies [43, 60]. The SDC method relies on the use of four parameters to be described below. Bielefeld's study reveals that proper parameterization is critical to the performance of the SDC method. Bielefeld approaches the parameterization problem empirically, evaluating a constrained set of values for each parameter.

In Singer's work, a GMM is trained for each language. During testing, utterances of speech not previously presented to the system are scored against each language-dependent GMM model and the model with the highest likelihood is hypothesized. This type of testing is similar to that described in Section (3.3).

In the work presented here, SDC features are used to characterize properties of the long-term acoustics. The SDC features are derived from the same set of mel-warped cepstral coefficients used to train the GMM tokenizer (Chapter 3), and the SDC features replace the cepstral coefficients as the training elements for the tokenizer.

The SDC features are constructed as a concatenation of delta-cepstra coefficients obtained at discrete intervals around a frame time of interest, say, *t*. The SDC computation requires four integer-valued parameters: *N, d, P,* and *k*. The *j*th SDC feature vector for frame *t* is computed as

$$\Delta c_{j,t} = c_{j,(t+d)} - c_{j,(t-d)} \tag{4.8}$$

where $c_{j,t}$ is the *j*th mel-warped cepstral coefficient at time *t*. *N* refers to the number of delta-cepstral coefficients used on each block; *d* is the time difference (in terms of frame indices) between frames used in computing the delta cepstra, as in (4.8). *k* indicates the number of blocks included in the computation. Last, *P* is the time shift between each of the *k* blocks. The parameters and the computation of the SDC are shown in Fig. 4.2. The parameter *q* in Fig. 4.2 is used to index the *k* blocks ranging from 0 to *k-1*.

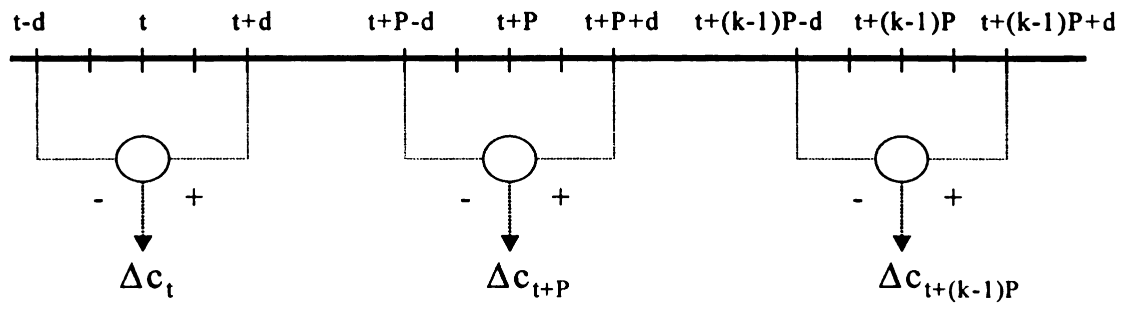

FIGURE 4-2 Block diagram for the computation of the shifted-delta cepstra coefficients.

The accumulated SDC vector at time t is obtained by the concatenating the delta coefficients from all k blocks. The vector resulting from Fig. 4-2, for example, is:

$$w_t = \begin{bmatrix} \Delta c_t \\ \Delta c_{(t+P)} \\ \vdots \\ \Delta c_{(t+(k-1)P)} \end{bmatrix} \tag{4.9}$$

The dimension of each vector w_t is Nk and the total number of SDC feature vectors obtained after processing a speech utterance is the same as that using conventional cepstral features.

4.5.1 SDC Parameterization

The absence of a closed-form model relating LID performance to the SDC parameters requires that parameterization be achieved empirically. Accordingly, the parameterization for the work at hand was guided by the results from similar previous work [52], and by the relations between phonemes and the information carried by the SDC features.

Rather than performing a computationally-expensive search through the parameters, the parameterization was achieved by a constrained search of alternative parameterizations. This search was performed using as reference the parameterization obtained by Bielefeld and the average duration of phones.

The computed average phoneme durations from the OGI corpus [30] were studied to establish the temporal range for SDC parameterization. Table 4-4 presents the shortest, longest and average durations of phonemes on the labeled partition of the OGI corpus. Minimum and maximum durations refer to the values for the shortest and longest phonemes observed, while the average duration is computed over the entire phonetic inventory of each language.

Language	Minimum duration (ms)	Maximum duration (ms)	Average duration (ms)
English	40.7	215.6	85.9
German	44.2	248.1	92.6
Hindi	42.1	157.7	79.8
Japanese	44.1	156.9	79.3
Mandarin	45.8	165.3	88.0
Spanish	37.1	182.9	80.3

TABLE 4-4. Phoneme durations for the labeled languages in the OGI corpus.

The 80 to 90 ms average duration seen in Table 4-4 is consistent with the SDC parameterization yielding the best performance in the studies by Bielefeld and by Singer. The best parameterization obtained by Bielefeld is 6-1-3-3 (*N-d-P-k*) and this set of parameters is used by Singer for his work on LID using mixture models. Based on these previous results, the 6-1-3-3 parameterization was used in the present work as a base for a search among possible configurations.

There are two main ideas explored with respect to the initial parameterization. The first factor considered was the effect of the offset parameter, *d*. This parameter is closely related to the time duration over which the feature vector is computed. Since the *d* parameter determines the distance between the frames over which the delta computation is performed, varying this parameter might capture temporal information of interest while decreasing the number of blocks, resulting in feature vectors of lower dimension.

The second parameter, *k*, is also intimately related to the time duration over which the feature vector is computed. For a fixed *d*, the *k* parameter determines the time frame covered by the SDC parameterization.

Although the SDC parameters are interrelated, the search for an effective parameterization was conducted independently over a different range of values for each parameter. Each of the values evaluated empirically was considered for a maximum coverage of about 90 ms with the exception of the *k* parameter, which was expanded to a maximum of 180 ms.

The results of the parameterization process along with the performance of the LID system based on using these features are presented in Chapter 5.

Chapter 5

Experiments and Results

This chapter presents a description of the experiments conducted for the various configurations of the LID system. There are two major concepts of interest to be addressed by these experiments. First, the focus is on overall system performance for 12-way closed set classification using the CallFriend corpus. Second, experiments are conducted to assess the performance of the GMM-tokenization system using a second speech corpus.

5.1 Corpus Description

The experiments studied in this work are based on the CallFriend corpus [61], which consists of unscripted conversations in 12 languages captured over domestic telephone lines. The corpus includes speech in the following 12 languages: Arabic, English, Farsi, French, German, Hindi, Japanese, Korean, Mandarin, Spanish, Tamil and Vietnamese. The subset of the CallFriend corpus used in these experiments is the same as that used for the 1995 NIST language recognition evaluation [62]. The training set of the corpus consists of 20 telephone conversations of approximately 30 minutes each from each of 12 languages. The development set consists of 1147 30-second utterances and the evaluation set of the corpus consists of 1492 30-second utterances. The distribution of the evaluation set is biased towards English, which includes five times as many testing samples as in the other languages. The training set of the corpus was used to train both the GMM tokenizers and the LMs. The evaluation set was used to test the full system with the development set used for system enhancements such as backend classification.

Although most of the testing in this work is performed on the newer CallFriend set because of its conversational style, the OGI corpus is included for validation of the results obtained from the system that yields the best results on the CallFriend corpus. The OGI corpus [30] , is also a collection of telephone speech based on monologues and responses to a set of questions. The OGI corpus is a smaller database than CallFriend, but includes all the same languages except Arabic. This database has been used extensively for LID research because of the availability of transcriptions for six of its languages, English, German, Hindi, Japanese, Mandarin and Spanish. The OGI corpus is divided into four partitions: initial training, development set, extended training and final test. In accordance with the NIST 1994 [25] evaluation guidelines, both the initial training set and the extended training set are used for training the system in this research for a total of about 70 two-minute training segments per language. The test set is composed of two sets: a 45-second utterance set and a 10-second utterance set. The 10-second set includes 625 segments while the 45-second test includes 198 calls. For both test subsets of the OGI corpus, the same number of testing utterances is available for each language.

Although both the CallFriend and OGI corpora have been used for LID research in the past, there are major differences between them. First, the CallFriend corpus is composed of conversational-style utterances while the OGI corpus consists of monologue-style utterances. The other major difference is the data availability and partitions. In the case of the CallFriend corpus, about 10 hours of speech are available per language compared to about 150 minutes available for OGI. Additionally, the CallFriend corpus contains both a development and evaluation partition compared to OGI, which only includes an evaluation partition. These differences in the amount of training data available and set partitions required certain modifications in the experimental procedure used to evaluate the system using the OGI corpus.

5.2 Experimental Framework

This section describes the general details of the experimental setup of this work. These conditions apply to all the system configurations described later in this chapter with some exceptions that will be noted as appropriate. The areas to be described include the training and testing for the GMM tokenizer, the LMs and the Gaussian backend classifier.

The GMM used for tokenization is trained on the set of features derived by front-end processing. There are two feature sets used: classical cepstral features and SDC features. In the case of classical features, a set of cepstral features is combined with a set of conventional delta features to provide the full set. The typical choice for speech recognition varies between 8 and 12 cepstral coefficients [4]. The number of cepstral coefficients used here is 10. The delta-coefficients are computed by subtracting the cepstral coefficients at time $t\text{-}2$ from the cepstral coefficients at time $t+2$. The dimension of the feature set using classical cepstral features is then 20, composed of 10 cepstral coefficients and 10 deltas.

The second choice is the SDC features, which were described in Chapter 4. The number of features in this case, *Nk*, depends on the parameterization choice made. *N* being the number of cepstral coefficients used to compute the deltas and *k* the number of blocks used. The final number of parameters used for the SDC parameterization is discussed in Section 5.6.

The next area is the training of the Gaussian backend. The training of the backend in the case of the CallFriend corpus is performed using the development partition, reserving the evaluation partition for testing the full system.

The backend classifier training for the OGI corpus test is handled differently. This difference arises since the OGI corpus includes only a single partition for testing. To address this problem, two different testing scenarios were explored. First, each of the testing sets, 45-seconds and 10-seconds, is halved. The halves are used for training and testing independently. The second scenario used for testing the full system under the OGI corpus is based on the leave-one-out or jackknife technique [51].

The leave-one-out technique evaluates the system by training the classifier using all the samples available except one. This one sample that is not used for training is the only sample available for testing. The technique is then repeated until all samples have been used for evaluation.

The GMM tokenizers are trained using data from a single language, similar to the case of the phoneme recognition systems. The LM is then trained by tokenizing the incoming feature vectors using the GMM. These tokens represent the most likely mixture or acoustic class. Each LM is trained using only samples representative of the language of interest, for example the Arabic LM is trained using Arabic input samples only.

5.3 Baseline GMM System

The baseline GMM system, shown in Fig. 5-1, uses a single tokenizer to perform the LID task. The results are obtained by evaluating the system using each of the available languages for training the tokenizer and averaging the error rates across the 12 languages. The training and testing are carried out according to the procedure described in Section 5.2.

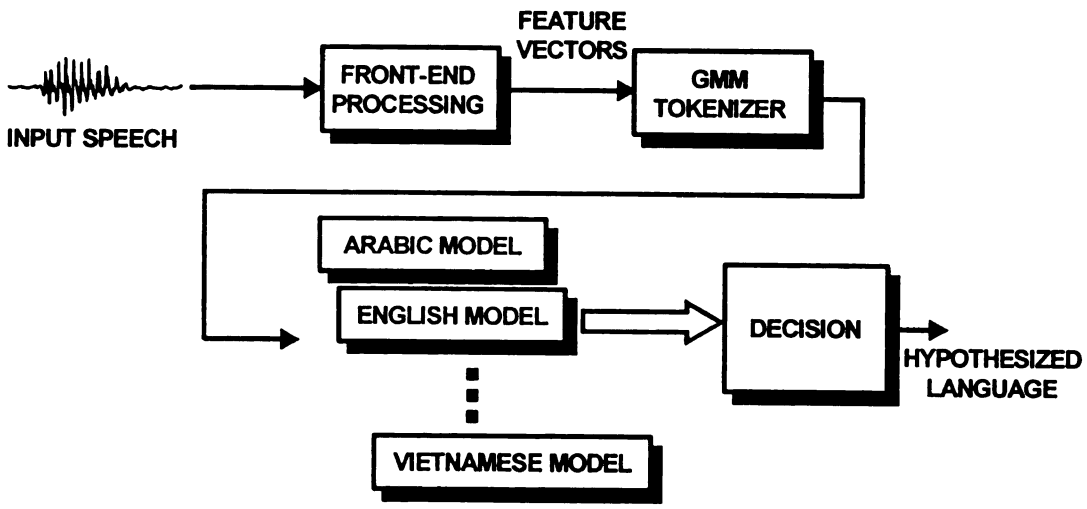

FIGURE 5-1. Baseline single tokenizer GMM system.

Training set	Mixture order				Single PRLM
	64	128	256	512	
Arabic	76.4	72.3	67.5	64.3	N/A
English	74.6	68.4	62.6	55.3	35.0
Farsi	77.1	72.3	67.7	64.8	N/A
French	80.0	74.7	70.7	65.9	N/A
German	77.4	71.9	69.2	63.6	44.7
Hindi	78.6	73.3	68.0	62.5	46.8
Japanese	78.4	73.4	69.7	64.8	49.5
Korean	76.2	72.6	68.0	63.3	N/A
Mandarin	77.4	73.2	70.0	65.0	44.4
Spanish	74.9	72.3	68.7	65.7	47.6
Tamil	77.5	72.3	68.1	64.1	N/A
Vietnamese	80.0	75.5	70.3	66.9	N/A

TABLE 5-1. Error rates for the single tokenizer GMM system for the CallFriend evaluation set as a function of the tokenizing language and the mixture order. A not available (N/A) note is used for results that can be determined given that phonemic transcriptions are not available in those languages.

Table 5-1 presents the results obtained in previous work for the single tokenizer PRLM system and for the GMM system developed in this work. The results for the single tokenizer PRLM systems are given for only those systems for which transcribed data are available. The results show decreased performance for the GMM systems with respect to the PRLM systems. The results also show better performance for the system using an

English based tokenizer, a result likely attributable to the English bias in the evaluation data.

There is a trend of better discrimination with increasing mixture order up to 512 mixtures. There are two main reasons for avoiding evaluation of the system with more than 512 mixtures. First, the amount of available training data does not allow the bigram LM to be trained reliably for the number of tokens generated using orders above 512. The rule of thumb for training pattern recognition systems is usually a 10:1 ratio of samples per estimated parameter [63]. A 1024 LM requires the estimation of one million parameters, requiring about 10 million training samples, far more than the 2 million training samples in the CallFriend corpus. Second, the computational complexity of evaluating an incoming utterance increases with the higher orders which could reduce the impact of the computational gains obtained.

5.4 GMM Systems with Long-Term Acoustics

This section presents alternatives for the enhancement of the GMM-tokenization system. The results presented in the previous section show relatively poor performance for the LID problem with a GMM-tokenization system based only on short-term speech information. As described in Chapters 3 and 4, previous phoneme-based systems focus on acoustic events with an average duration around 80-100 milliseconds. On the other hand, the GMM tokenization system focuses on speech frames with a 10-millisecond duration. Given the disparity in performance between the GMM-based system and the phoneme-based system, the baseline GMM system is modified to include information about longer acoustic events. The techniques evaluated are described in Sections 4.2-4.4 and results are reported sequentially here.

5.4.1 Temporal Encoding

The temporal encoding represents an attempt to exploit the acoustic structure of phones from within the GMM system. The phone is typically modeled as a three-state hidden Markov model where the beginning and end states are assumed to model transition regions [64]. The temporal encoding technique is used to explore the acoustic structure of the phoneme in conjunction with the acoustic events targeted by the GMM. The block diagram for the GMM system with temporal encoding is shown in Fig. 5-2.

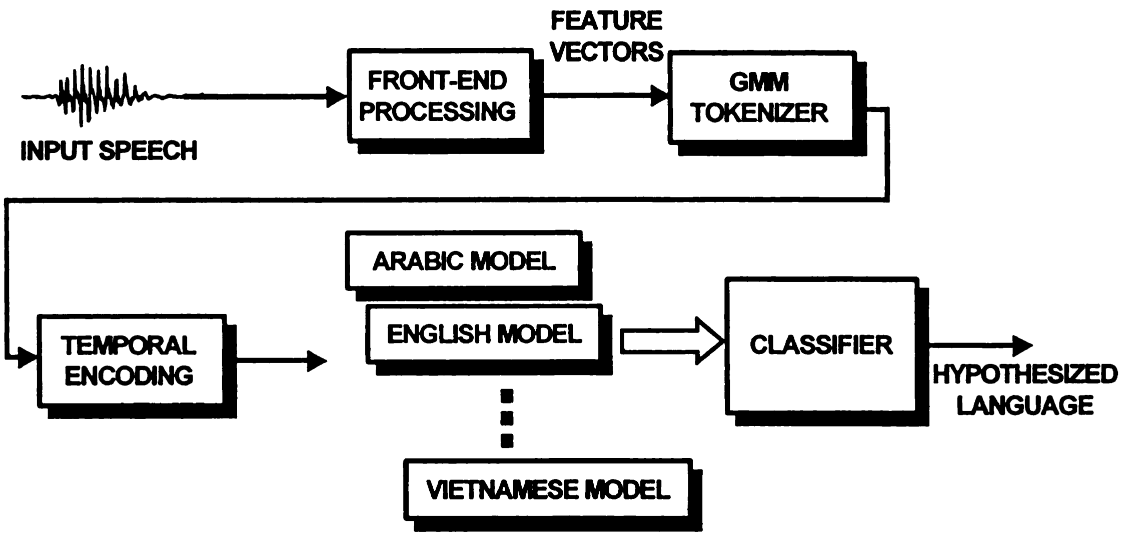

FIGURE 5-2. Single tokenizer GMM system with temporal encoding.

Training data	Mixture order			
	64	128	256	512
Arabic	74.5	70.3	65.2	63.1
English	71.1	68.4	62.1	57.0
Farsi	75.4	72.2	66.2	64.0
French	77.4	73.1	68.8	64.1
German	74.0	70.4	64.9	62.4
Hindi	74.7	69.8	66.0	62.1
Japanese	75.9	71.4	68.6	63.7
Korean	74.3	70.8	67.0	64.3
Mandarin	73.9	72.9	68.4	63.1
Spanish	73.8	70.8	68.0	64.8
Tamil	74.5	69.7	65.8	62.7
Vietnamese	77.2	73.1	68.8	66.0

TABLE 5-2. Error rates for the single tokenizer GMM system with temporal encoding for the CallFriend evaluation.

The results in Table 5-2 show that the temporal encoding technique results in marginal improvements in the error rates for 10 of the 12 languages (c.f. Table 5-1). The small impact of this technique on performance is related to the events on which the GMM is focused. The observation of the GMM tokenizer output stream reveals that the sequence does not support the hypothesis that the stream consists of long strings of the same token. This problem becomes more evident with higher mixture orders. The use of higher orders makes the derived acoustic units obtained closer to each other in the acoustic space. This concentration of acoustic classes produces too fine a tokenization for the stream of tokens to contain stable regions of the same token, minimizing the effect of the temporal encoding.

5.4.2 Multigrams

The multigram technique represents an alternative method for the inclusion of temporal information in the tokenization process. The idea of the multigrams is to model frequently-occurring sequences of acoustic events. These sequences are hypothesized to carry information related to the phonetic structure of the language. This technique is used to derive these events following a maximum likelihood probabilistic approach (c.f. Section 4.3) rather than phonetic knowledge provided by a human expert.

There are some practical considerations in the multigram computation process. First, the technique requires a length parameter that must be determined prior to beginning the maximum likelihood analysis. Second, the size of the dictionary of multigrams has to be constrained. Since the dictionary is used in conjunction with the GMM tokenizer to partition the incoming test utterances, the size of the dictionary directly influences the size of LM. The size of the LM is then dictated by the amount of training data and influences the computation time required for LID.

The experiments indicate that the most important factor both for performance and computational complexity is the LM size. In order to maintain a low computational cost and given the available data to reliably train the LMs, it is desirable to keep the dictionary size restricted to 512 tokens. This limitation then creates constraints on the parameters for both the mixture order and the multigram length.

First, mixture orders above 128 result in a dictionary that is mostly composed of length-one acoustic elements. This dictionary results from the elimination, through the iteration process, of the longer sequences. This choice of a maximum order of 128 mixtures or greater effectively results in the same experiment as that described in Section 5-3. Second, the length of the multigrams is also parameterized to a maximum length of five.

The multigrams experiments are conducted using only English speech to train both the GMM tokenizer and the multigrams. The purpose of constraining these experiments to English as the tokenizing language is to obtain a set of preliminary results before further exploring the technique. Results for the multigrams technique are shown in Table 5-3 for different tokenizer orders and multigram lengths.

Maximum multigram length	Mixture Order			
	128	64	32	16
5	55.90	59.25	67.09	72.25
4	56.03	59.38	68.77	74.20
3	58.04	60.86	67.90	74.93
2	59.18	60.66	70.11	76.41

TABLE 5-3. Error rates for the single tokenizer GMM system with different multigram lengths and mixture order for the CallFriend evaluation set.

The results indicate small benefits of using the multigram technique for speech tokenization compared to the baseline system in Section 5.3. There are two possible reasons for these results. First, the explosive increase in dictionary size, as the order of the Gaussian mixture was increased. Since the dictionary size has to be constrained because of both available data and computational considerations, the resulting dictionary is composed of the shorter, most common, sequences that tend to carry less discriminative information about the languages. Second, increasing the model order further results in an acoustic structure that is too fine to build consistent sequences.

Possible modifications for this method include methods of consolidating acoustic sequences with similar characteristics as a means to deal with both dictionary size and dictionary entries similarities. Two alternatives were considered to address the problems presented by multigrams. These methods include a modified multigram method that incorporates continuous distributions to model the sequences, and a VQ approach. Given the lower computational complexity of VQ, it was chosen as the alternative to multigrams.

5.4.3 Vector Quantization

The third technique evaluated to incorporate temporal speech information was VQ. The VQ technique was implemented according to the description in Section 4.4. The set of parameters studied for this experiment includes the order of the mixture model, the length of the sequence to be vector-quantized and the size of the codebook. The results from these parameterizations are shown in Fig. 5-3. The plot shows the results obtained for a single tokenizer system based on English speech. Similarly to the multigram case, both the GMM and the VQ codebook are trained with English. The plot shows results for different orders and different segment lengths. The codebook size is 256 elements that performed better on average than codebook sizes of 512, 128, 64 and 32 elements. It is not evident why the codebook size of 256 outperformed other sizes but it is a consistent result since most applications that use acoustic modeling of speech typically use anywhere from 128-512 acoustic units [50].

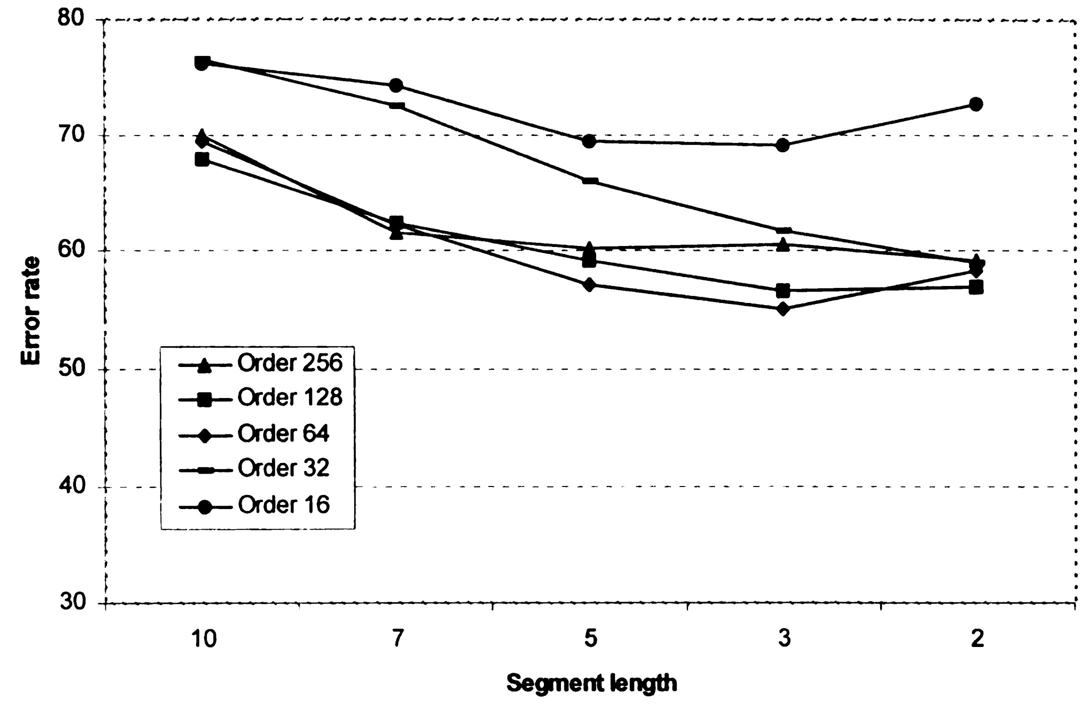

FIGURE 5-3. Error rates for the single tokenizer GMM system with VQ processing for the CallFriend evaluation set. Results are shown for various GMM orders.

The results using long-term information about the speech do not show significant improvements over the baseline system without long-term processing. The common problem with these methods is the inability to reduce the sequences of acoustic events to a set that captures the phonetic structure of the languages. Additional experiments were conducted to study the tokenization consistency of the long-term processing results when compared to the phoneme labels resulting in multiple token sequences for each phoneme. These results support the explanation that the phonetic structure is not captured by either of these methods. The next section presents another approach to enhancing the baseline GMM system.

5.5 GMM System Revisited

Given the unacceptable error rates of the baseline GMM tokenization system, a different approach was needed to enhance the system performance to a competitive level to that of phoneme-based systems. A number of alternatives have been proposed by other investigators for phoneme-based systems. These alternatives include the use of backend classifiers [65] and the use of multiple single-language tokenizers in parallel [43]. This section presents a series of enhancements to the baseline GMM system including the effect of a backend classifier on single tokenizer systems and then extending the use of a backend classifier for a system with multiple tokenizers. The effect of mixture order on each of these systems is also assessed.

5.5.1 Backend Classifier

A backend classifier provides the means of exploiting LM scores further by studying the score patterns as an alternative for correcting the errors obtained in classification. Although there is a wide range of options for a backend classifier, a Gaussian backend classifier (GBE) was found to yield the best performance in initial experiments. The GBE is also computationally simpler than the other alternatives evaluated in this work such as neural networks.

The GBE inputs, consisting of the LM scores, are processed by linear discriminant analysis (LDA) [51] prior to evaluation by the classifier. The LDA is performed to obtain a superior representation of the data for discrimination. Recall from Chapter 3 that although the LDA process yields a lower dimensional space where the training set samples are easier to separate among the different classes, it is not guaranteed that this will necessarily result in a better discrimination among the testing samples. Given the new representation in a lower dimensional space, the LDA process also reduces the number of inputs, typically to the number of output classes minus one. In the case of the CallFriend corpus the number of input features after LDA processing is reduced to 11 given that there are 12 classes.

The GBE employed in this work is trained using a diagonal, grand (same for all classes) covariance matrix of the linear combination of LM scores. Although choosing a per-class, full covariance matrix could potentially lead to better performance, it would also require additional data per class than the available amount. The use of a grand covariance matrix allows for all the available data from all classes to be used in the estimation of the covariance matrix, resulting in a more robust estimate of this matrix statistic.

In the set of experiments conducted for this research, the development partition of the CallFriend corpus is used to train the GBE and the evaluation segment of the corpus is used to test the full system. Table 5-4 shows the effect of using a GBE for the single tokenizer system. The table shows the performance of the single tokenizer systems (one system per language) when the GBE is used, in comparison to the baseline GMM system. The table also presents the effect of GBE on systems trained using various GMM orders.

Training data	Mixture order							
	64		128		256		512	
	No GBE	With GBE	No GBE	With GBE	No GBE	With GBE	No GBE	With GBE
Arabic	76.40	48.2	72.30	41.90	67.50	38.40	64.30	36.20
English	74.60	46.60	68.40	41.40	62.60	38.10	55.30	35.90
Farsi	77.10	49.70	72.30	48.60	67.70	42.30	64.70	38.30
French	80.00	52.40	74.70	55.20	70.70	41.20	65.90	38.50
German	77.40	53.00	71.90	46.50	69.20	43.40	63.60	39.00
Hindi	78.60	48.90	73.30	44.50	68.00	40.10	62.50	35.70
Japanese	78.40	55.90	73.40	45.20	69.70	40.40	64.80	40.50
Korean	76.20	52.20	72.60	45.40	68.00	40.40	63.30	38.50
Mandarin	77.30	48.50	73.20	46.70	70.00	42.40	64.90	38.10
Spanish	74.90	50.90	72.30	44.90	68.70	41.40	65.70	37.50
Tamil	77.50	50.10	72.30	43.60	68.10	40.60	64.10	38.00
Vietnamese	80.00	54.80	75.50	48.50	70.30	45.40	66.90	41.00

TABLE 5-4. Error rates for the single tokenizer GMM system with and without a Gaussian backend classifier for CallFriend evaluation set.

The effect of using the GBE is even more clearly in Fig. 5-4. This figure shows the average error rate across all possible tokenizers for the single tokenizer system. The figure shows the effect of the order on the single tokenizer system with a backend.

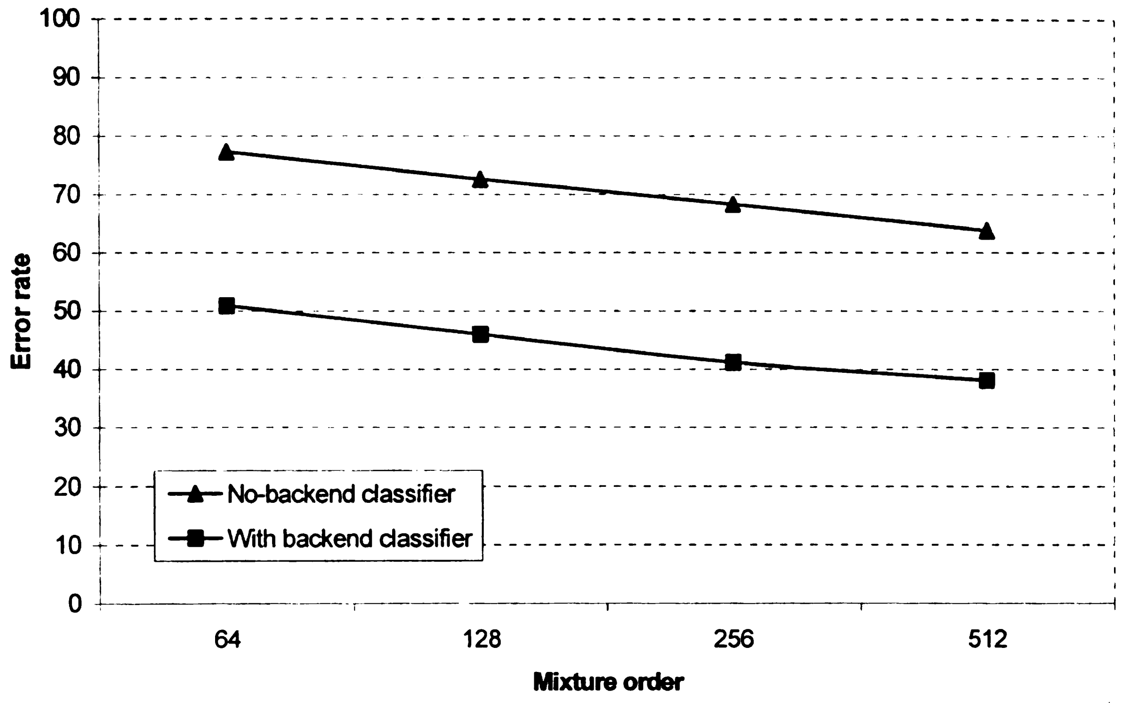

FIGURE 5-4. Average error rates for the single tokenizer GMM system with a backend classifier and different mixture orders for the CallFriend evaluation set.

From the results in Table 5-4, it is apparent that the addition of a GBE has a positive impact on the system performance regardless, of the language used for tokenization or the order of the GMM. Orders below 64 are not included because of the poor performance obtained, and orders above 512 are not used since not enough data are available to properly train the LMs. From the results obtained by using a GBE, it is believed that the LM scores provide enough consistency such that the classification errors obtained can be corrected.

The use of the backend classifier will now be extended to the case of multiple single-tokenizer GMM systems in parallel.

5.5.2 Parallel GMM System

The success of the addition of a backend classifier to the baseline GMM system and previous work in the area of parallel implementations of phoneme-based system [43] are the main motivations for the augmented parallel implementation of the GMM system. The parallel implementation of phoneme-based and GMM-based systems is motivated by the assumption that the decoded streams provide out of each tokenization system provide additional evidence about the input language increasing the confidence of the hypothesized alternative. The parallel implementation for the case of phoneme-based systems has already been evaluated resulting in better performance than stand-alone single phoneme-recognition systems [43]. The architecture for the implementation of the parallel GMM-based system is shown in Fig. 5-5. Although the figure only shows two tokenization systems in parallel, a full implementation of the GMM-based system for the CallFriend corpus includes 12 systems in parallel.

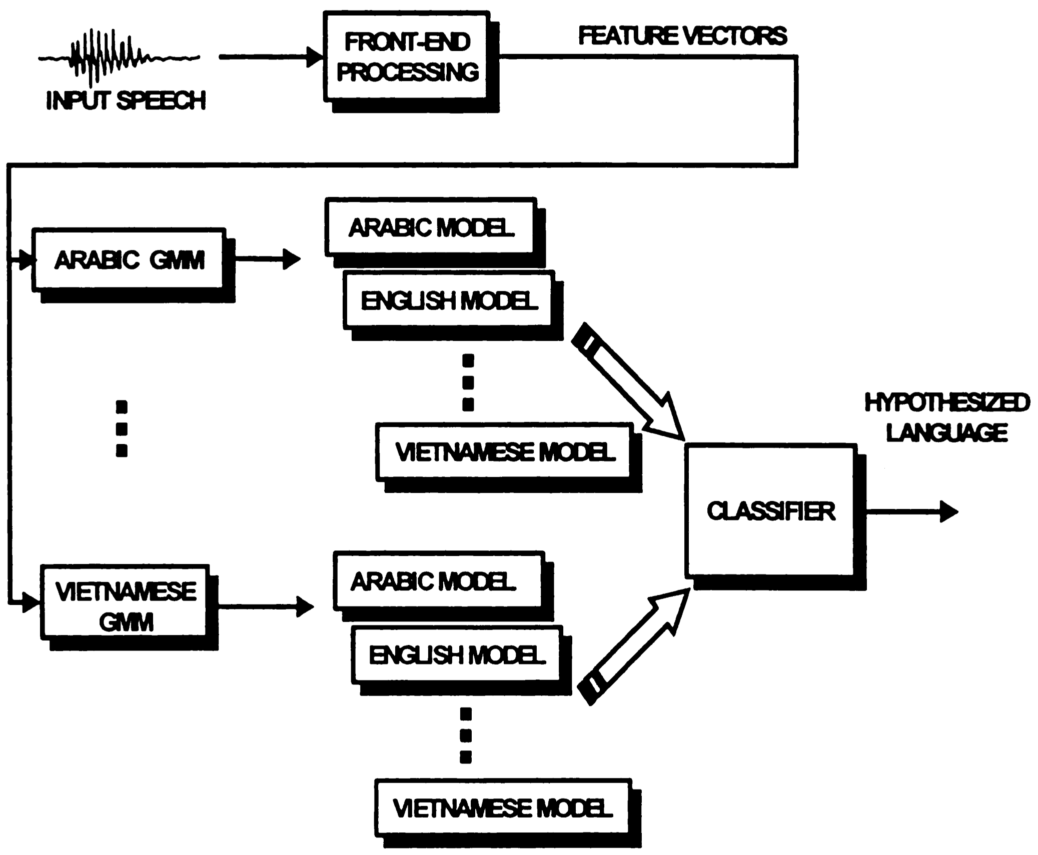

FIGURE 5-5. Parallel implementation of the GMM system using multiple single language tokenizers.

The implementation includes multiple single-tokenizers in parallel, resulting in 12 LM scores per tokenizer. All LM scores are fed to the GBE where the input features are preprocessed by LDA. Similarly to the case shown for the single tokenizers, the CallFriend development set is used to train the classifier and the CallFriend evaluation set is used to evaluate the system. For the parallel implementation of the GMM system, only the best performing mixture order, 512, is evaluated.

The results of the parallel implementation, as the number of tokenizers is increased, is shown in Table 5-5. Although using a particular set of tokenizers could yield better results for some cases, in the current implementation the number of single-tokenizer systems is increased by adding one tokenizer at a time from Arabic to Vietnamese, alphabetically, until all 12 were used. For example, if a single tokenizer is studied then it is only Arabic, if two tokenizers are chosen those are Arabic and English and so on until a system with all 12 tokenizers is used. There is some evidence that choosing the tokenizers carefully yields better performance than the systematic approach used but the process of choosing subsets of all the tokenizers could also lead to over fitting the system to particular experimental corpora.

Number of tokenizers	Error rate
1	36.20
2	33.45
3	35.05
4	35.32
5	36.60
6	35.19
7	36.80
8	35.59
9	38.81
10	34.72
11	39.14
12	36.26

TABLE 5-5. Error rates for the parallel GMM system with conventional cepstral features for the CallFriend evaluation set.

Table 5-5 shows the limited impact of using multiple tokenizers. The error rate varies between 39% and 33% without a clear trend as the number of tokenizers is increased. These results will be revisited in the next section where the parallel implementation is enhanced by adding acoustic scores as part of the system.

5.5.3. Parallel GMM System with Acoustic Scores Fusion

The combination or fusion of multiple information sources task has been proposed as an alternative for LID by Parris [66], among others. The parallel implementation of the GMM tokenization system is a natural candidate for the fusion of multiple sources of information. The block diagram for the enhanced system is shown in Fig. 5-6. The GMM tokenization process produces acoustic likelihood scores as a by-product, which can be combined with the LM scores at the classifier stage. The enhanced implementation of the system has 13 scores per tokenizer, 12 LM scores and one acoustic score, compared to the 12 LM scores of the implementation in the previous section.

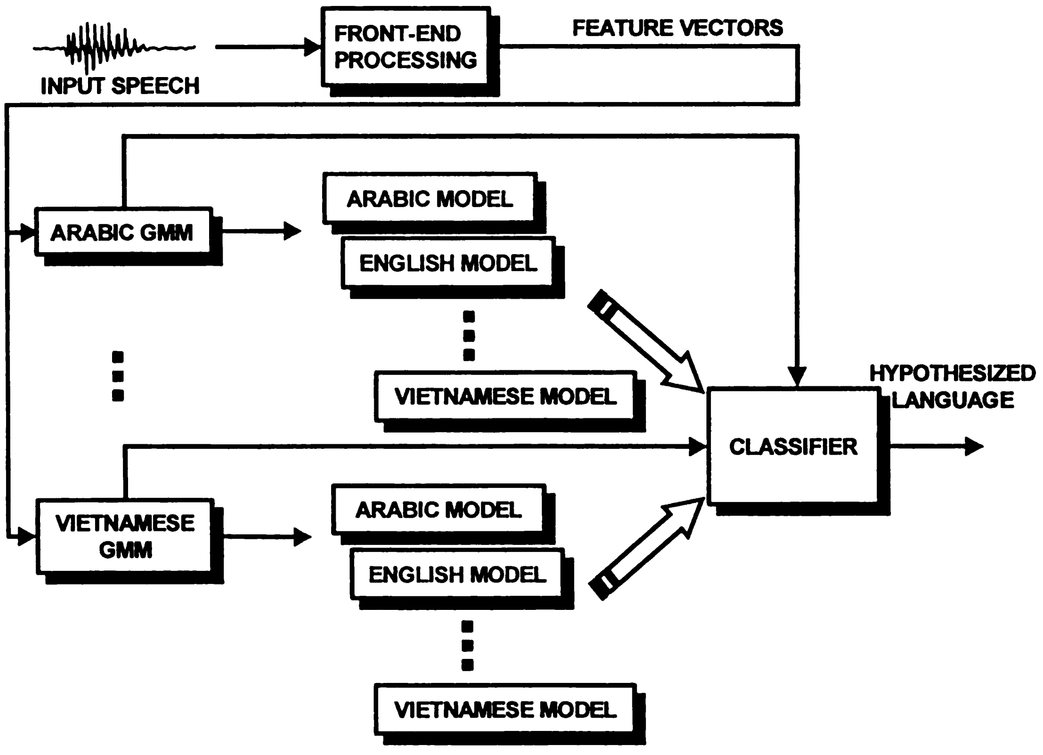

FIGURE 5-6. Parallel implementation of the GMM system with acoustic scores fusion.

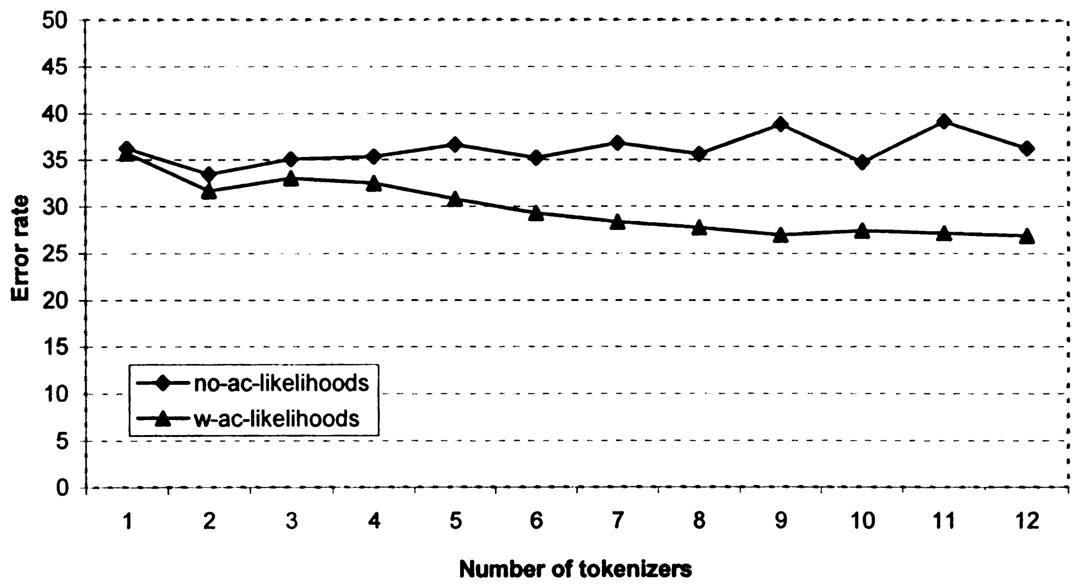

FIGURE 5-7. Error rates comparison for the parallel system implementation, including systems without and with the fusion of acoustic scores.

Figure 5-7 presents a comparison between the parallel system results in the previous section and the enhanced parallel system including acoustic scores. The figure shows the improved performance of a parallel system implementation with acoustic score fusion. The plot shows that contrary to the behavior of the initial parallel implementation, the system error rate decreases as the number of tokenizers is increased. This decrease in error rate shows that the information captured by the acoustic processing is clearly of benefit when combined with the LM scores obtained from the tokenization process. The error rate for the 12-tokenizer system with acoustic fusion is 26.88%. Another implementation of the parallel system with acoustic fusion, combined with the inclusion of SDC features, is presented in the next section.

5.6 SDC Parameterization Experiments

The SDC features were introduced in Chapter 4. This section presents a series of experimental results aimed at obtaining a suitable parameterization of the SDC features for the parallel system described in section 5.5.

The experiments conducted focused on two main issues. First, the effect on recognition of the time-related SDC parameters *d*, *k* and *P*, was studied to establish the desired temporal coverage. Second, once the *d*, *k* and *P* parameters were set, the focus was shifted to the number of coefficients, *N*.

As explained in Chapter 4, it is reasonable to study a set of parameters that focuses on temporal segments of similar duration to that of phones. Based the LID error rates obtained in the previous sections for cepstral and delta cepstral features, the *d*, *k* and *P* parameters are combined with a fixed *N* value of 10 coefficients. Table 5-6, shows results for different values of *d*, the time difference between the frames over which the delta coefficients are computed. The table shows results for the average error rate for the baseline single tokenization system. The table also includes the error rate for the acoustic likelihood based system using all 12 languages.

	SDC parameters 10-*d*-*P*-*k* 10-1-0-1	10-2-0-1	10-4-0-1
Average single tokenization error	56.87	54.67	51.81
Acoustic only error rates	53.02	56.23	56.17

TABLE 5-6. Error rates for the CallFriend evaluation set using SDC parameter *d* = 1,2 and 4.

The results in Table 5-6 do not show a clear difference between the candidate values for *d*. Based on these results, *d*=4 yields better performance for GMM-tokenization while *d*=1 results in better error rates for acoustic likelihood processing. The *d* parameter will be set to 1 for two main reasons. First, it is consistent with previous results using SDC parameterization [52]. Second, it provides a parameterization alternative for the remaining time parameters *P* and *k*, where the SDC segments do not overlap.

The *P* parameter was set to 3, which is the minimum value that eliminates gaps between SDC blocks. The current configuration for the remaining experiments is then 10-1-3-*k*. The results for various *k* values are shown in Table 5-7.

	k = 2	k = 3	k = 4	k = 5	k = 6
Average single tokenization error	49.24	48.58	50.21	53.12	55.63
Acoustic only error rates	41.09	31.84	33.65	35.05	39.28

TABLE 5-7. Error rates for the CallFriend evaluation set for SDC parameter *k*= 2, 3, 4, 5 and 6.

The results for the various values of the k parameter clearly show $k = 3$ yields better performance than the other values evaluated. This result is consistent with those obtained by Bielefeld [52] and results in a temporal coverage in the same range as phonemes.

The last parameter to be evaluated as a verification step is the current choice of N=10 coefficients. The initial choice for N was motivated by the baseline results for the GMM tokenization system in Section 5.3 and the set of SDC parameters studied by Bielefeld [52]. The system is evaluated using values for N = 6,8 and 10. The results for this evaluation are shown in Table 5-8.

	SDC parameters		
	6-1-3-3	8-1-3-3	10-1-3-3
Average single tokenization error	48.83	48.12	48.58
Acoustic only error rates	32.17	33.11	31.84

TABLE 5-8. Error rates for the CallFriend evaluation set for SDC parameter N=6, 8 and 10.

The parameterizations shown in Table 5-8 show similar performance for the different values of N. The final choice of SDC parameters was 10-1-3-3 based on the slightly lower error rate of the acoustic system. The temporal segment parameterized by the 10-1-3-3 set of parameters is about 90 milliseconds resulting in a similar time span as that observed for the average phoneme durations obtained for the OGI labeled corpus. The results for the parallel GMM system with acoustic score fusion are presented in the following section.

5.7 SDC-based Parallel GMM System

The results of the parallel implementation for the GMM tokenization system are shown in Fig. 5-8. The plot includes results for the system with and without acoustic likelihood scores fusion. The trend seen is similar to the implementation of the system using conventional cepstral features.

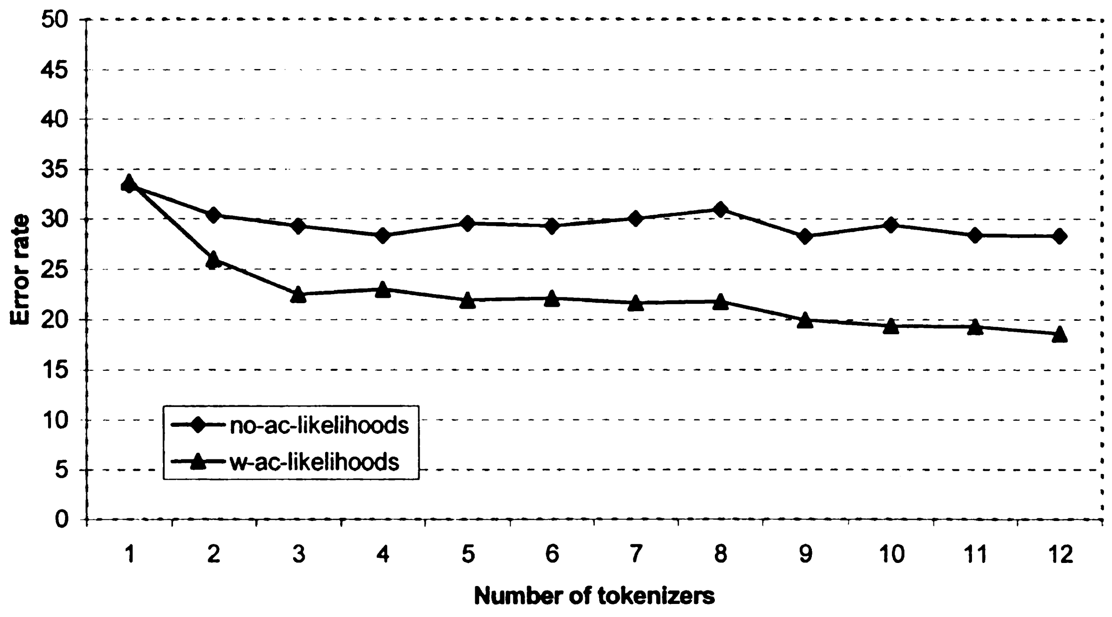

FIGURE 5-8. Error rates comparison for the SDC-based parallel GMM system implementation, including a system without the fusion of acoustic scores and with fusion of the acoustic scores.

A comparison between the parallel implementations of the GMM tokenization system with classical cepstral features and with SDC features is shown in Figure 5-9. The full system using all tokenizers based on SDC features clearly outperforms the system based on cepstral features. One particular point of interest is the effect of adding tokenizers in each system. In the case of the system implemented with classical cepstral and delta-cepstral features, performance worsens when the Farsi-based tokenizer is added, and when the Spanish-based tokenizer is included. In the case of the SDC-based system, performance is affected adversely when the French-based, the Hindi-based and the Korean-based tokenizers are included. This difference in behavior suggests that the information captured for each of the cases is different. The improvement obtained when the SDC features are utilized also shows the positive impact of including increased temporal information in the tokenization process. The error rate for the 12-tokenizer system using SDC features and score fusion is 18.57% compared to the 26.88% obtained for the system using conventional cepstral features.

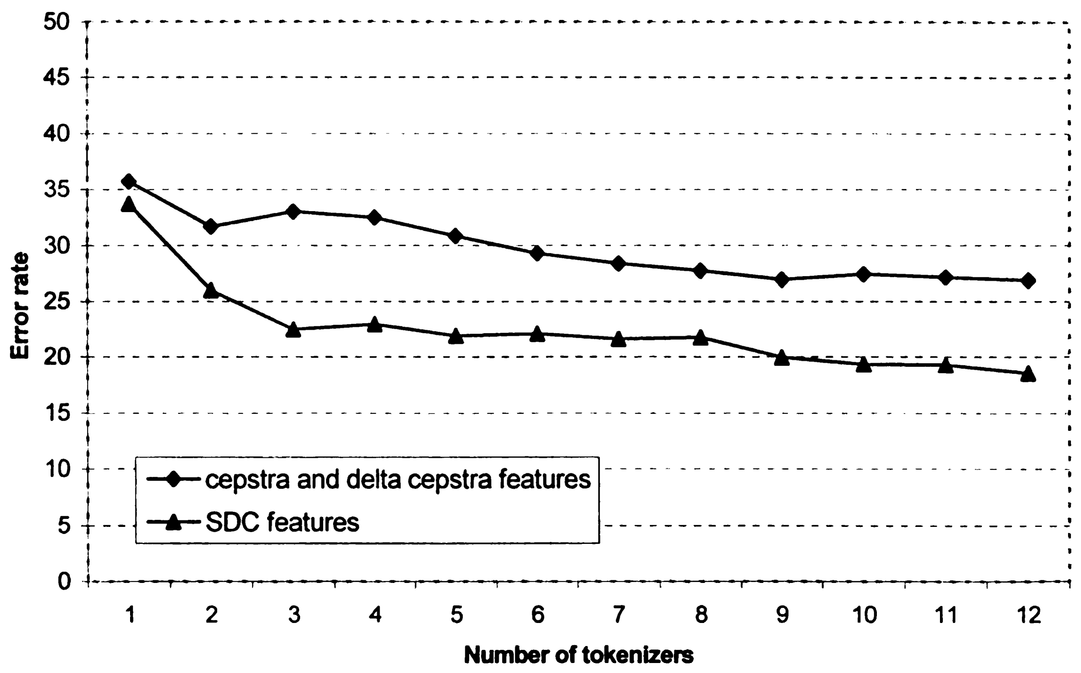

FIGURE 5-9. Error rates comparison between the parallel GMM system implementation using SDC features and cepstral features.

The 12-tokenizer system was evaluated using another set, the OGI corpus, with the results shown in the next section. Results for the evaluation of lower mixtures order are included in the Appendix.

5.8 Evaluation of the SDC-based Parallel GMM System with the OGI Corpus

There are multiple reasons to evaluate the proposed SDC-based parallel GMM tokenization system with acoustic scores fusion with the OGI corpus. First, although the CallFriend corpus is more recent and has been used for the most current LID system evaluations, there are numerous reports in the literature presenting results using the OGI corpus. Second, evaluating the system with a different corpus provides information about system adaptability. In particular, the fact that the OGI corpus contains considerably less training data than the CallFriend corpus was one of the main challenges faced in this evaluation.

The availability of fewer training data examples in the OGI corpus requires some changes to the previous experimental setup. First, the order of the Gaussian mixture model has to be lowered from 512 mixtures to 128 mixtures. This imposition was necessary in order to have enough data to train the LMs. Second, although results are shown for the full system using a GBE, the limited amount of training samples requires some additional changes be made to the process of training and testing the GBE as described below.

Contrary to the case of the CallFriend corpus where the data are divided into training, development and evaluation sets, the OGI corpus includes only a training segment and a testing segment. The training segment includes the initial and extended partitions while the testing segment includes a 45-second and a 10-second partition. These limitations potentially result in a backend classifier that is not as robust as that trained with the CallFriend corpus.

The first set of experiments was conducted by splitting each of the testing segments into two partitions. The 45-second utterance set is divided into two sets of 99 samples each. During the system evaluation, each partition was used for training and testing to estimate the error rate for the full set. A similar scheme was used to evaluate the 10-second utterance set by dividing the set into a partition of 313 samples and a second partition of 312 samples. Similarly to the case of the 45-second segments, the roles of the sets were reversed in order to evaluate the full set.

Figure 5-10 presents the results for the evaluation of the full system for both test partitions in the OGI corpus. The trends observed are similar to those seen for the CallFriend corpus, where the error rate is lowered by the addition of tokenizers. Also the lower error rate seen for the 45-second utterances compared with the 10-second utterances is expected and consistent with the results reported in the literature.

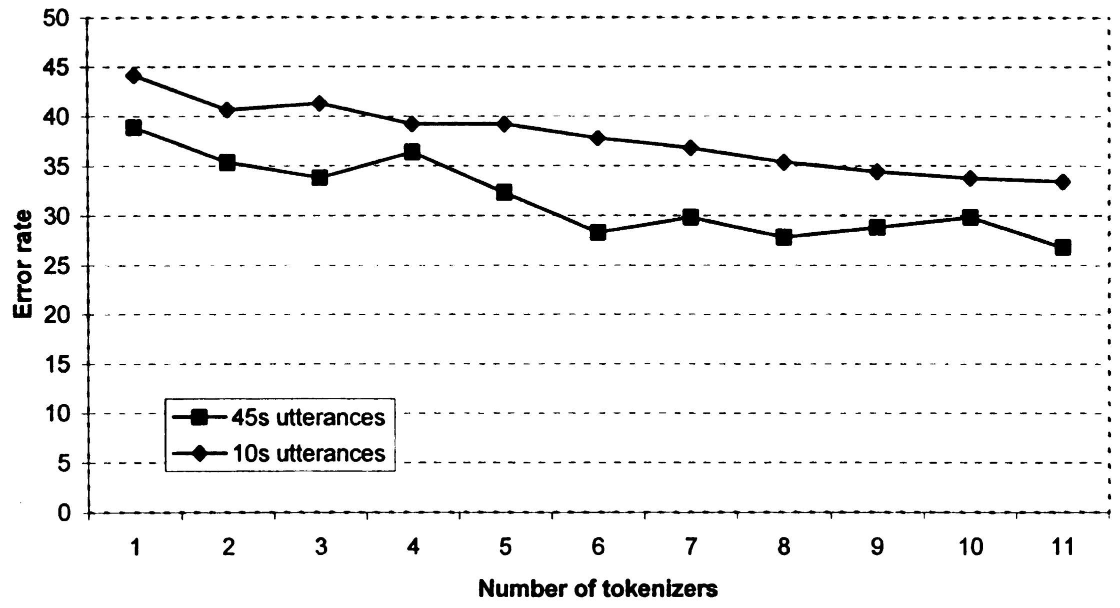

FIGURE 5-10. Error rates for the OGI testing set using the SDC-based parallel GMM system.

A second set of experiments was performed with the OGI corpus to estimate the performance of the system when additional training data are available. This experiment was aimed at evaluating the system performance when a more robust backend classifier is trained. In this case the leave-one-out technique was implemented to estimate the error rate. This training scheme results in the system using 197 utterances to train the classifier for evaluating the 45-second performance and 624 utterances to train the classifier for the 10-second partition.

Figure 5-11 shows a comparison of the performance of the system for the 45-second utterance for the different classifier training schemes. Figure 5-12 shows the same comparison for the evaluation of the 10-second utterances. For both cases the system performance is improved by the increase in the amount of training data. The error rate for the full system is lowered from 26.77% to 23.74% for the 45-second utterance test and from 33.45% to 29.44% for the 10-second utterance test.

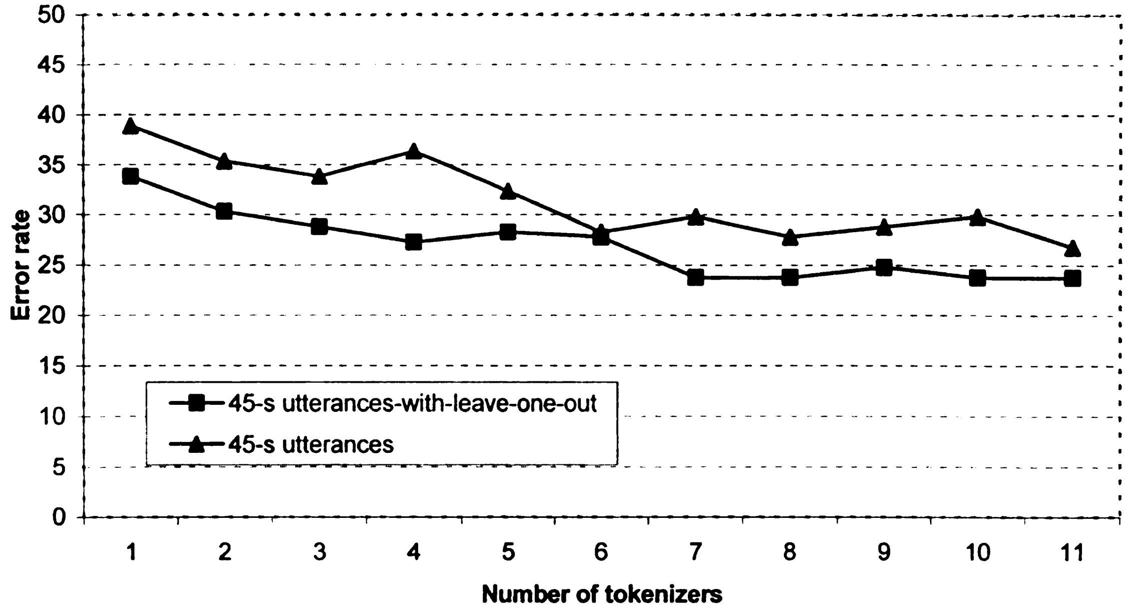

FIGURE 5-11. Comparison of the performance of the system evaluation for the 45-second set of the OGI corpus under different classifier training alternatives.

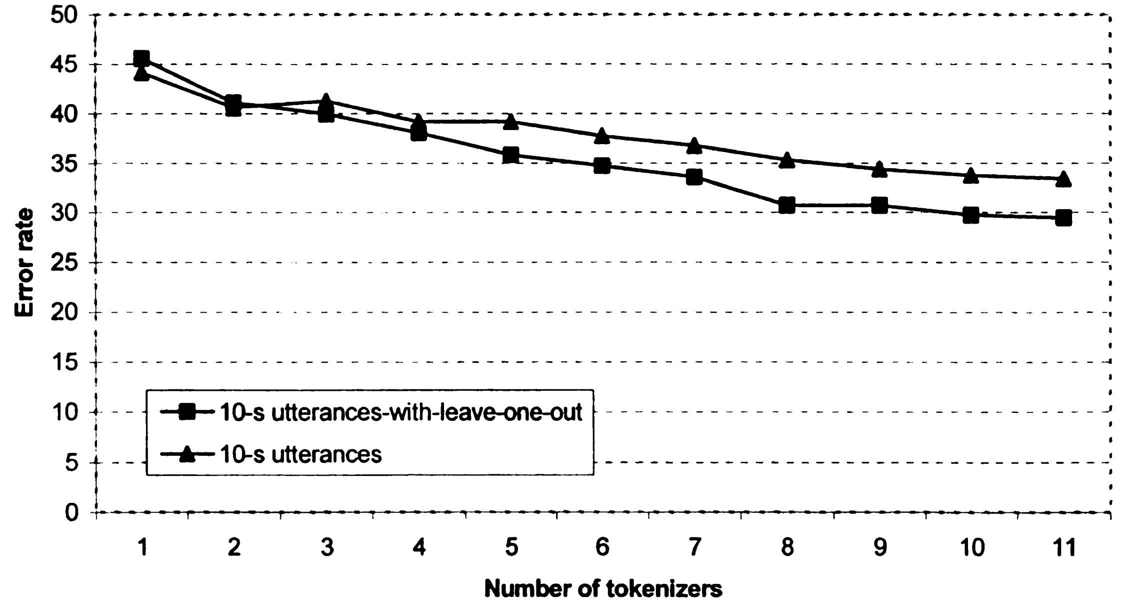

FIGURE 5-12. Comparison of the performance of the system evaluation for the 10-second set of the OGI corpus under different classifier training alternatives.

A third set of experiments was conducted to further understand the potential performance of the system using the OGI corpus if enough data were available and higher GMM orders could be used. For this experiment set, the original system architecture is altered to accommodate the new information source. In previous experiments the acoustic scores used for fusion have been generated as part of the tokenization process. In this experiment the tokenization part of the system is fused with acoustic scores obtained from a GMM system trained with a higher order. Now each language includes a tokenization stage of order 128 and a second GMM of order 512 to compute the acoustic scores. This experiment was expected to produce results closer to the performance attainable for a system trained completely with a 512-mixture order.

Figure 5-13 presents the comparison of the evaluation of the two systems for the 45-second utterances. The plots show the results for the leave-one-out training scheme for a system based on a 128-order for both tokenization and acoustic scoring and for the modified system where the tokenization is performed on a 128-order mixture but the acoustic scoring is computed based on a 512-order mixture. Figure 5-14 shows similar results for the 10-second utterances.

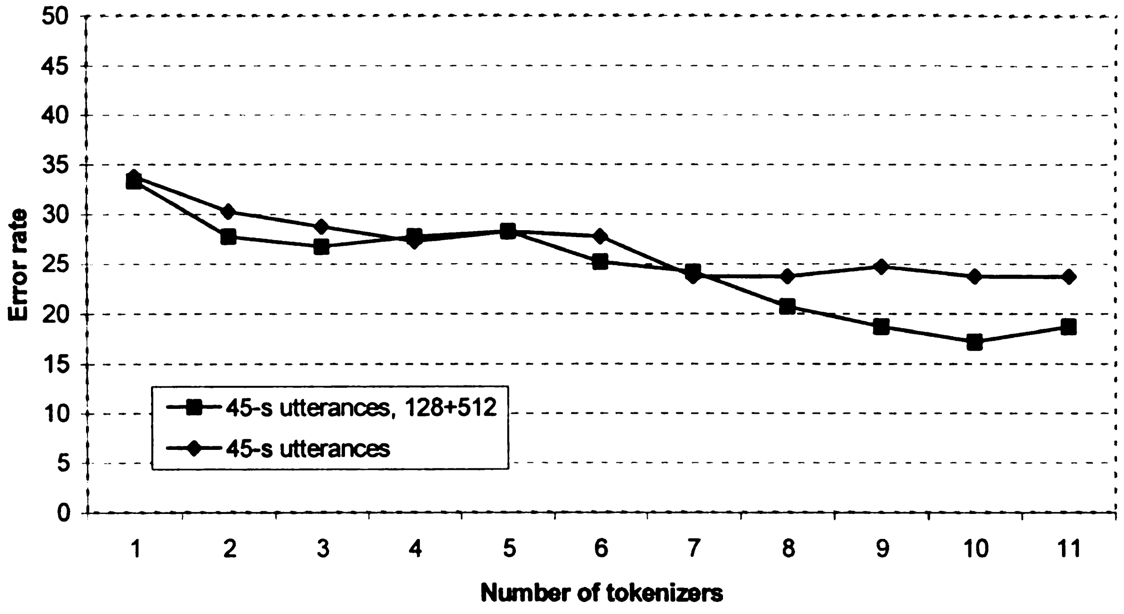

FIGURE 5-13. Error rates comparison for the parallel system using the OGI 45-second set with leave-one-out evaluation and higher mixture acoustic scores fusion.

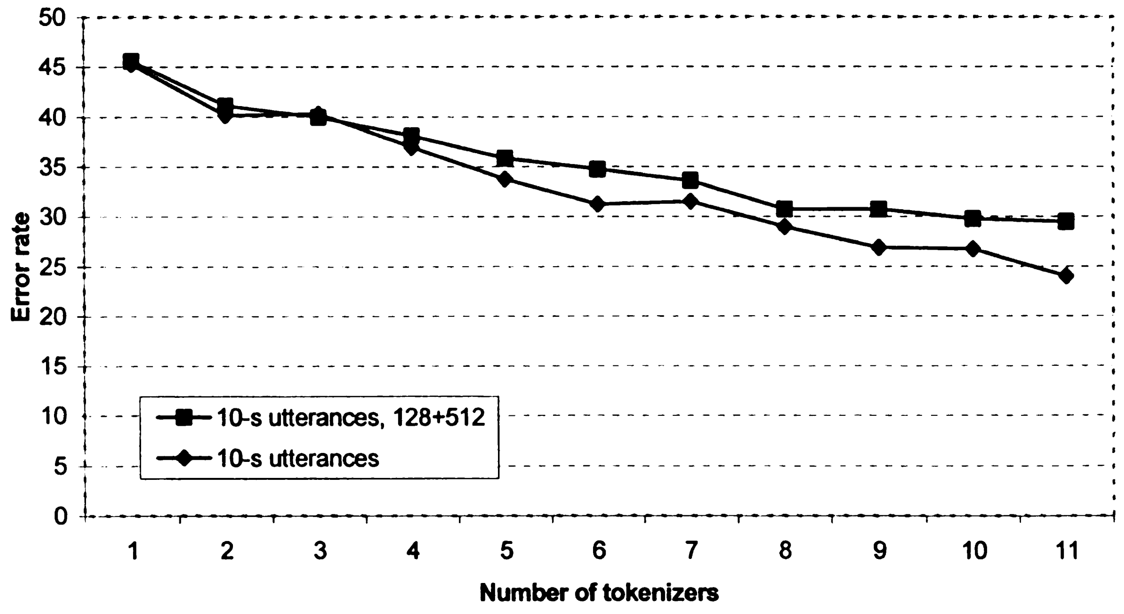

FIGURE 5-14. Error rates comparison for the parallel system using the OGI 10-second set with leave-one-out evaluation and higher mixture acoustic scores fusion.

The results in Figures 5-13 and 5-14 show initial evidence of the expected performance of the parallel system for the OGI corpus if enough data were available to train the LMs at higher orders. As expected, the results of this experiment show improvements in the error rates from previous experiments. The 10-second utterance test results in a 24.0% error rate and the 45-second utterance test results in a 18.69% error rate. Similar experiments with the CallFriend corpus show that using the combination of a 512 GMM order acoustic scoring system along with a 128 GMM order tokenization system is below the performance obtained for a system that uses 512-mixtures for both tokenization with language modeling and acoustic scoring.

5.9 Analysis of the Confusion Matrix for the CallFriend evaluation set

The confusion matrix is a tool used in pattern recognition to represent the results of a classification experiment. The rows indicate the correct class while the columns indicate the recognized class. Each row-column intersection, C_{ji}, represent the number of times a testing sample from class *j* is classified as class *i*. The diagonal elements represent the number of correct classifications. Table 5-9 presents the confusion matrix obtained for the full system described in Section 5.7.

	Ara	Eng	Far	Fre	Ger	Hin	Jap	Kor	Spa	Man	Tam	Vie
Ara	68	0	3	1	1	0	0	0	1	3	1	2
Eng	0	388	0	3	0	10	21	28	3	0	1	24
Far	1	0	65	3	3	1	0	1	5	0	0	1
Fre	3	2	1	59	4	1	1	2	2	1	0	4
Ger	1	2	7	2	65	3	0	0	0	0	0	0
Hin	1	2	2	1	0	43	3	2	2	5	10	5
Jap	0	0	1	0	0	2	67	5	2	1	0	1
Kor	0	0	1	0	1	1	2	68	3	0	2	0
Spa	0	2	5	0	1	0	5	1	138	0	0	4
Man	1	1	1	2	0	7	2	1	0	128	4	6
Tam	4	0	0	0	0	5	3	1	0	4	56	0
Vie	0	3	0	0	0	2	1	1	2	0	0	70

TABLE 5-9. Confusion matrix for the SDC-based parallel system using the CallFriend evaluation set.

The most important result in Table 5-9 is the lower performance obtained during the classification of utterances of Hindi speech. This result is consistently seen in LID using either acoustic or phoneme-recognition systems. It is not clear why the utterances of speech coming from Hindi perform consistently worse than the other languages, but possible explanations include the multiple dialects in the Hindi language, and the inability of the current modeling to accurately model the characteristics of Hindi.

5.10 Miscellaneous Experiments

A number of additional experiments were conducted to assess possible shortcomings of the previous experiments. The most notable of these experiments include: multi-language tokenization, classical cepstral coefficient stacking and cross-corpus experiments.

The multi-language experiments consisted of evaluating the GMM system with a single tokenizer trained on data, instead of one tokenizer per language, from all the available languages using classical cepstral features. The motivation was to model the general acoustics better under the assumption that better tokenization could be obtained. This experiment resulted in similar performance to experiments using single tokenizers and could not be improved substantially by the addition of acoustic scores or a backend classifier.

Another experiment was conducted to study the effect of adding temporal information about the speech to the tokenizer by concatenating consecutive vectors of classical cepstral features. This experiment was motivated by the results obtained with SDC features. The results of the experiment were below the results of not only the SDC system but also the baseline GMM system that used cepstral and delta-cepstral features. This result also suggests that the increased performance obtained from the use of SDC features is not only a product of the temporal information but also of the delta coefficients which capture information about the speech dynamics.

The last major body of experiments to be reported consists of cross-corpus experiments. The results obtained by the system for the OGI corpus experiment exposed the system need for large amounts of training data to train robust LMs and the backend classifier. An experiment was developed to study the alternative of training the system using the CallFriend corpus, which includes a considerable amount of training and development data, while evaluating the system with the smaller OGI corpus. In this experiment, every component of the system is trained using the CallFriend data and the evaluation is performed with the 10-second and 45-second sets of the OGI corpus. The results obtained for this experiment were very poor, presumably due to poor acoustic matching between the corpora. A different experimental setup for the cross-corpus approach is presented in Appendix C.

This chapter included the development of the experimental system for language identification, along with a description of the results obtained. The next chapter summarizes the work and presents alternative for future studies.

Chapter 6

Summary, Conclusions and Future Work

This chapter summarizes the major results and contributions of the research including comments on strengths and weaknesses of the various system configurations investigated. Some possible paths for future work are also presented.

6.1 System Goals and Achievements

The main purpose of this work has been to study alternatives for the LID task that reduces the computational complexity required by previous systems, while also minimizing human expert intervention. In order to achieve these goals, a system has been developed that focuses on data-driven techniques to model the low-level acoustic events of the languages and to discriminate among languages by studying the statistics of these events.

Contrary to other successful approaches that focus on phonemes and phonotactics as their discriminatory tool, the approach developed here focuses on frame-level information about the speech for the discrimination task. Other approaches focusing on similar information have generally been unsuccessful. The current approach is only one of the few to adequately exploit short-term acoustic information for this task.

The system developed in this work also has the ability to combine temporal information of the speech, likely related to phonemes, and use this information to outperform systems based on phonetic information. Results for systems like PPRLM perform at 21.5% percent error rate compared to the 18.6% classification accuracy obtained for the SDC parallel GMM tokenization system for the CallFriend corpus.

The experiments conducted with a second corpus, the OGI corpus, resulted in lower performance than that obtained by the PPRLM system. The 45-second test results in a 18.69% error rate compared to a 10% for the PPRLM system and the 10-second test results in a 24.0% error rate compared to 21.0% for the same test using PPRLM. There are various explanations for this lower performance, the proposed system could result in improve performance if more training data were available or the PPRLM could provide better performance as the utterances are longer. Additionally it is possible that the acoustic matching between the phoneme recognizer and the data accounts for this performance but the same could be argued for the comparison of the systems on the CallFriend corpus.

6.2 Advantages

The system developed in this work has not only resulted in competitive performance to that of the PPRLM system but has also reduced the computation time. The PPRLM system performs its recognition tasks in about three times that needed for the implementation of the GMM system for the 512-order implementation. Further reductions in computation time are obtained by reducing the mixture order to 256 or 128. Performance on the CallFriend evaluation set using lower mixture orders is shown in Appendix A. These reductions result in 2% and 5% performance decreases, respectively. This lower performance can be acceptable in certain applications. Complete results for the evaluation of the system on the CallFriend corpus are included in the appendix.

The system also eliminates the need for any transcription of the speech. Contrary to phoneme-recognition systems, this system requires only knowledge about the language of the speech. This advantage allows the system to be fitted toward the particular conditions of interest by re-training with new speech examples.

6.3 Disadvantages

The major disadvantage of the GMM tokenization system is the amount of data needed to properly train the LMs and backend classifier. This fact has been explored using the OGI corpus where the mixture order had to be limited to 128 mixtures compared to a 512-order for the CallFriend set.

The availability of training data is also critical to build the backend classifier. The limitation in training data is one the possible reasons for the system under performing the phoneme-based systems for the OGI corpus for both testing sets.

Another issue related to the amount of training data is the reliance of the system on a backend classifier. Since the system relies on the backend classifier to provide an additional increase in performance, relatively large amounts of training data are necessary to reliably training the system.

6.4 Future Work

There are two major paths that could result in additional improvements in the current system. These paths include the selection of the tokenizing languages and the fusion of additional sources of information. For each of the cases there is already some empirical evidence that support this claim.

First, the selection of the tokenizing language could be performed for particular applications. The results presented in Chapter 5 show how the addition of some of the tokenizers erodes performance. In addition, some initial results are shown in Appendix D where for a given number N all possible combinations of tokenizers were computed. Results are shown for the best, worst and average error rates.

The fusion of different sources of information seems to be the most promising alternative for additional improvements on LID systems. Besides the results shown in this work, where the system is improved by including acoustic scores into the process, additional evidence exists that including phonotactic information as part of the fused elements improves the current error rates by an additional 5% [67].

A second source of information that could be exploited within the fusion framework is that derived from the language prosody. Possible language prosody measurements include rate of speech, pause durations and pitch contours. The use of pitch features in isolation for LID has not shown any performance advantages [12] but perhaps when combined with acoustic and tokenization information they may add new information and thus improve performance.

Additionally, recent results in the area of LID using higher orders Gaussian mixtures are showing promise. The initial results in this area have been shown by Singer [59]. This system uses the acoustic likelihoods obtained from GMM LMs to perform its discrimination task. Current results are comparable to PPRLM results while reducing drastically the computational burden.

6.5 Contributions

The major contributions of this research are as follows:

1. Introduced the use of the GMM for tokenization of speech for LID.
2. Provided classification results of the GMM-tokenization system with acoustic score fusion.
3. Introduced the use of information about the speech dynamics, through SDC features, as a means to discriminate across languages.

4. Provided results of systems capable of providing competitive performance with current systems while reducing the human intervention and providing real-time capabilities.

Appendix A

The purpose of this appendix is to present the results of the evaluation of the parallel GMM system with acoustic score fusion and using SDC features for different mixture orders. The results presented here represent faster implementations of the system described in Chapter 5. The system is evaluated using the CallFriend evaluation partition.

The plots present the results for the effect of adding tokenizers at each mixture order for orders of 256, 128 and 64.

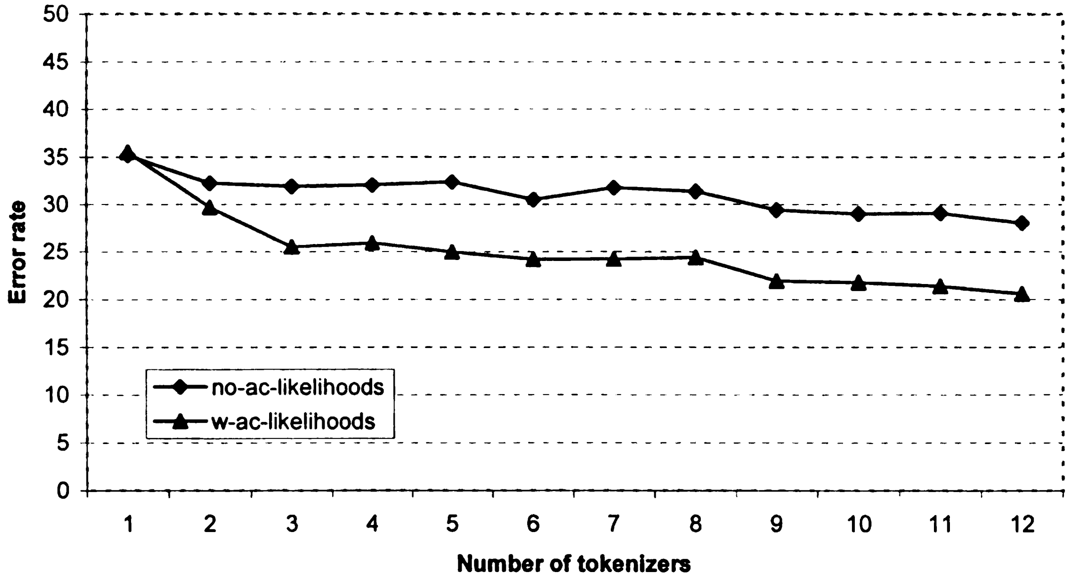

FIGURE A-1. SDC-based parallel GMM system using 256 mixtures.

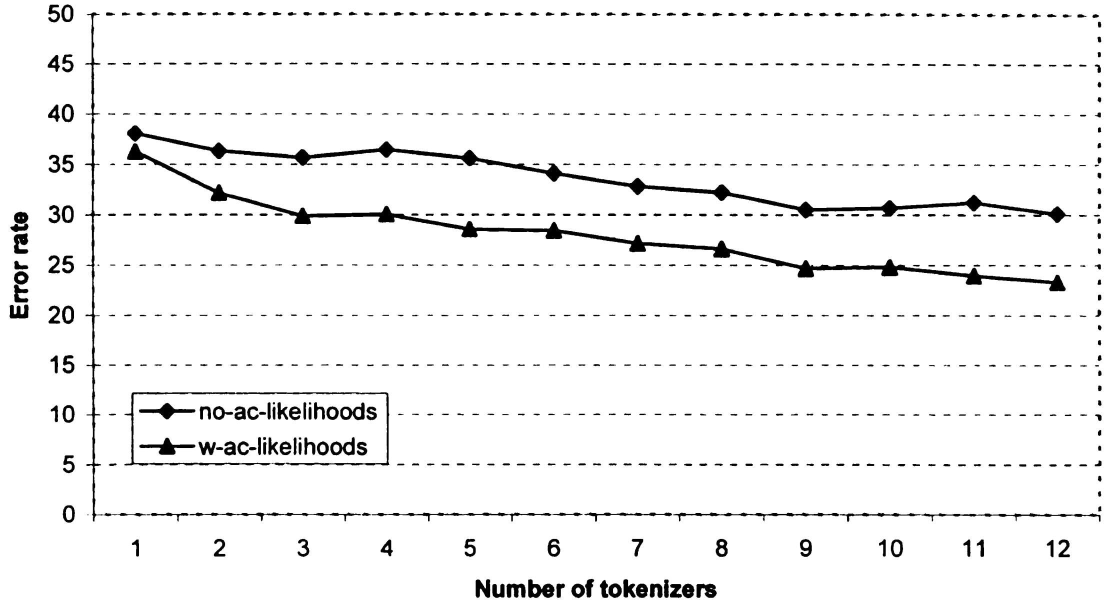

FIGURE A-2. SDC-based parallel GMM system using 128 mixtures.

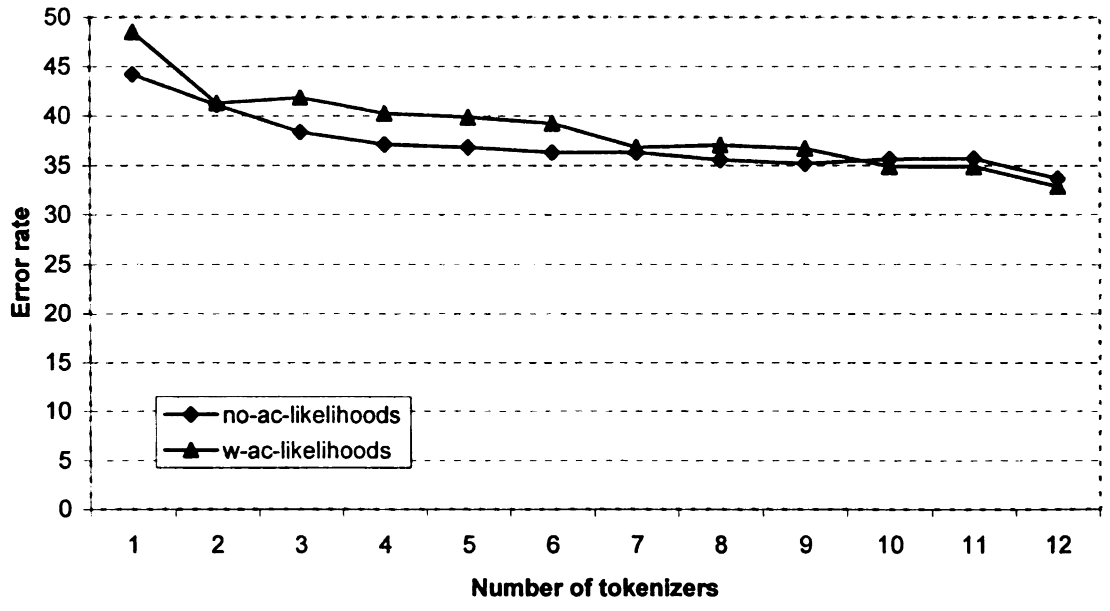

FIGURE A-3. SDC-based parallel GMM system using 64 mixtures.

There are two major observations about the systems shown in Figures A-1, A-2 and A-3. First, there is a clear trend of acoustic fusion enhancing the performance of the systems as additional tokenizers are added similar to the trend observed in Chapter 5 for the 512-order system. Second, the performance of the system increases with the mixture order. This trend was also observed for the single tokenizer systems described in Sections 5.3 and 5.5.1. These results are summarized in Figure A-4.

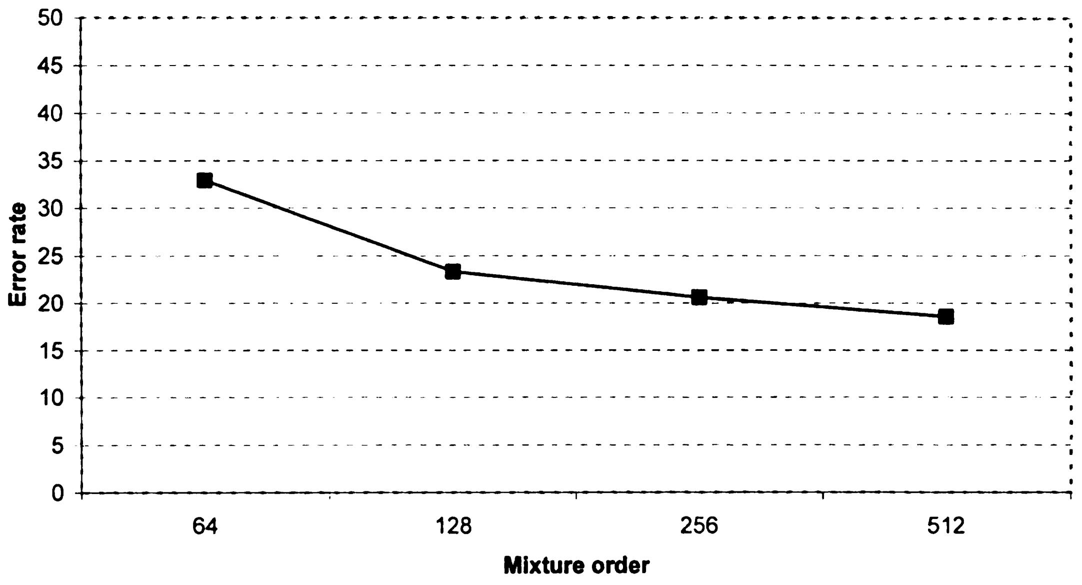

FIGURE A-4. SDC-based parallel GMM system performance as a function of the GMM order.

Appendix B

The purpose of this appendix is to present the results of the evaluation of the parallel GMM system with acoustic score fusion and using SDC features for the shorter speech segments of the CallFriend corpus. The results shown are for the evaluation of the system using both the 3-second utterances and the 10-second utterances. The parameters used for the system are the same as those used in section 5-7.

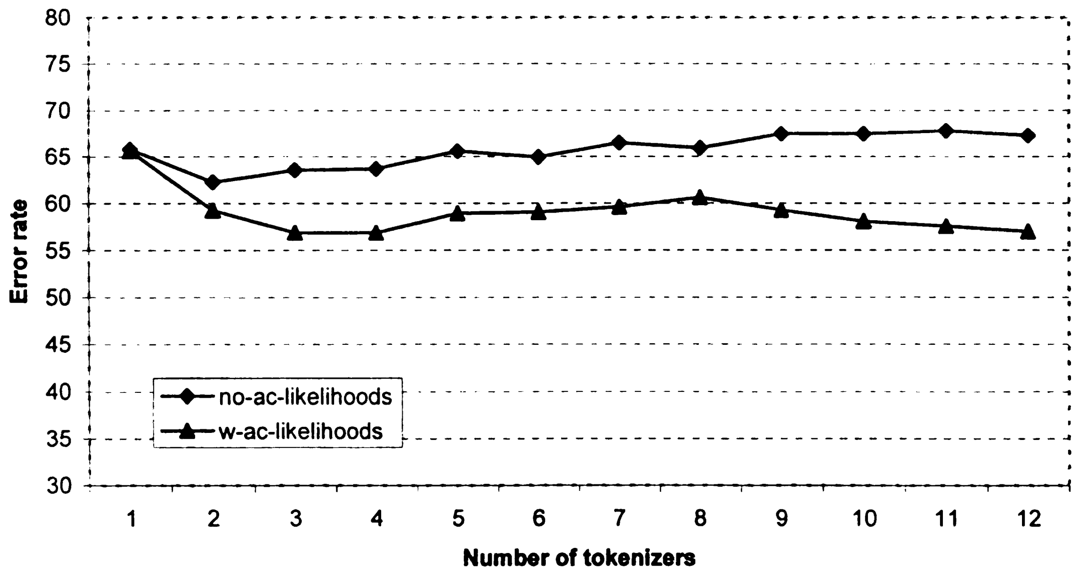

FIGURE B-1. SDC-based parallel GMM system performance of the system on the 3-second segments of the CallFriend evaluation set.

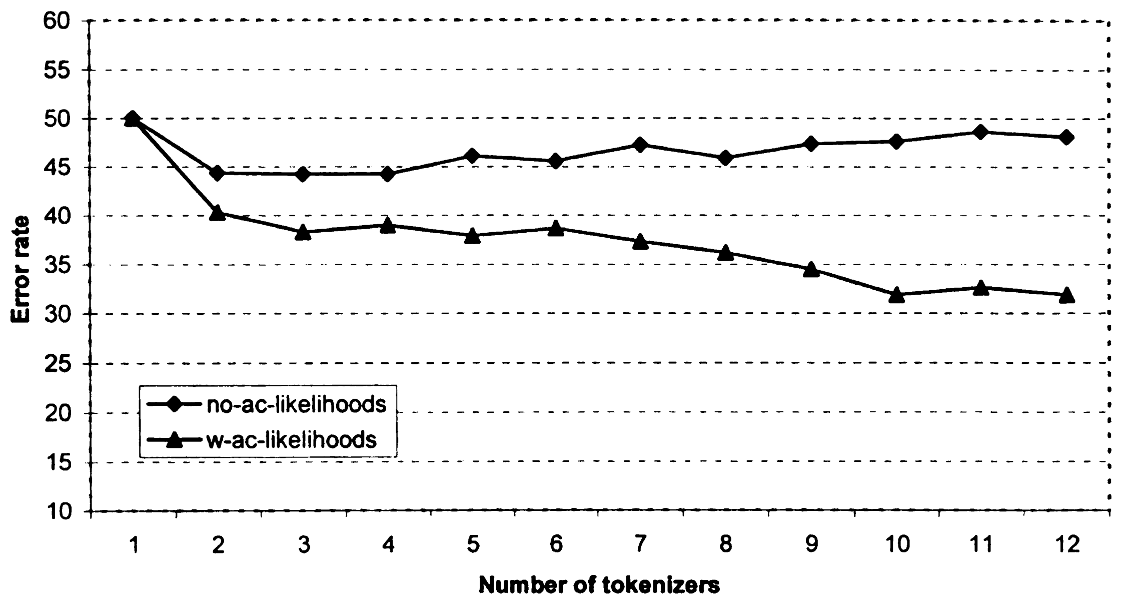

FIGURE B-2. SDC-based parallel GMM system performance of the system on the 10-second segments of the CallFriend evaluation set.

Appendix C

This appendix presents the results of using multiple corpora to train and evaluate the system. The parameters of this experiment are slightly different than those presented in Chapter 5. In the first experiment, the GMM tokenizer is trained on data from either the OGI corpus or the CallHome corpus while the bigram LMs are trained using the CallFriend TRAIN partition. Similarly to the experiments in Chapter 5, the backend classifier is trained with the development partition of the CallFriend corpus with the evaluation partition of the CallFriend corpus used for the evaluation of the full system. The order of the GMM trained is 512 with SDC parameters 10-1-3-3.

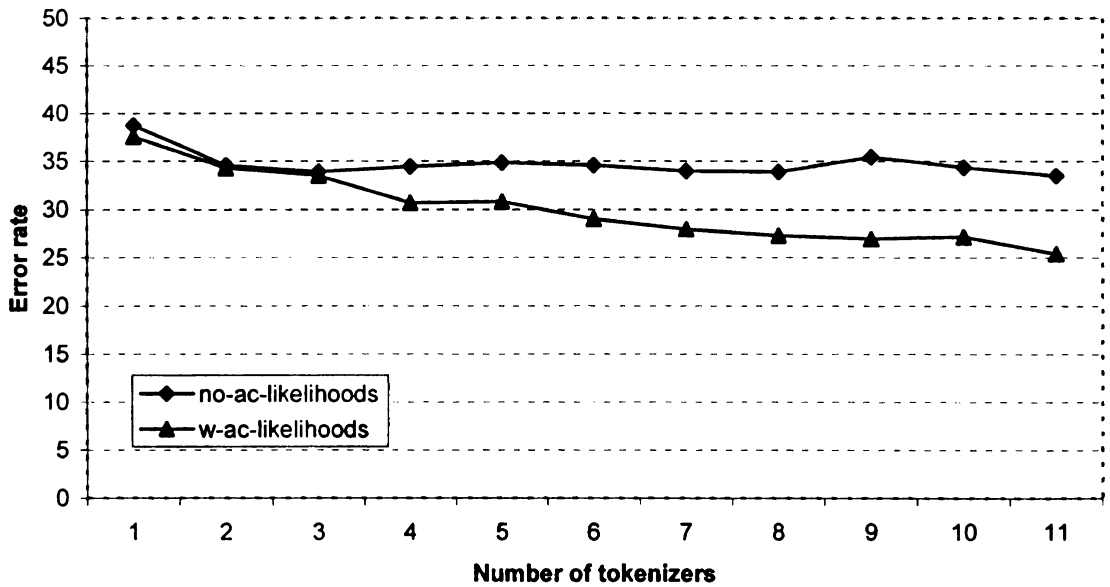

FIGURE C-1. SDC-based parallel GMM system performance of the system on the 30-second segments of the CallFriend corpus using OGI training data for the GMM tokenizer

The CallHome corpus is used to train the GMM tokenizers in the following experiment. The plot in Fig C-2 includes results for only six tokenizers as the CallHome corpus contains training data for only: Arabic, English, German, Japanese, Mandarin and Spanish.

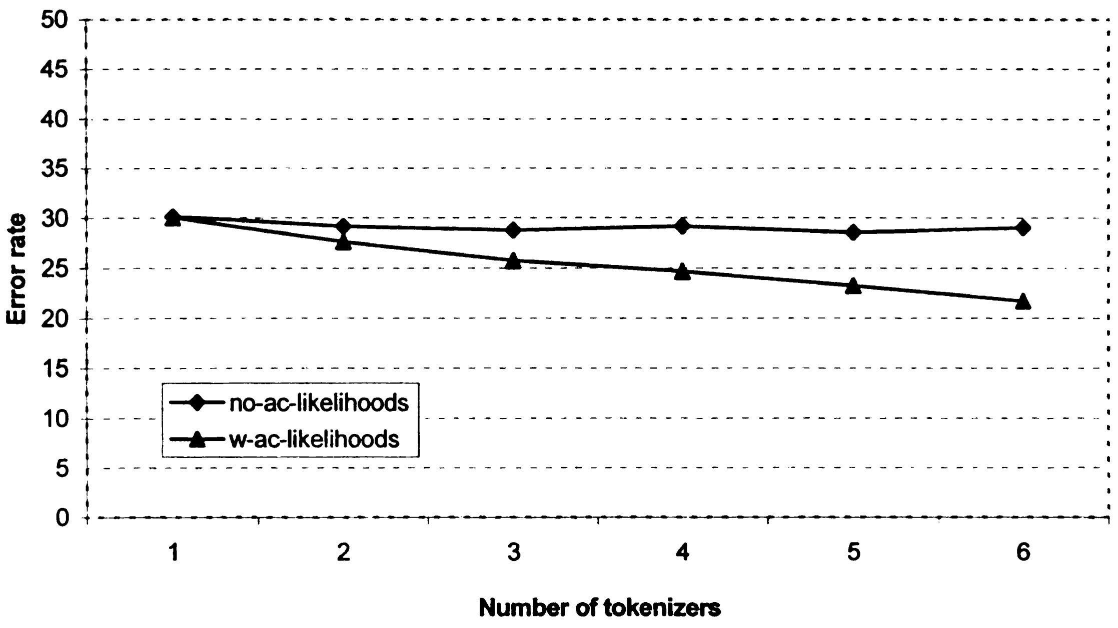

FIGURE C-2. SDC-based parallel GMM system performance of the system on the 30-second segments of the CallFriend corpus using CallHome training data for the GMM tokenizer

The second experiment presented in Appendix C presents the results of evaluating the system using the OGI corpus with cross-corpus approach. In this case, the CallFriend and CallHome sets are used to train the GMM tokenizers with the OGI corpus used for the evaluation of the system. The parameters utilized for this experiment are the same as those previously employed in the first set of experiments in Section 5.8.

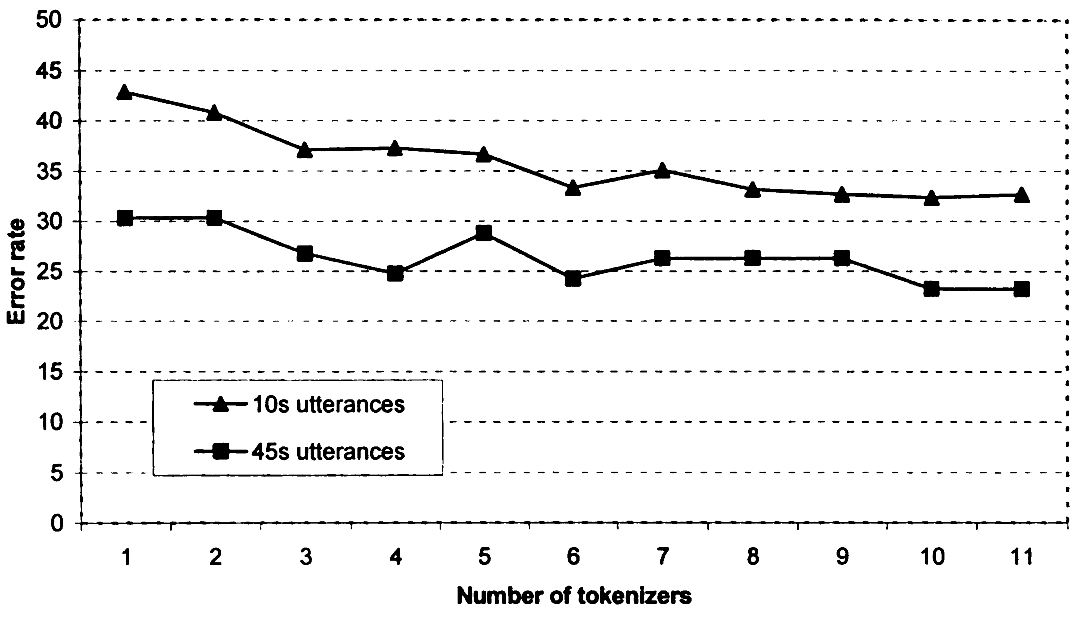

FIGURE C-3. SDC-based parallel GMM system performance of the system on the OGI corpus test set using CallFriend training data for the GMM tokenizer

For the plot in Figure C-4, the CallHome training data from Arabic is not included, resulting in a five-tokenizer system.

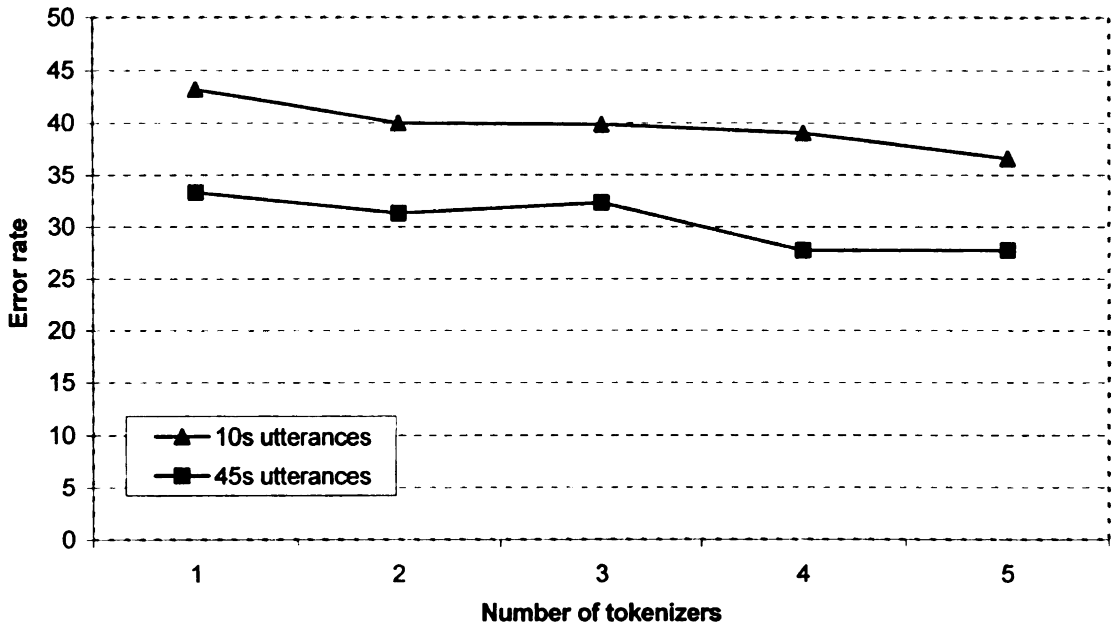

FIGURE C-4. SDC-based parallel GMM system performance of the system on the OGI corpus test set using CallHome training data for the GMM tokenizer.

Appendix D

This appendix shows the results of evaluating the system in Section 5-7 using all possible combinations of *N* tokenizers as *N* is varied. This result shows even more clearly than before that properly using the languages of the GMM tokenizers could result in at least similar performance to that of the full-system while reducing the system size and therefore the computational complexity.

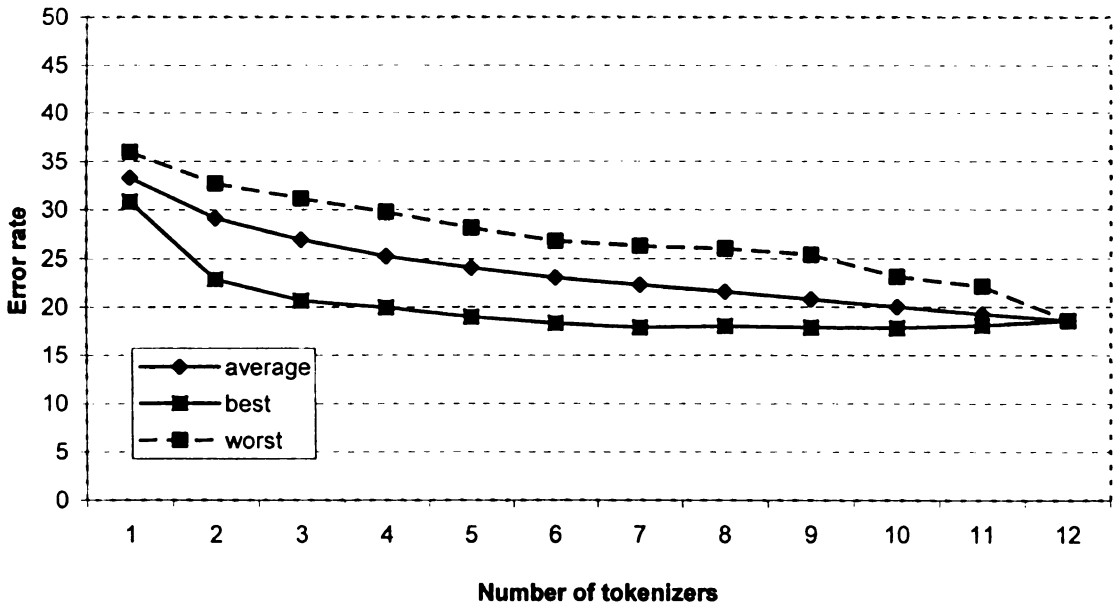

FIGURE D-1. SDC-based parallel GMM system performance of the system as different combinations of tokenizers are used. The plot shows the system best, average and worst combination of tokenizers results.

REFERENCES

[1] Y. K. Muthusamy, E. Barnard, and R. A. Cole, "Reviewing Automatic Language Identification," *IEEE Signal Processing Magazine*, vol. 11, pp. 33-41, 1994.

[2] J.-M. Hombert and I. Maddieson, "The Use of Rare Segments for Language Identification," In *Proc. European Conference on Speech Communication and Technology*, Seattle, Washington, 1999.

[3] M. Sugiyama, "Automatic Language Recognition Using Acoustic Features," In *Proc. International Conference on Acoustics, Speech and Signal Processing*, Toronto, Canada, 1991.

[4] J. R. Deller Jr., J. H. L. Hansen, and J. G. Proakis. *Discrete-Time Processing of Speech Signals*. IEEE Press, Piscataway, NJ, 2000.

[5] K.-P. Li, "Automatic Language Identification Using Syllabic Spectral Features," In *Proc. International Conference on Acoustics, Speech and Signal Processing*, Adelaide, Australia, 1994.

[6] K.-P. Li, "Experimental Improvements of a Language Id System," In *Proc. International Conference on Acoustics, Speech and Signal Processing*, Detroit, Michigan, 1995.

[7] S. Itahashi and L. Du, "Language Identification Based on Speech Fundamental Frequency," In *Proc. European Conference on Speech Communication and Technology*, Madrid, Spain, 1995.

[8] L. R. Rabiner, "A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition," *Proceedings of the IEEE*, vol. 77, pp. 257-286, 1989.

[9] S. Itahashi, T. Kiuchi, and M. Yamamoto, "Spoken Language Identification Utilizing Fundamental Frequency and Cepstra," In *Proc. European Conference on Speech Communication and Technology*, Seattle, Washington, 1999.

[10] M. Savic, E. Acosta, and S. K. Gupta, "An Automatic Language Identification System," In *Proc. International Conference on Acoustics, Speech and Signal Processing*, Toronto, Canada, 1991.

[11] S. E. Hutchins and A. Thymé-Gobbel, "Experiments with Prosody for Language Identification," In *Proc. Speech Research Symposium XIV*, Baltimore, Maryland, USA, 1994.

[12] F. Cummins, F. Gers, and J. Schmidhuber, "Language Identification from Prosody without Explicit Features," In *Proc. European Conference on Speech Communication and Technology*, Seattle, Washington, 1999.

[13] A. K. Jain, M. Jianchang, and K. M. Mohiuddin, "Artificial Neural Networks: a Tutorial," *Computer Magazine*, vol. 29, pp. 31-44, 1996.

[14] M. Barkat, J. Ohala, and F. Pellegrino, "Prosody as a Distinctive Feature for the Discrimination of Arabic Dialects," In *Proc. European Conference on Speech Communication and Technology*, Seattle, Washington, 1999.

[15] F. Pellegrino and R. André-Obrecht, "An Unsupervised Approach to Language Identification," In *Proc. International Conference on Acoustics, Speech and Signal Processing*, Phoenix, Arizona, 1999.

[16] F. Pellegrino and R. André-Obrecht, "From Vocalic Detection to Automatic Emergence of Vowel Systems," In *Proc. International Conference on Acoustics, Speech and Signal Processing*, Munich, Germany, 1997.

[17] A. S. House and E. P. Neuberg, "Toward Automatic Identification of the Language of an Utterance," *Journal of the Acoustical Society of America*, pp. 708-713, 1977.

[18] L. F. Lamel and J.-L. Gauvain, "Language Identification Using Phone-based Acoustic Likelihoods," In *Proc. International Conference on Acoustics, Speech and Signal Processing*, Adelaide, Australia, 1994.

[19] K. M. Berkling, T. Arai, and E. Barnard, "Analysis of Phoneme-based Features for Language Identification," In *Proc. International Conference on Acoustics, Speech and Signal Processing*, Adelaide, Australia, 1994.

[20] T. Hazen and V. Zue, "Automatic Language Identification Using a Segment-Based Approach," In *Proc. European Conference on Speech Communication and Technology*, Berlin, Germany, 1993.

[21] T. Hazen and V. Zue, "Recent Improvements in an Approach to Segment-Based Automatic Language Identification," In *Proc. International Conference on Spoken Language Processing*, Yokohama, Japan, 1994.

[22] F. Jelinek and R. L. Mercer, "Interpolated Estimation of Markov Source Parameters from Sparse Data," In *Proc. Workshop on Pattern Recognition in Practice*, North Holland, Amsterdam, 1980.

[23] F. Jelinek. *Statistical Methods for Speech Recognition*. MIT Press, Cambridge, Massachusetts, 1999.

[24] M. A. Zissman and E. Singer, "Automatic Language Identification of Telephone Speech Messages Using Phoneme Recognition and N-Gram Modeling," In *Proc. International Conference on Acoustics, Speech and Signal Processing*, Adelaide, Australia, 1994.

[25] M. A. Zissman, "Language Identification Using Phoneme Recognition and Phonotactic Language Modeling," In *Proc. International Conference on Acoustics, Speech and Signal Processing*, Detroit, Michigan, 1995.

[26] CallFriend Corpus, Linguistic Data Consortium, 1996, http://www.ldc.upenn/ldc/about/callfriend.html

[27] Y. Yan and E. Barnard, "An Approach to Automatic Language Identification Based on Language-dependent Phone Recognition," In *Proc. International Conference on Acoustics, Speech and Signal Processing*, Detroit, Michigan, 1995.

[28] Y. Yan and E. Barnard, "An Approach to Language Identification with Enhanced Language Model," In *Proc. European Conference on Speech Communication and Technology*, Madrid, Spain, 1995.

[29] Y. Yan and E. Barnard, "Experiments with Conversational Telephone Speech for Language Identification," In *Proc. International Conference on Acoustics, Speech and Signal Processing*, Atlanta, Georgia, 1996.

[30] Y. K. Muthusamy, R. A. Cole, and B. T. Oshika, "The OGI multi-language telephone speech corpus," In *Proc. International Conference on Spoken Language Processing*, Alberta, Canada, 1992.

[31] H. K. Kwan and K. Hirose, "Recognized Phoneme-based N-gram Modeling in Automatic Language Identification," In *Proc. European Conference on Speech Communication and Technology*, Madrid, Spain, 1995.

[32] P. Dalsgaard, O. Andersen, H. Hesselager, and B. Petek, "Language-identification Using Language-dependent Phonemes and Language-independent Speech Units," In *Proc. International Conference on Spoken Language Processing*, Philadelphia, USA, 1996.

[33] J. Navratil and W. Zuhlke, "Phonetic-context Mapping in Language Identification," In *Proc. European Conference on Speech Communication and Technology*, Rhodes, Greece, 1997.

[34] O. Andersen, P. Dalsgaard, and W. Barry, "On the Use of Data-Driven Clustering Technique for Identification Of Poly- and Mono-Phonemes for Four European Languages.," In *Proc. International Conference on Acoustics, Speech and Signal Processing*, Adelaide, Australia, 1994.

[35] O. Andersen and P. Dalsgaard, "Language-identification Based on Cross-Language Acoustic Models and Optimized Information Combination," In *Proc. European Conference on Speech Communication and Technology*, Rhodes, Greece, 1997.

[36] J. Navrátil and W. Zühlke, "Double Bigram-Decoding in Phonotactic Language Identification," In *Proc. International Conference on Acoustics, Speech and Signal Processing*, Munich, Germany, 1997.

[37] J. Navratil and W. Zuhlke, "An Efficient Phonotactic-acoustic System for Language Identification," In *Proc. International Conference on Acoustics, Speech and Signal Processing*, Seattle, Washington, 1998.

[38] T. Schultz, I. Rogina, and A. Waibel, "LVCSR-based Language Identification," In *Proc. International Conference on Acoustics, Speech and Signal Processing*, Atlanta, Georgia, 1996.

[39] S. Kadambe and J. L. Hieronymus, "Language Identification with Phonological and Lexical Models," In *Proc. International Conference on Acoustics, Speech and Signal Processing*, Detroit, Michigan, 1995.

[40] S. Kadambe and J. L. Hieronymus, "Spoken Language Identification Using Large Vocabulary Speech Recognition," In *Proc. International Conference on Spoken Language Processing*, Philadelphia, USA, 1996.

[41] S. Kadambe and J. L. Hieronymus, "Robust Spoken Language Identification Using Large Vocabulary Speech Recognition," In *Proc. International Conference on Acoustics, Speech and Signal Processing*, Munich, Germany, 1997.

[42] D. Matrouf, M. Adda-Decker, L. F. Lamel, and J.-L. Gauvain, "Language Identification Incorporating Lexical Information," In *Proc. International Conference on Spoken Language Processing*, Sydney, Australia, 1998.

[43] M. A. Zissman, "Comparison of Four Approaches to Automatic Language Identification of Telephone Speech," *IEEE Transactions on Speech and Audio Processing*, vol. 4, 1996.

[44] H. Hermansky, N. Morgan, A. Bayya, and P. Kohn, "Rasta-PLP Speech Analysis Technique," In *Proc. International Conference on Acoustics, Speech and Signal Processing*, San Francisco, California, 1992.

[45] S. B. Davis and P. Mermelstein, "Comparison of Parametric Representations for Monosyllabic Word Recognition in Continuously Spoken Sentences," *IIEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 28, pp. 357-366, 1980.

[46] D. A. Reynolds and R. C. Rose, "Robust Text-independent Speaker Identification Using Gaussian Mixture Models," *IEEE Transactions on Speech and Audio Processing*, vol. 3, pp. 72-83, 1995.

[47] D. A. Reynolds, *A Gaussian Mixture Modeling Approach to Text-independent Speaker Identification.* Atlanta, GA, 1992.

[48] D. A. Reynolds, "Automatic Speaker Recognition Using Gaussian Mixture Speaker Models," vol. 8, pp. 173-192, 1995.

[49] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum-likelihood from Incomplete Data Via the EM Algorithm," *Royal Statistics Society*, pp. 1-38, 1977.

[50] L. R. Rabiner and B. H. Juang. *Fundamentals of Speech Recognition.* Prentice Hall, Englewood Cliffs, New Jersey, 1993.

[51] R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern Classification.* Second ed. Wiley & Sons, New York, 2001.

[52] B. Bielefeld, "Language Identification Using Shifted Delta Cepstrum," In *Proc. Fourteenth Annual Speech Research Symposium*, 1994.

[53] F. Bimbot, R. Pieraccini, E. Levin, and B. Atal, "Variable-length Sequence Modeling: Multigrams," *IEEE Signal Processing Letters*, vol. 2, 1995.

[54] S. Deligne and F. Bimbot, "Language Modeling by Variable Length Sequences: Theoretical Formulation and Evaluation of Multigrams," In *Proc. International Conference on Acoustics, Speech and Signal Processing*, Detroit, Michigan, 1995.

[55] S. Deligne and F. Bimbot, "Inference of Variable-length Acoustic Units for Continuous Speech Recognition," In *Proc. International Conference on Acoustics, Speech and Signal Processing*, Munich, Germany, 1997.

[56] S. Deligne and F. Bimbot, "Inference of Variable-length Linguistic and Acoustic Units by Multigrams," *Speech Communication*, vol. 23, 1997.

[57] J. Rissanen, "Modeling by Shortest Data Description," *Automatica*, vol. 14, pp. 465-471, 1978.

[58] R. M. Gray, "Vector Quantization," *IEEE Acoustics, Speech and Signal Processing Magazine*, vol. 1, pp. 4-29, 1984.

[59] E. Singer, R. Greene, M. A. Kohler, and D. A. Reynolds, "Automatic Language Identification Using Gaussian Mixture Models," In *Proc. Submitted for publication to International Conference on Acoustics, Speech and Signal Processing*, Orlando, FL, 2002.

[60] M. A. Zissman, "Automatic Language Identification Using Gaussian Mixture and Hidden Markov Models," In *Proc. International Conference on Acoustics, Speech and Signal Processing*, Minneapolis, Minnesota, 1993.

[61] CallFriend, Linguistic Data Consortium, 1996, http://www.ldc.upenn/ldc/about/callfriend.html

[62] M. A. Zissman, "Predicting, Diagnosing and Improving Automatic Language Identification Performance," In *Proc. European Conference on Speech Communication and Technology*, Rhodes, Greece, 1997.

[63] A. K. Jain, *Introduction to Pattern Recognition (class notes).* Michigan State University, 1999.

[64] K.-F. Lee and H.-W. Hon, "Speaker-Independent Phone Recognition Using Hidden Markov Models," *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. 37, pp. 1641-1648, 1989.

[65] M. A. Zissman, "Automatic Language Identification of Telephone Speech," *Lincoln Laboratory Journal*, vol. 8, pp. 115-144, 1995.

[66] E. S. Parris and M. J. Carey, "Language Identification Using Multiple Knowledge Sources," In *Proc. International Conference on Acoustics, Speech and Signal Processing*, Detroit, Michigan, 1995.

[67] P. A. Torres-Carrasquillo, D. A. Reynolds, and J. R. Deller Jr., "Language Identification Using Gaussian Mixture Model Tokenization," In *Proc. International Conference on Acoustics, Speech and Signal Processing*, Orlando, Fl. , USA, 2002.