A COST, COMPLEXITY AND PERFORMANCE COMPARISON OF TWO AUTOMATIC LANGUAGE IDENTIFICATION ARCHITECTURES

by

HENDRIK PETRUS COMBRINCK

BEng (Electronic Engineering)

Submitted in partial fulfilment
of the requirements for the degree

MASTER OF ENGINEERING

Department of Electric and Electronic Engineering
Faculty of Engineering
University of Pretoria

November 1999

Title:	A Cost, Complexity and Performance Comparison of Two Automatic Language Identification Architectures
Author:	Hendrik Petrus Combrinck
Supervisor:	Prof. E.C. Botha
Department:	Department of Electrical and Electronic Engineering
Degree:	Master of Engineering (Computer)

Abstract

This dissertation investigates the cost-complexity-performance relationship between two automatic language identification systems. The first is a state-of-the-art architecture, trained on about three hours of phonetically hand-labelled telephone speech obtained from the recognised OGI_TS corpus. The second system, introduced by ourselves, is a simpler design with a smaller, less complex parameter space. It is a vector quantisation-based approach which bears some resemblance to a system suggested by Sugiyama. Though trained on the same data, it has no need for any labels and is therefore less costly. A number of experiments are performed to find quasi-optimal parameters for the two systems. In further experiments the systems are evaluated and compared on a set of ten two-language tasks, spanning five languages. The more complex system is shown to have a substantial performance advantage over the simpler design - 81% versus 65% on 40 seconds of speech. However, both results are well under reported state-of-the-art performance of 94% and would suggest that our systems can benefit from additional attention to implementation detail and optimisation of various parameters. Given the above, our suggested architecture may potentially provide an adequate solution where the high development cost associated with state-of-the-art technology and the necessary training corpora are prohibitive.

Uittreksel

Hierdie verhandeling ondersoek die verwantskap tussen twee outomatiese taalherkenningstelsels in terme van koste, kompleksiteit en werkverrigting. Die een stelsel is gebaseer op 'n tegnologie-spits argitektuur en word afgerig op ongeveer drie uur se foneties handgemerkte telefoonspraak verkry uit die OGI_TS korpus. Die kompeterende stelsel wat ons voorstel, is 'n eenvoudiger ontwerp met 'n kleiner, minder komplekse parameterruimte. Dit is 'n vektorkwantiserings-gebaseerde benadering wat in sekere opsigte ooreenstem met 'n vorige stelsel van Sugiyama. Hoewel ons stelsel op dieselfde data afgerig word, hoef die data nie gemerk te wees nie; ons stelsel is gevolglik heelwat goedkoper. Stelselwerkverrigting word geoptimeer deur 'n gedeelte van die parameterruimte ekperimenteel te ondersoek. Die kwasi-optimale stelsels word in verdere eksperimente met mekaar vergelyk oor 'n stel van 10 twee-taal herkenningstake wat vyf tale onderspan. Die komplekse stelsel lewer heelwat beter werkverrigting as die eenvoudiger alternatief - 81% versus 65% op 'n 40 sekonde uiting. Daar moet egter in gedagte gehou word dat beide resultate beduidend swakker is as gepubliseerde tegnologie-spits werkverrigting van 94%. Dit sou dus wou voorkom asof ons stelsels verbeter kan word deur meer aandag aan implementeringskwessies en optimering van verskeie parameters te skenk. Met hierdie feite inaggenome kan ons voorgestelde stelsel potensieel handig te pas kom in situasies waar die ontwikkelingskoste van tegnologie-spits stelsels, en die gepaardgaande spraak korpora, andersins beperkend so wees.

Acknowledgements

First of all, deep thanks to my thesis supervisor, Professor Liesbeth Botha, for unending optimism, dedication and general perseverance inducing behaviour.

I salute the courage of Herman le Roux and Febe de Wet who, with me, boldly stepped into the unknown world of spoken language processing through the looking glass that Professor Botha presented. I thank them for their companionship during the time that we explored together. Thanks to Febe for casting hidden Markov model theory in LaTeX.

Thanks to Charl Barnard and Darryl Purnell for two generations of UNIX system administration. My thanks also to Darryl and Christoph Niewoudt for the use of their hidden Markov model software.

I would also like to thank the first generation of CEFIM people that I shared space, thoughts and sweat with, including Gavin Ehlers, Francois Lessing, Gert van Tonder, Rudolph Pienaar and Louis Coetzee. These people helped me greatly in taking my first uncertain steps in the then bewildering world of networks, UNIX and LaTeX.

And special thanks to my good friend Karl Geggus for many, many discussions and shared insights on the nature of Life, the Universe and Everything throughout the time that I have worked on this dissertation.

To my parents, Steph and Hester Combrinck,

who taught me things not found in books.

Arrakis teaches the attitude of the knife – chopping off what's incomplete and saying: "Now it's complete because it's ended here."

– Muad'dib in *Dune* by Frank Herbert

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 The problem

The goal of *Automatic Language Identification* (ALI) is to classify a speech signal as belonging to one of a number of previously encountered languages – *automatic*, because the task is performed by a machine. As with all spoken language activities, people can identify languages familiar to them with exceptional speed and accuracy. Their skill tends to disguise the inherent complexity of the problem. Human speech, apart from being massively redundant, contains information about the sex, age, unique vocal tract, socio-economical background, native geographical location, emotional status, nationality and language of the speaker. In addition, the signal is altered by background noise as well as distortion introduced by the communication channel. As a matter of fact, if one works with the whole audio band accessible to the human auditory system, the intended symbolic message constitutes only about 0.1% of the total information content of a speech signal [1]. When extracting language-specific information, it is implied that the process is robust with regard to the avalanche of additional information. Automatic language identification is not a trivial problem and a successful solution will have to draw from a number of different fields of expertise

– human physiology and psychology, acoustics, information theory, statistics, pattern recognition, phonetics and linguistics – all forged into a working system in a way that is worthy of the term *engineering*.

We will discuss the context and possible applications of ALI in Section 1.2. It is followed by a brief overview of the basic approach to the problem in Section 1.3 and a look at ALI state-of-the-art in Section 1.4. Section 1.5 explains our contribution to ALI and we conclude with a brief outline of the dissertation in Section 1.6.

1.2 The context

The first question that comes to mind, maybe, is why do we need ALI technology at all? As an immediate practical example, in a multi-lingual society such as South Africa, automatic language identification can be useful in an emergency service context, where callers who phone in are normally in shock and prone to confused babbling in their native tongue. An ALI system can switch callers to an operator that can understand them without wasting valuable time in trying to establish which language they are using.

The longer answer is that ALI is a vital part of any multilingual spoken language system. These will eventually include public speech-based information retrieval systems, services that handle hotel bookings, real-time translation systems and the like. Such applications are only starting to receive attention as they become more feasible, but in a shrinking global community they will by necessity eventually enjoy high priority. Already a central aim of technology is to develop so-called SILKy interfaces to machines that are locked on a path of ever-increasing complexity. SILK is an acronym for Speech, Image, Language and Knowledge, the enabling technologies for truly intuitive man-machine interfaces.

South Africa has a large number of education-deprived people who find it difficult to

function in a high-technology environment. In order to remain globally competitive, the people of South Africa will have to adapt, but we need technology to access technology. Spoken language systems might provide part of the answer by placing the burden of human-machine interaction on the machine. In a country with 11 national languages, ALI may eventually play a vital part. The basic approach to this enabling technology is outlined in the next section.

1.3 The approach

Although people can sometimes recognise a familiar language even from a partially spoken word, they find it difficult to express exactly how they achieve this feat. An obvious difference among languages is that of different words, but when confronted with an unknown language, it is near-impossible to tell where words begin and end; yet people still manage a better than chance performance. Perceptual experiments provide some clues [2, 3, 4, 5]. Subjects may find an unknown language similar to a known one, or describe it as nasal (French), harsh (German), sing-song (Mandarin), rhythmic or guttural. In addition, speakers are sensitive to sounds not found in languages familiar to them, like the click-sounds in a number of southern African languages, or the velar fricatives in Arabic. Apart from the obvious differences in the sound inventories of languages, there are more subtle patterns in the frequency of occurrence of certain sounds and combinations thereof. Hawaiian for instance, has a very small number of consonants. The cluster /sr/ is very common in Tamil, but not found in English at all. Hindi has four different consonants that are all likely to sound to native English speakers like their own /t/.

The challenge of ALI then, is to identify and exploit these differences in a systematic way. In brief, state-of-the-art systems achieve their goal by means of a phone recognition front-end that transcribes the speech stream into known speech units or phones.

A language modelling subsystem extracts statistical information about phone distributions and the results are used for classification. The process will be described in detail in Chapter 3. The next section considers the current state of ALI technology.

1.4 State of the art

Although ALI research can be traced back for at least twenty years, very little work has been done until 1993 (Muthusamy mentions a total of fourteen papers [4]). Then, with the work of Muthusamy, the OGI Telephone Speech (OGI_TS) Corpus [4, 6] was put in the public domain and adopted by the American National Institute of Standards and Technology (NIST) as the standard for evaluating ALI algorithms. It contains a large amount of telephone speech in 11 languages. (NIST had organised an annual evaluation event for state-of-the-art ALI systems from 1992 to 1996, but has unfortunately discontinued this useful contribution to ALI research.) A subsequent explosion in the field produced a flood of papers and a number of groups working full-time on the problem. The intensity of the attack leaves automatic language identification today as a mature field. A number of systems that can identify around 10 languages with high accuracy (approximately 80% recognition rate for a 10-language task on a 45 second utterance) have been demonstrated [7].

Research is now turning to the more difficult problems of accent and dialect recognition and ALI system that does not need a labelled speech corpus for training. Chapter 2 elaborates on existing ALI literature and practice. The next section covers our contribution to ALI research.

1.5 Contributions of this dissertation

Although state-of-the-art ALI systems perform very well, they require large hand-labelled speech corpora. Such corpora are typically very expensive and difficult to obtain. In addition these systems do not scale very well. Both factors are serious concerns in the light of envisioned systems eventually handling hundreds of languages. We address these issues by introducing an alternative approach that can use unlabelled data for training and we quantitatively compare its performance to current ALI technology.

We implemented two ALI systems. The systems share the same general structure. A feature extraction front-end operating on a speech stream is followed by a transcription block that takes the feature stream as input and produces a symbol (or token) stream. The token stream is analysed statistically, extracting language specific information describing its probabilistic syntax, which is used to classify the speech sample. The systems differ in the nature of the tokens that are used.

The first is a state-of-the-art architecture, referred to as *Parallel Phone Recognition followed by N-gram Language Modelling* (PPRLM). It uses hand-labelled speech data to train a phone recogniser during the training phase. When testing the system, the phone recogniser produces a phone string for processing by the N-gram language modelling block. The phone recognition is performed with a Continuous Density Hidden Markov Model (CDHMM).

Our system (the second system implemented) produces a string of equal-length sub-phonetic units as tokens. Whereas the phonetically-based system needs to be trained from labelled data, the sub-phonetic approach uses vector quantisation (VQ) to "discover" suitable sub-phonetic units and consequently has no need for labelled data. Having independently arrived at this solution, it was found to bear some resemblance to the work of Sugiyama [3, 8]. He suggested two VQ-based systems. We are only

concerned with the second, more effective one. The effort documented here differs in three important respects from Sugiyama's approach.

- We use a separate VQ codebook for each language, as opposed to a universal codebook spanning all the system languages.
- The way that we utilise the histograms describing the occurrence of token N-grams is completely different.
- Our system is tested on the recognised OGI_TS corpus, using more than five hours of speech from 416 speakers in five languages as opposed to the 40 minute, 153 speaker, 20 language corpus used by Sugiyama.

The state-of-the-art PPRLM architecture yields significantly better results then our VQ-based approach, but at the price of vastly increased effort needed to label a training corpus. The latter, not requiring human intervention, trades reduced complexity and effort for performance. In addition, when adding a new language to the first system, one can use the existing phonetic classifiers of previously incorporated languages to perform suboptimal transcription of the new language. By contrast, when integrating a new language into the second system, it can take full advantage of the new information.

1.6 Organisation of dissertation

First of all, in Chapter 2, we will review existing ALI research. Chapter 3 examines the theory behind ALI, which will allow as to present our systems in more detail. This is followed by a description of experiments and results in Chapter 4. Chapter 5 concludes with a discussion of the results and some pointers for future work.

Chapter 2

Previous work

2.1 Introduction

Muthusamy reviews ALI research prior to 1993 thoroughly in [4]. There are more recent reviews by Muthusamy *et al.* [9], Zissman [7] and Berkling [10]. Later work is dominated by the sustained efforts of two or three groups.

Because few of the early attempts used the same corpora and much of the research details (number and gender of speakers, recording conditions and quality, languages used, test and training set division, etc.) were somewhat nebulous, it is rather difficult to compare these attempts quantitatively. Still, they provide interesting ideas. Most of these approaches used some form of unsupervised training to estimate a set of optimal system parameters. Recently, with the availability of the OGI_TS corpus, ALI systems that use multilingual, phonetically labelled corpora, have received much more attention and have delivered good results.

Section 2.2 continues with a discussion of perceptual research. We then turn to a number of different approaches to the ALI previously studied in Sections 2.3, 2.4, 2.5 and 2.6. Significant trends are highlighted in Section 2.7, followed by conclusions in

Section 2.8.

2.2 Perceptual studies

Spoken language is generated *by* the human speech system, *for* the human auditory system; consequently machines perform rather badly in most aspects of spoken languages processing when compared to human beings. Since we are trying to mimic a human ability, studying people engaged in ALI might provide us with valuable clues. Earlier perceptual studies on language identification [2, 3, 4, 5] are scarce in engineering-oriented literature. This might be largely due to the pattern recognition approach to spoken language problems: in the same way that one does not need to fly like a bird in order to fly, one does not need to process speech like the human speech centre in order to recognise it. It does become increasingly clear, however, that human speech perception is extremely complex and operates in non-obvious ways. We briefly review results from a number of studies. Muthusamy *et al.* reports on two experiments; since the second [5] was more comprehensive and the trends similar to that of the first, we discuss only the second experiment.

28 subjects (14 male, 14 female) listened to and classified one-, two-, four- and six-second excerpts of telephone speech from 10 languages. The subjects included native speakers of all the languages. The languages were English, Farsi, French, German, Japanese, Korean, Mandarin, Spanish, Tamil and Vietnamese. The average classification performance over all subjects and languages as a function of the duration of the excerpts rose from 44.8% (one second) to 65.3% (six seconds). The average performance over all subjects, languages and durations was 56.7%, with individual language scores ranging from 28.3% for Korean to 93.1% for English. Performance of listeners seems to be a strong function of the number of languages known; 44.1% for speakers who knew one language up to 66.7% for speakers who knew four.

After the experiments, the subjects were debriefed in an attempt to determine how they performed the classification task. It seems that subjects used a combination of phoneme- and word-spotting as well as phonetic and prosodic cues. German, for instance, sounded "harsh" (maybe because of velar fricatives) and had the distinctive word "ich". The tonal quality of Mandarin was important and sometimes confused with Japanese. French had several nasal sounds and distinct intonation.

The experiments suggest that humans integrate data from diverse information sources and that they can identify languages using very short utterances. Unfortunately the duration of the excerpts were not extended to the durations used when comparing automatic systems. (The NIST evaluation uses 10 second and 45 second phrases.) This would have enabled direct comparisons between humans and automatic systems. As it is, Muthusamy states that human performance asymptotes for much shorter durations of speech than automatic systems [9]. Although humans are very good with familiar languages, their performance vary greatly over a number of variables. People who know ten or more languages are few and far between, and for tasks with a large number of languages, machines probably already outperform humans. Future systems might handle tens to hundreds of languages which will eventually rule out any human competition.

More recently, Stockmal, Bond *et al.* performed a number of perceptual experiments [11, 12, 13]. In an earlier paper [11] they report on two studies designed to explore which perceptual properties inherent within the phonological structure of languages are salient to foreign language listeners. In the first of these, fifteen subjects were asked to judge whether pairs of spoken foreign language sentences were selected from same or different languages and to explain how they had made the judgement. Multi-dimensional scaling (MDS) was conducted on the subject responses for the "same language" condition. The resulting map suggested that responses could be characterised along two dimensions: phonologically based psychoacoustic properties (prosodics) and talker specific characteristics (voice quality and speech rate).

In the second study this perceptual feature relationship was tested using similarity judgements. Thirty subjects rated similarity on a seven-point scale for the same set of sentence pairs that had been judged in study one. MDS analysis of the "different language" condition yielded a map in which the language relationships closely approximated those which had been derived by focusing on phonological properties.

A later study [12] explored the attributes of languages to which listeners attend, using magnitude estimation and multi-dimensional scaling techniques. In magnitude estimation, listeners assign any numerical value to a set of stimuli. In response to the question: "How similar is this language to English?" fifty college students assigned numerical values to spoken samples of foreign languages. The languages represented Europe, Asia and Africa. Differences between the mean ratings for each language and English were used to construct a proximity matrix which was submitted to MDS analysis. The optimum solution employed three dimensions. The first dimension was interpreted as "familiarity," the second as "speaker affect," and the third as "prosodic pattern." The MDS maps suggest that listeners were using English as a standard of comparison to the acoustic-phonetic properties of other languages.

Typically, in studies like these, foreign language samples have been provided by different speakers so that language and speaker characteristics could be confused. This problem is addressed in [13]. Three experiments were conducted using the same speaker for different pairs of languages. Listeners were able to discriminate between two unknown languages, even when spoken by the same speaker, showing that listeners can distinguish speaker characteristics from language characteristics. The experiment also suggested that prosodics was the major determining factor.

It would seem then that humans can accurately identify languages known to them using sometimes less than one second of speech. They can learn to identify unknown languages and they use various strategies and cues that range over different classes of information and different levels of organisation inherent in a spoken language signal.

The next section examines some attempts to build ALI systems that mimic this human ability in the most basic sense, using only simple acoustic features.

2.3 Unsupervised training with acoustic features

The simplest and earliest approaches to the ALI problem attempted to use the raw speech signal or a simple transformation thereof to train language classifiers. These systems were unsupervised in the sense that the features were not labelled to be used in a two-stage process where the features are first identified and then presented to a second stage where language identification takes place. Our VQLM system would fit roughly into this category.

In what probably constitutes the first sustained ALI effort, Leonard and Doddington [14, 15, 16, 17] used spectrally-based features to build a dictionary of reference sounds for each language. They assumed that a language can be characterised by chunks of sound that occur more frequently in one language than in competing candidates. By detecting these reference sounds in an utterance, the most likely language can be found. They moved from single sounds to sequences of sounds and from automatic to manual selection of reference sounds. The automatic selection was done using various information-theoretic measures. The first three studies used read speech from 100 adult males over five languages, equally divided into training and test sets. The languages, recording conditions and sampling frequency were not specified. Results ranged from 60% to 100% for different language pairs with an overall recognition rate of 64% on 60s of speech for the first study. The results were subsequently improved to 88% in study 2 and 80% in study 3 (where reference sounds were manually selected). They added two more languages in study 4, resulting in a total of 66 speakers in the training set and 65 in the test set. This system achieved a 62% recognition rate. Using a confidence measure to reject test utterances with a low confidence level, recognition

accuracy was raised to 100% at a 68% rejection level.

Cimarusti and Ives [18] extracted frame-based features derived form Linear Predictive Coding (LPC) analysis. These included autocorrelation coefficients, cepstral coefficients, filter coefficients, log area ratios and formant frequencies. The features were used to find an optimal polynomial decision function used for classification. The data consisted of three minutes of read speech (10kHz sampling frequency) from five adult male speakers of each of the following languages: American English, Czech, Farsi, German, Korean, Mandarin, Russian and Vietnamese. Using randomly divided training and test sets (size unknown), they achieved an overall recognition rate of 84%.

Foil [19] used formant vectors as features and vector quantisation (k-means clustering) to find the 10 best formant vector clusters for each language. A vector quantisation distortion measure was used for classification. 10 hours of data from three unspecified languages were recorded from radio (SNR of 5dB). The number of speakers were not specified. Using 5s of speech for an identification decision, they managed a 39% recognition rate. The addition of a confidence measure boosted accuracy to 64% at an 11% rejection rate.

Goodman *et al.* [20] extended Foil's work, making a number of improvements to all aspects of the system and increasing performance by more than 50% on Foil's data set. The system was also tested on three other data sets. The one set was a six language database of noisy speech (SNR of 9dB). Further details concerning the data were not available.

Savic *et al.* [21] used features derived from hidden Markov models (HMMs) and pitch contours that they integrated with a voting scheme. Each language was modelled by a five-state linear predictive HMM. It was found that the HMM states roughly correspond to articulatory states of the vocal tract. Preliminary results showed considerable inter-language variation in the transition probabilities. The data used was ten minutes of read speech in each of four languages: English, Hindi, Mandarin and Spanish. It was

recorded in a noise-free room, sampled at 10kHz and low-pass filtered at 4.5 kHz. The number of speakers, train/test set division and classification results are unspecified.

Sugiyama [3, 8] tested two approaches. In the first a vector quantisation codebook of acoustic features was created for each language. The test utterance was then quantised with each language codebook and accumulated quantisation distortion was calculated. The utterance was classified as belonging to the language with minimum VQ distortion. (This is similar to Foil and Goodman's work.) The second approach used a universal codebook created from all the training data. Each language was characterised by an occurrence probability histogram. The test utterance was quantised using the universal codebook and the Euclidean distance between its occurrence probability histogram and the language specific histograms (previously calculated) were used for classification. The data was taken from a multilingual corpus, CCITT SG-XII, distributed by NNT, Japan and consisted of 16 sentences (8s duration) uttered twice by four male and four female speakers in each of the following 20 languages: American English, Arabic, Mandarin, Danish, Dutch, (British?) English, Finnish, French, German, Greek, Hindi, Hungarian, Italian, Japanese, Norwegian, Polish, Portuguese, Russian, Spanish and Swedish. There were a total of 76 male and 77 female speakers (some languages did not have eight speakers). The first technique had a recognition rate of 65% on 64s test utterances and the second, 80%.

Nakagawa *et al.* [22] used acoustic features and examined four different HMM approaches. The first method used VQ and was identical to that of Sugiyama. The two HMM systems used discrete and continuous HMMs respectively. Each language was modelled using a single ergodic HMM (with a varying number of states). The continuous HMM used a single Gaussian mixture, while they experimented with different numbers of mixtures in the fourth, a Gaussian mixture model system. For the latter three systems, the likelihood for each language was accumulated over all feature frames and used for classification. The continuous HMM and GMM systems were shown to outperform the VQ and discrete HMM approaches. The data was generated by fif-

teen native male speakers for each of four languages: English, Japanese, Mandarin and Indonesian. There were 50 sentences per speaker with an average duration of 3 seconds, sampled at 12 kHz (SNR of 49.2dB). The training set consisted of about 300 utterances of 10 speakers in each language and the test set of about 100 utterances from the remaining 5 speakers in each language. The classification performance for the various techniques were 81.1% for the continuous HMM and GMM approaches and 77.4% and 47.6% respectively for the VQ and discrete HMM systems. With enhancements to the continuous HMM system, it reached 86.3%. The addition of five more languages (French, German, Korean, Malay and Russian) reduced the recognition rate of the original continuous HMM system to 48.0%.

Kwasny *et al.* [23] used raw speech data with a neural network classifier in a simple two-language, two-speaker experiment. The experiments were continued with a recurrent neural network in [24, 25]. In the first study the data consisted of eight read utterances (12.5 seconds each) generated by two bilingual speakers in French and English (2 utterances per speaker, per language). The classifier achieved 100% classification. The last study used four speakers (still on a English-French task). The results are not really meaningful on such small samples.

More recently Zissman [26, 7] investigated the use of GMMs and HMMs similar to the work of Riek [27] and Nakagawa. These experiments were done more recently, using the OGI_TS corpus. Experimental details are given in Sections 3.7.2 and 3.7.5. We also used the OGI_TS corpus in our experiments and describe it in Section 4.2.

Du Preez and Weber [28] reported good results using high-order HMMs (i.e. HMMs that have memory about previous states visited). Though still new, this seems to be a promising approach to high-quality unsupervised systems. Experiments were performed on free-format English and Hindi utterances from the OGI_TS corpus. A hundred minutes of each language were used for training. The system achieved 79.8% and 97.4% recognition rate on 5s utterances (247 trials) and 45s utterances (39 trials)

Researchers	Languages	Data	Duration	Results
Leonard, Doddington	7	66 train, 65 test	60s	62%
Cimarusti, Ives	8	40 male	Unknown	84%
Foil	3	600 minutes total	5s	39%
Goodman *et al.*	Unknown	Unknown	Unknown	Unknown
Savic *et al.*	4	40 minutes total	Unknown	Unknown
Sugiyama	20	76 male, 77 female	64s	80%
Nakagawa	4	40 train, 20 test	3s	86%
Kwasny *et al.*	2	1 male, 1 female	12.5s	100%
Zissman	10	OGI_TS	45s	53%
Du Preez, Weber	2	OGI_TS	45s	97%

Table 2.1: Automatic language identification systems using acoustics features and an unsupervised training approach.

respectively.

The systems discussed here, summarised in Table 2.1, were mostly the result of earlier studies and represent a wide variety of different approaches and experimental conditions. The work of Sugiyama deserves, maybe, special attention. The results (80% recognition rate) seem to be impressive given the large number of languages (20). The other salient study is that of Du Preez and Weber. High-order HMMs might just manage to marry the necessary complexity demanded of ALI systems with the the low cost associated with data-driven systems.

2.4 Phone recognition with language modelling

In search of better performance, researchers were relentlessly driven to increasing complexity and more comprehensive modelling of language on higher levels of organisation. This section gives an overview of systems that incorporate *a priori* knowledge about the phonetic and phonotactic structure of spoken language.

House and Neuberg [29] used phonetically transcribed text to show that languages could be classified using broad phonetic categories (stop, fricative, vowel, silence). They modelled a language as a Markov process emitting a stream of symbols belonging to broad phonetic categories and assumed that the parameters of the models could be estimated from enough training data. The data consisted of manually transcribed phonetic texts for eight languages: American English, Chinese, Greek, Japanese, Korean, Russian, Swahili and Urdu. They achieved perfect discrimination.

Li and Edwards [30] extended the work of House and Neuberg to real speech data. They investigated broad phonetic category segments as well as syllable-based Markov models. The system handled five (unspecified) languages; two Asian and three Indo-European. The data was all-male, read speech recorded under unspecified conditions at an unspecified sample rate. The training set consisted of four minutes of speech from ten speakers for each of the five languages. The testing set contained two minutes of speech per speaker. The highest recognition rate was 80%, obtained with the syllable-based model.

D'Amore and Mah [31] and Schmitt [32] used N-gram analysis of text to do language and topic identification and clustering. Albina *et al.* [33] extended this technique to speech.

Tucker *et al.* [34] and Lamel and Gauvain [35] used monolingual phone recognisers to label multilingual training corpora that could then be used to build language-dependent phone recognisers for ALI.

Muthusamy [4] examined broad phonetic features, amongst others, in his comprehensive Phd dissertation. Muthusamy *et al.* [36] compared acoustic features, broad-category segmentation and fine phonetic classification using multiple monolingual neural network phone recognisers. They found fine phonetic classification to outperform the other two approaches on a English-Japanese identification task. The data was taken form the OGI_TS corpus, with a training set of 85 utterances and a test set of

30. The best results for acoustic features, broad-category and fine phonetic features were respectively 70.0%, 83.2% and 86.3%.

In stead of using language-specific sets of phones for multiple monolingual phone recognisers, Andersen and Dalsgaard [37, 38, 39, 40] and Berkling *et al.* [41, 42, 43] experimented with inter- and intra-language clustering of phones to produce a group common to all classifiers (poly-phonemes) and sets of language-dependent phones (mono-phonemes) for each classifier. Amongst others they aimed to optimise discriminative, as opposed to representative, information content. Dalsgaard and Anderson tested variants of their system on a three way classification task (American English, German and Spanish) using the OGI_TS corpus and achieved 88% and 83% respectively in [38] and [39].

Hazen and Zue (OGI_TS, 10 languages, 45s duration, 47.7% recognition rate) [44], Zissman and Singer [45] and Tucker *et al.* [34] have also experimented with monolingual front-end phone recognisers. This work was extended by Zissman and Singer [45] and Yan and Barnard (OGI_TS, 11 languages, 77.1% recognition rate on 10s and 90.8% on 45s utterances) [46] to multiple monolingual front-ends where there need not be as many front-ends as languages to be classified. They also introduced a high-level language model optimisation scheme and in [47] duration modelling of phones were found to be useful. The system was tested on six languages in the OGI_TS corpus (English, German, Hindi, Japanese, Mandarin and Spanish) and achieved 81.1% and 92.0% on 10s and 45s utterances respectively.

Hazen and Zue (OGI_TS, 11 languages, 79.7% recognition rate) [48] as well as Lamel, Gauvain *et al.* [49] experimented with a single, multilingual front-end. The latter group used the IDEAL telephone speech corpus, containing four languages: British English, Spanish, French and German, with about 19 hours of speech per language. Their best recognition rate of 91% on 10s utterances was not significantly better than the 90% of their system with multiple monolingual front-ends.

Researchers	Languages	Data	Duration	Results
House, Neuberg	8	Text-based	Unknown	100%
Li, Edwards	5	50 speakers	120s	80%
Muthusamy	2	OGI_TS	45s	86%
Andersen, Dalsgaard	3	OGI_TS	45s	88%
Yan, Barnard	11	OGI_TS	45s	91%
Hazen, Zue	11	OGI_TS	45s	80%
Lamel, Gauvain	4	IDEAL corpus	10s	91%
Berkling, Barnard	2	OGI_TS	45s	93%
Navratil, Zuhlke	9	OGI_TS	45s	91%

Table 2.2: Automatic language identification systems using phone recognition followed by language modelling.

Berkling [10] and Berkling and Barnard [50] investigated variable length, inaccurate phone sequences (as opposed to fixed N-grams). A language is too complex to be modelled (represented) with a single Markov or hidden Markov model, but one can attempt to extract significant (discriminative) portions of the model. So, although the model will not represent the language completely, the language will have a very definite and possibly unique effect on the parameters of the model. Using the OGI_TS corpus they achieved 93% correct classification on the English-German language pair. Parris *et al.* [51, 52] experimented with much the same idea.

Navratil and Zuhlke [53] examined a way to improve phonotactic probability estimations and looked at language models based on binary decision trees. Using 9 languages from the OGI_TS corpus, they achieved 77.4% and 90.6% on 10s and 45s utterances respectively.

Kadambe and Hieronymus [54] used a "lexical access module" to spot language-specific patterns in the phone sequences.

The research documented in this section (as summarised in Table 2.2) shows clearly the seemingly unavoidable movement towards increased complexity. At the same

time, some of the work (like that of Berkling) already show an awareness of this problem and attempt to move beyond it.

2.5 Prosody-based features

Prosody is concerned with the "music" as opposed to the "lyrics" of speech. Spoken language have characteristic sound patterns that can be analysed in terms of **duration**, **pitch** and **stress**. The efforts mentioned here have tried to use this information to identify language.

Foil [19] and Savic *et al.* [21] examined the use of pitch contours. Muthusamy [4] experimented with pitch, speech rate, syllabic timing and segmental duration. Hazen and Zue [44, 48] used fundamental frequency (F_0) contours and segment duration as part of their system. Using a Phone Recognition followed by Language Modelling (PRLM)-type system they achieved 77.5% on a 11-language task (OGI_TS corpus) for 45s utterances. A prosodic duration model only managed 44.4% on the same task and the F_0-model 20.9%. An integrated system, using all the components had a recognition rate of 79.1%.

Itahashi and Du [55] also investigated fundamental frequency.

Hutchins and Thyme-Gobbel [56] obtained good results using rhythmic and tonal characteristics. They recognised four main language categories: stress-timed (for instance English), syllable-timed (Spanish), mora-timed (Japanese) and tone languages (Mandarin). Using data from the OGI_TS corpus representing these four languages, they investigated a set of 220 prosodic and derived features and showed that successful features depend very much on the prosodic nature of the languages under consideration [57]. The best results on different language pairs with different feature sets ranged from 71% to 86%. This is substantially lower than results achieved with PRLM approaches [7].

Although perceptual experiments show that prosodic information plays an important role when humans classify unknown languages [5, 11, 12, 13], it seems to be difficult to utilise this information in an effective way in automated systems. While in some cases producing reasonable results on their own, prosodic features add very little performance on top of phone-based systems.

2.6 Other approaches

Finally, there are a number of approaches that are unique or otherwise difficult to classify. We present them in this section.

Ives [58] built an expert system using production rules operating on formant-based features. 50 distinguishing features obtained from experts were converted to numerical thresholds using patterns in the training data. Based on these thresholds, a minimum set of nine production rules were designed and used for classification. His database consisted of 50 hours of speech from 122 male speakers in eight languages: American English, Czech, Farsi, German, Korean, Mandarin, Russian, Vietnamese. The data was sampled at 10kHz and low-pass filtered at 5kHz. The experiments were performed on a total of 720 five-second utterances and the system had a recognition rate of 92%.

Li [59] imported ideas from speaker identification. Features from the test utterance are compared to those in each language of the N closest matching speakers of that language. The utterance is classified as belonging to the language of the closest matching speaker.

Hieronymus and Kadambe [60], Schultz *et al.* [61] and Mendoza *et al.* [62] investigated ALI using Large Vocabulary Continuous Speech Recognition (LVCSR) systems. This approach requires a fully-fledged LVCSR system for each of the target languages. The utterance is presented to each of the systems which returns a transcription and

a likelihood score. The one with the highest likelihood score represents the chosen language. This approach is very resource hungry in terms of labelled data, word dictionaries for all the languages and raw processing power for the continuous speech recognition task. If the LVCSR systems already exist, though, it can be a very powerful solution. Hieronymus and Kadambe tested their system on a 5-language task (English, German, Spanish, Japanese and Mandarin) using the OGI_TS corpus. The best recognition rates were 93% for 10s utterances and 98% for 45s utterances. Mendoza *et al.* achieved respectively 97.3% and 98.3% on 10s and 45s utterances, also using the OGI_TS corpus in a three-way task (English, Japanese and Spanish). These results are substantially better than those reported for 3- and 5-language tasks in Section 2.4.

Kwan and Hirose used a neural network [63] and recurrent neural network [64] with a unigram histogram of phones as feature vector to flag and reject unknown languages in a system where the input is not limited to a set of known alternatives.

Matrouf *et al.* [65] presented a pragmatic hybrid approach to ALI. They incorporated an N-most-frequently-used word recogniser into a standard phone-based language model system. They exploit the principle that the most frequently used words in a language account for a large proportion of all word occurrences; the 100 most frequently used words in a language may easily constitute in the excess of 40 percent of all word occurrences. The data was taken from the IDEAL corpus, containing British English, French, German and Spanish. The best result was 92% for 5s spontaneous speech utterances, incorporating the 500 most frequently used words for each language. The recognition rate was slightly higher (96%) for read and elicited speech.

Yan and Barnard [66] examined the possibility of adapting existing ALI systems to new languages by modifying the phonotactic language models, using only a limited amount of training data. They trained a 9-language system with only 60% of the available data and used a fully-trained 6-language system to adapt the system parameters. Adapting the 9-language system in this way improved performance from 87.3% to 91.9% on 45s

Researchers	Languages	Data	Duration	Results
Ives	8	122 speakers	5s	92%
Hieronymus, Kadambe	5	OGI_TS	45s	98%
Mendoza *et al.*	3	OGI_TS	45s	98%
Matrouf *et al.*	4	IDEAL corpus	5s	96%
Yan, Barnard	9	OGI_TS	45s	92%

Table 2.3: Additional, uncategorised approaches to automatic language identification.

utterances. This compared well with the 9-language baseline performance of 92.5%.

Reynolds *et al.* [67] investigated the problem of clustering utterances according to language and/or speaker characteristics when no information regarding the speaker or language classes is available.

The results of the different studies documented in this section are summarised in Table 2.3. The two most promising concepts are those of LVCSR-based systems investigated in three independent studies and the work of Yan and Barnard on adapting existing systems with a small amount of data from additional languages. The next section examines trends emerging from the work presented in the four previous sections.

2.7 Trends

Berkling [10] points out a number of trends in the structure of ALI over the last few years. One of these is the move from low-level (spectral) features to high-level features (fine-phonetic or even words) and then again to lower-level features (clustered phones). This observation implies the distinction between acoustic features (low level) and structural features (high level). The acoustics features are derived directly from sampled speech waveforms and are infinitely variable. There is vague statistical structure in these features. On a next level (that of phones), the variability is much less –

the acoustic features are now vector quantised to a small set of possible symbols. The much stronger statistical patterns observed on this level are captured by phonotactics. On a next level (that of words) one finds that the statistical relationships become so strong as to be deterministic in some cases. Most words in a language are unique to that language and can therefore be used to identify the language conclusively. Yet, the ability to recognise words, as opposed to phones, imply (much) more *a priori* information to compensate for the brittle nature of such systems. That information must reside in the system in one way or another and leads to increased complexity. Currently, much work focuses on reducing the complexity, while retaining the information with the most discriminative power [10].

2.8 Summary

Although trends begin to emerge from ALI research, a significant number of possible approaches remains on the detail level of implementation. Obvious processing blocks like hidden Markov models, Gaussian mixture models, neural networks and N-gram language modelling subsystems are ubiquitous, but the way in which these blocks are used in conjunction with a large number of signal pre-processing techniques, make for diverse and very complex systems. The fact that many such systems are only described on conceptual level in publications, makes duplication of experiments difficult. Meaningful research therefore requires a sustained effort and solid infrastructure. The most prominent, sustained efforts during the past few years have been those of the *OGI Centre for Spoken Language Understanding* [9] and *MIT Lincoln Labs* [7].

Since the basic problem is well-studied and solutions (at least in concept) abound, research becomes more specialised and turns to more practical issues like task independence [68], discrimination between similar languages, dialects and accents [69, 70, 71],

uniform performance over languages, system complexity as well as efficiency and extensibility. Our work is primarily concerned with system cost and complexity, finding a simple, cost-effective approach to the problem that still yields acceptable performance.

The next chapter presents the theory behind automatic language identification.

Chapter 3

Theory of automatic language identification

3.1 Introduction

In this chapter we give a detailed description of the theory behind ALI systems. The concepts that are covered are general enough to encompass nearly all the systems mentioned in Sections 2.3 and 2.4 as well as the two systems that we have implemented. We show how the individual components function and how they interact to form an ALI system. Apart from the fundamental linguistics theory, treated in Section 3.2, there are three main components involved: feature extraction (Section 3.3), token alignment or phone recognition (Section 3.5) and language or phonotactic modelling (Section 3.6). In addition we discuss vector quantisation in Section 3.4 and provide an overview of ALI architectures in Section 3.7. Section 3.8 concludes the chapter with a summary.

3.2 Spoken language

Language can be thought of as a set of conventions that a group of people use to capture and communicate concepts. In a general sense language can be seen to exist independent of speech, since it is entirely possible to communicate using symbols unrelated to speech. Consider for instance Chinese and Japanese icons, Egyptian pictograms, the ancient Peruvian knot language, Quipu (that used coloured threads knotted together) or ASL (American Sign Language) used by deaf people. Of course, on the other hand we know that language has evolved hand in hand with speech – some words for instance being the result of sound imitation. In our context, language refers to a fairly stable set of speech signals used for communication by a group of people. Let $\Lambda = \{\lambda_1, \lambda_2, \lambda_3, \ldots, \lambda_N\}$ be the set of N languages under consideration.

3.2.1 Linguistics

Now consider a person (the **information source**) communicating by means of spoken language. His vocal cords and speech channel serve as a **transmitter** by generating variations in air pressure that are propagated outwards. We measure the variation in pressure a short distance from his mouth and call this the speech signal $s(t)$. It is a continuous, analogue, stochastic signal, carrying vast amounts of information in a complex, non-linear way. The signal passes through a **channel**, $H(\cdot)$, that may in general distort it in a time-varying, non-linear way. This includes various types of noise. Let the modified signal be $x(t) = H(s(t), t)$. The auditory system of the person listening, serves as **receiver** and the awareness of the person as **destination.** In all spoken language processing applications, the receiver and destination are replaced by a machine. For any language λ_i, we can expect $x(t)$ to be constrained in various characteristic ways.

When studying language, a most fundamental distinction is that between **form** (means

of communication) and **meaning** (the goal of communication). We will throughout this dissertation be concerned with the form, rather than the meaning (semantics) of an utterance. Proceeding from this distinction, the form of a language can (linguistically) be studied on two levels, those of **grammar** and **phonology** respectively. Grammar is concerned with the structure of a language on the level of **morphemes,** (i.e. words, loosely speaking) representing concepts, and **syntax,** the rules that dictate the way in which the words are strung together. Phonology studies **phonemes** or the basic sound units of speech, **phonotactics**, the syntax of phonemes, and **prosodics**, the "music" as opposed to the "lyrics" of speech.

We describe these terms in more detail below. The interested reader is referred to [72, 73, 74, 75] for an introduction to linguistics.

Phonetics. Phonetics study speech sounds (phones) as physical entities on a sub-language basis. The speech signal $x(t)$, $T_0 \leq t \leq T_{N-1} + d_{N-1}$, can be thought of as a concatenation of N segments $v_i(t)$ of duration d_i, $i = 0, 1, \ldots N-1$ with $(T_{i-1} + d_{i-1}) \leq T_i \leq t \leq (T_i + d_i) \leq T_{i+1}$, i.e. the segments are non-overlapping and not necessarily touching. Each segment (phone) can be mapped to a phoneme in the context of a certain language.

Phonology. Though the human speech system is potentially capable of an infinite range of sounds, there exist in any language only a limited number of recurrent, fairly distinctive speech units. Such an inventory of speech units differ widely from one language to another and phonology studies these units in the context of a certain language, λ_i. Let $\Phi_i = \{\phi_{i1}, \phi_{i2}, \phi_{i3}, \ldots, \phi_{iN_i}\}$ be the set of N_i phonemes that occurs in language λ_i. The number of phonemes, N_i, in a language ranges form about 15 to 50, with a peak at 30.

Phonotactics. Not only does phoneme inventories differ from language to language, but the frequency distributions of phonemes and combinations of phonemes are also very distinctive. Some combinations that occur frequently in one language

may be illegal in another. Phonotactics is concerned with the constraints that a language places on the sequential occurrence of phonemes and can be used to recapture some of the dynamical nature of speech lost during feature extraction. It is also the principle way in which machines distinguish one language from another. We will mostly be concerned with phonotactics in a statistical sense, i.e. probabilities of the form $P(\phi_{ij_t}|\phi_{ij_{t-1}}, \phi_{ij_{t-2}}, \ldots, \phi_{ij_{t-T+1}})$ for some language λ_i.

Prosodics. Languages have characteristic sound patterns that can be analysed in terms of **duration**, **pitch** and **stress**. The study of these patterns is called prosodics. Prosodic features are sometimes referred to as suprasegmental, because they are not confined to phonetic segments. Changes in pitch, or the melody of an utterance, is referred to as intonation. Used in English to communicate emotion, as well as semantic and pragmatic information, it distinguishes between otherwise identical words in tone languages like Mandarin or Xhosa. In stress languages, like English, one syllable in most words has a heavy stress or "accent" that sets it off from the other syllables.

Morphology. Morphology studies the way in which words are built up from the smallest meaningful parts in a language. If a system is capable of recognising words, the words can be checked against a language-specific dictionary and used to identify a language conclusively. Such an approach is paid for by loss of robustness and flexibility.

Syntax. The ways in which words can be legally strung together is studied under the label *syntax*, and constitute distinctive information that is utilised by Large Vocabulary Continuous Speech Recognition (LVCSR) systems.

A few words about the potentially confusing terms phonology, phonetics, phoneme and phone are in order. **Phonetics** is concerned with the study of speech sounds without reference to any particular language. Rather, speech sounds are analysed with re-

spect to their articulation (articulatory phonetics), transmission (acoustic phonetics) and perception (auditory phonetics). These sound units, called **phones**, are physical entities classified strictly according to their acoustic properties and can therefore be described without knowing which language they belong to. Phonetics is general and descriptive. **Phonology**, on the other hand, is particular and functional. It studies the exploitation of sounds in a specific language. From a phonological point of view sound units are called **phonemes**. Phonemes are abstractions that only have meaning in the context of a particular language. Two phonetically different phones in the same environment that distinguishes between different words, are recognised as different phonemes. The phones [r] and [l] are phonemes in English because they distinguish between words pairs like *lamb - ram, light - right* and *lobe - robe.* These are called **minimal pairs**, since they differ only enough to be recognised as distinct words and show that [r] and [l] are in **contrast**. The two phones are represented by the phonemes /r/ and /l/ respectively. In many languages, however, [r] and [l] either do not both occur, or do not distinguish between minimal pairs. In Chinese and Japanese for instance, the difference between [r] and [l] is not phonemic. Speakers of these languages find it extremely difficult to distinguish between [r] and [l]; they simply perceive it as the same sound and generate it interchangeably. Similarly, the English /l/ in *leaf* and *field* are actually phonetically two different sounds, called "clear" and "dark" respectively. They are recognised as different phonemes in Russian amongst others, though they sound very much alike to English speakers. Such phonetically different sounds, that are recognised as one phoneme in a language are called **allophones**.

3.2.2 Discussion

We have presented a brief introduction to relevant linguistic terms and concepts. The pattern recognition approach to spoken language processing tends to trivialise the value of detailed psychological, physiological and linguistic understanding of speech.

Though this is changing, a common engineering approach seems to be that the distinction between phones and phonemes, for instance, are not of much importance in real-life systems. However, to us it would seem that the evolution of ALI systems show that these differences are in fact important. It should for instance be clear from the example in the previous section that an English phoneme recogniser should distinguish between [r] and [l], while a Japanese system should not. Speech perception is an extremely complex subject, much of which is still not understood. Unfortunately it has received little attention in a ALI context [38, 7]. In the next section we scratch the surface of this fascinating world.

(Please note that mathematical symbols do not in general retain their meaning over section boundaries.)

3.3 Auditory perception and feature extraction

Perceived speech is a function of the speech signal and transmission channel as well as the human auditory system. From a spoken language point of view, human speech perception is studied under the label of auditory phonetics, a daunting mix of physics and physiology. From an engineering point of view we want to process the speech signal in a way that allows us to extract the relevant information concisely. The mel-scaled cepstrum is a popular feature extraction scheme that incorporates some perceptual modelling while maintaining efficiency.

The pre-processing and feature extraction stages of a pattern recognition system serve as an interface between the real world and a classifier operating on an idealised model of reality. Information that is discarded at this stage is forever lost; conversely, noise that is accepted will degrade the performance of the classifier that tends to be sensitive to complexity in the data. The signals that spoken language systems have to deal with are unique in the sense that they are generated by a biological system, for a biological

system. Human speech is adapted to, and uniquely constrained by the vocal and auditory systems; the result shows a distinct lack of engineering common sense. As a matter of fact, psychophysical studies over the last number of decades tend to leave us with the uncomfortable feeling that the world perceived through our senses is rather different from the one that we measure with our instruments. We will now consider some revealing aspects of human auditory perception and then examine the mel-scaled cepstrum algorithm in order to draw some conclusions.

3.3.1 Peculiarities of the human auditory system

A pure tone is uniquely defined by its intensity and frequency. The perceptual counterparts of these quantities are termed loudness and pitch respectively. Mostly we agree that pure tones can be ordered in such a way that one tone is "higher" or "lower" than another. Pitch is the criterion that we use to make such decisions. Like loudness, it is a complex, non-linear function of both frequency and intensity. Frequency does however remain the dominant factor in pitch perception; Stevens, Volkmann and Newman defined the mel scale, which relates pitch to frequency as depicted in Figure 3.1 [76]. It was later refined by Stevens and Volkmann in a classical paper [77]. The form of the curve was determined by perceptual experiments designed to find a linear relation among perceived pitches. A pitch of 2000 mels is therefore subjectively "twice as high" as a pitch of 1000 mels. (The unit "mel", incidentally, is derived from the word "melody".) The numeric range of the mel scale and its relation to sound intensity was fixed by defining a 40dB tone with a frequency of 1000Hz as having a pitch of 1000 mels. We fitted a curve on Stevens' and Volkmann's original data to obtain equations 3.1 and 3.2 where f denotes frequency in Hertz and ν, pitch in mels.

$$\nu(f) = \frac{4491.7}{(1 + \exp(7.1702 - 1.9824 \log_{10}(f))} - 30.360 \tag{3.1}$$

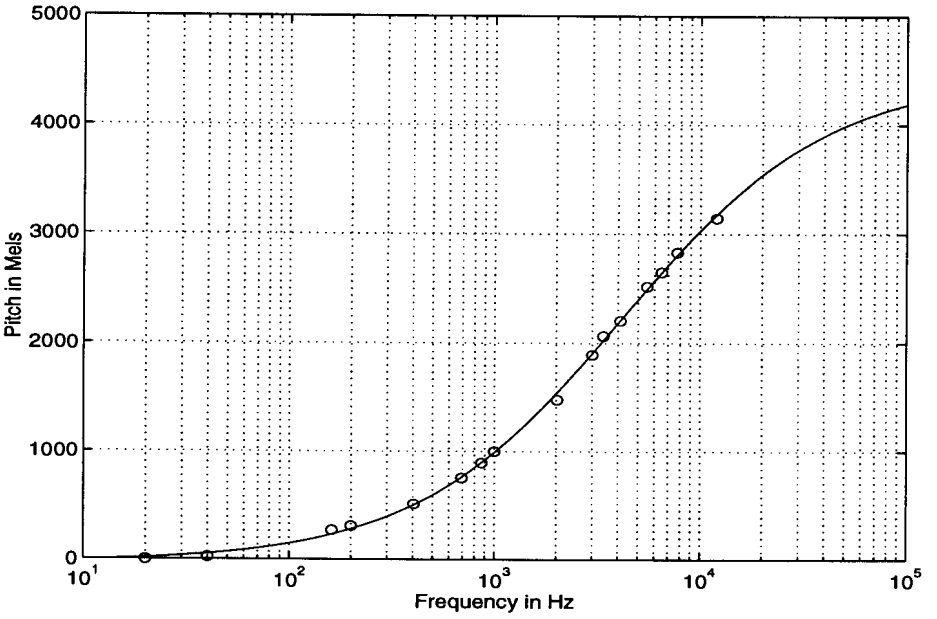

Figure 3.1: The mel scale

$$f(\nu) = 10^{\frac{7.1702 - \ln(\frac{4491.7}{\nu + 30.360} - 1)}{1.9824}} \tag{3.2}$$

Loudness is a psychological term used to describe the magnitude of an auditory sensation. Though mainly determined by the intensity of the perceived speech signal, it is also a function of frequency, as well as a number of psychological factors like fatigue, attention and alertness. Fletcher and Munson investigated and defined loudness [78]. They extended their work in [79], where they also addressed masking, an interesting auditory phenomenon: the threshold at which a tone can be perceived is raised when heard in the presence of another tone (or band of noise). In addition the effect of a sound on the auditory system persists for milliseconds. This latter effect is called forward masking. The perception of sound is therefore context sensitive in the frequency domain (masking) as well as the time domain (forward masking) on a very low level.

Studying masking, Fletcher found that a pure tone is masked essentially only by noise

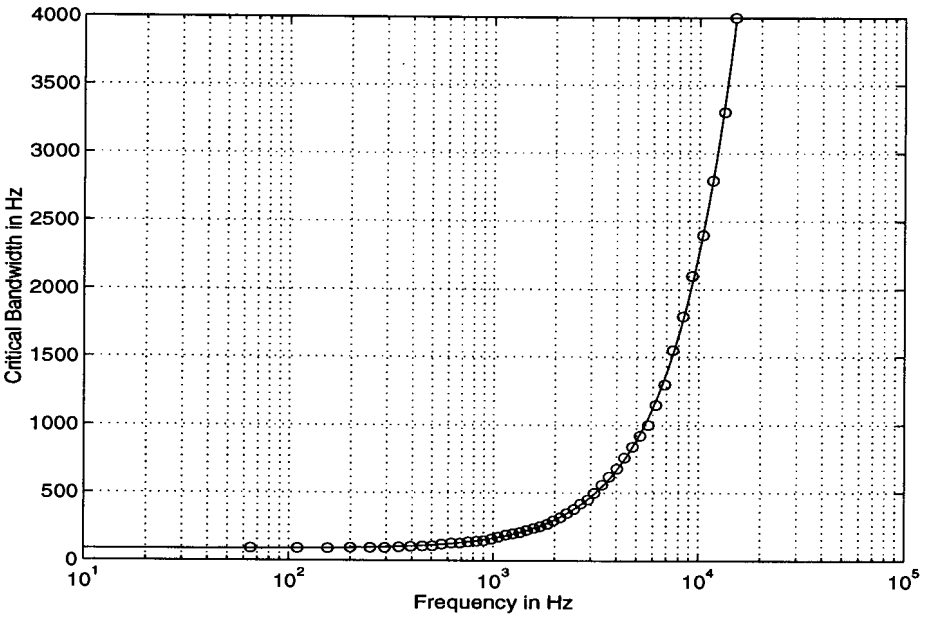

Figure 3.2: Critical bandwidth as a function of frequency

components within a certain narrow band centred at the frequency of the tone [80]. Differential pitch sensitivity, the smallest detectable change in frequency, is closely linked to these bands [81]. As a matter of fact, a number of perceptual phenomena seem to indicate that there exist what came to be called critical bands [82, 83, 84, 85]. The bandwidth of critical bands increase with frequency as shown in Figure 3.2. Equations 3.3 and 3.4 describe typical critical band cutoff frequencies.

$$\text{freq}_{\text{crit}_{\text{low}}}(f) = 1.3056 f^{0.95987} - 64.193 \tag{3.3}$$

$$\text{freq}_{\text{crit}_{\text{high}}}(f) = 0.70616 f^{1.0497} + 81.288 \tag{3.4}$$

The bark scale, another pitch scale that corresponds closely in form to the mel scale, is defined in terms of critical bands [86, 87]. The knowledge presented here can serve as a useful guide when developing a spoken language system.

3.3.2 Machine perception

Engineers were lured into the world of auditory perception mainly through attempts to optimise telephone systems. A classic study on the intelligibility of speech can be found in [88]. As the performance of digital computers exploded, it opened the world to speech recognition experiments. Following Bridle and Brown [89], Mermelstein [90] investigated the ability of the mel-scaled cepstrum, a non-linear, spectrally-base feature set, to distinguish between similar sounding consonants. In a later publication Davies and Mermelstein [91] found the mel-scaled cepstrum to be significantly superior to four other feature extraction front-ends in a syllable-oriented, speaker dependent, continuous speech recognition task. A recent study compared the mel-scaled cepstrum to two feature extraction front-ends based on auditory models in a speaker dependent word recognition task [92]. It was shown that the more complex front-ends provided little improvement in performance (a difference of 0.6 to 4 percentage points in error rate) to compensate for increased complexity and processing time ($\frac{1}{3}$ real time as opposed to respectively, 40 and 120 times real time). In addition it was shown that the mel-scaled cepstrum approach significantly outperforms a traditional LPC-based front-end. These results were extended to a (male) speaker independent, continuous task [93]. The mel-scale cepstrum does therefore seem to be a good choice.

We now present an algorithm to calculate the mel-scaled cepstrum. This algorithm has been reconstructed from [93, 94, 90, 92] and our own experience.

3.3.3 The mel-scaled cepstrum algorithm

Let the N-sample speech signal be

$$\mathbf{x} = x_0, \ldots, x_{N-1}. \tag{3.5}$$

3.3.3.1 Pre-emphasis

The speech signal is pre-emphasised to compensate for spectral tilt (i.e. $S'(w) = S(w).w^a$). This is a high-pass filtering operation and can be executed in either the time or frequency domain. The filter in the time-domain is of the form

$$x_i \leftarrow x_i - ax_{i-1}, \quad 0.9 \leq a \leq 1.0, \tag{3.6}$$

where the parameter a is not critical and is usually taken to be 0.95.

3.3.3.2 Normalisation

The maximum signal amplitude is normalised to one.

$$x_i \leftarrow \frac{x_i}{\max_{j=0,\ldots,N-1} |x_j|}, \quad i = 0,\ldots,N-1 \tag{3.7}$$

3.3.3.3 Blocking

The filtered, normalised signal is broken into M overlapping frames and stored in an $M \times W$ matrix $\mathbf{Y}$ with its rows $\mathbf{y}_i$ representing the frames. V is the step size and W the frame size. Frame size usually range from 10ms to 20ms and step size between 20 and 50 percent of frame size.

$$y_{ij} \leftarrow x_{Vi+j}, \quad j = 0,\ldots,W-1, \quad i = 0,\ldots,M-1 \tag{3.8}$$

3.3.3.4 Windowing

Each frame is multiplied with a window function to minimise signal discontinuities in the time domain and the resulting spectral artifacts.

$$y_{ij} \leftarrow y_{ij} w_j, \quad j = 0, \ldots, W-1, \quad i = 0, \ldots, M-1 \tag{3.9}$$

The Hamming window, described by equation 3.10, is a popular choice.

$$w_j = 0.54 - 0.46 \cos\left(\frac{2\pi j}{W-1}\right), \quad j = 0, \ldots, W-1 \tag{3.10}$$

3.3.3.5 Power spectrum

The power spectrum of each window is calculated and represented by the the $M \times U$ matrix $\mathbf{S}$. W (and $U = W/2$) will be constrained by the FFT algorithm in a practical implementation. We used a prime factor FFT which gives more freedom in the choice of W than standard radix-2 algorithms. Still, W needs to be one of a limited set of integers that will in general not be the same as the number determined by the choice of frame size. To work around this, $\mathbf{y}_i$ can be zero-padded or the frame size can be adjusted to coincide with a valid number.

$$\mathbf{s'}_i \leftarrow \mathrm{fft}(\mathbf{y}_i), \quad i = 0, \ldots, M-1 \tag{3.11}$$

$$s_{ij} = \left| s'_{ij} \right|^2, \quad j = 0, \ldots, U-1, \quad i = 0, \ldots, M-1 \tag{3.12}$$

3.3.3.6 Mel filter bank

The mel filter bank consists of overlapping triangular filters with the cutoff frequencies determined by the centre frequencies of the two adjacent filters. The filters have

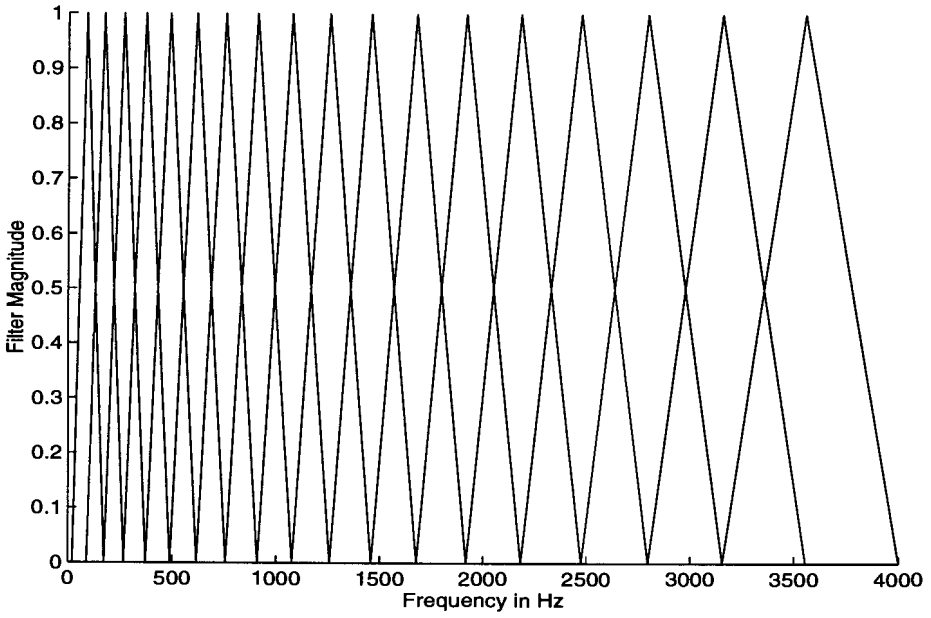

Figure 3.3: A mel scale filter bank

linearly spaced centre frequencies and fixed bandwidth on the mel scale, modelling the differential pitch sensitivity of the ear. This arrangement (depicted in Figure 3.3) results in a logarithmic spacing on the frequency scale with bandwidths roughly corresponding to the critical bandwidth curve. If one takes f_{min} as 20Hz (0 mels) and f_{max} as half the sampling rate, then the mel filter bank is defined by equations 3.13 to 3.15 and represented by the $K \times U$ matrix **F**. f_c is the centre frequency of a filter. The low and high cutoff frequencies, f_l and f_h, are the centre frequencies of the two adjacent filters. The number of filters, K, is usually between 13 and 24. Care must be taken that it is not too large, since crowding the filters will result in poor frequency resolution at low frequencies. Let

$$\gamma = \frac{W-1}{f_{\text{max}} - f_{\text{min}}}, \tag{3.13}$$

$$\begin{aligned} I_l &= \gamma(f_l - f_{\min}), \\ I_c &= \gamma(f_c - f_{\min}) \text{and} \\ I_r &= \gamma(f_h - f_{\min}), \end{aligned} \tag{3.14}$$

then

$$f_{ij} = \begin{cases} \frac{1}{f_c - f_l}(\frac{j}{\gamma} + f_{\min} - f_l) & \text{if } \lceil I_l \rceil \leq j \leq \lfloor I_c \rfloor \\ 1 + \frac{1}{f_c - f_h}(\frac{j}{\gamma} + f_{\min} - f_c) & \text{if } \lceil I_c \rceil \leq j \leq \lfloor I_r \rfloor \\ 0 & \text{otherwise.} \end{cases} \tag{3.15}$$

An approximation to the mel scale, frequently used in other implementations of the filter bank, is to have a number of linearly spaced filters with equal bandwidth under 1000Hz and then logarithmically spaced filters above 1000Hz where the centre frequency of each filter is 1.1 times the preceding centre frequency [92, 93].

3.3.3.7 Log energy filter coefficients

Now, one of the confusing aspects of the mel-scaled cepstrum is that the mel filter bank is not really a filter. In stead of just weighting each point of the spectrum with the filter weights, one calculates a type of inner product and find what is probably best described as an energy coefficient for each filter. To compensate for the increasing bandwidths of the filters, the energies are normalised as per equation 3.17. This part of the processing is completed by taking the logarithm of each energy coefficient. It is a crude attempt to model the non-linear intensity-loudness relationship which is logarithmic in nature. These operations, implemented by equation 3.16, result in the $M \times K$ matrix **P**.

$$p_{ij} = \log_{10}\left(\frac{1}{A_j}\sum_{k=0}^{U-1} s_{ik} f_{jk}\right), \quad \begin{matrix} j = 0, \ldots, K-1 \\ i = 0, \ldots, M- \end{matrix} \tag{3.16}$$

where

$$A_j = \sum_{k=0}^{U-1} f_{jk}. \tag{3.17}$$

3.3.3.8 Inverse discrete cosine transform

The inverse cosine transform is used to orthogonalise the filter energy vectors. It has been suggested to be an efficient approximation to the optimal Karhunen-Loeve transform [95, 96, 97]. Because of this orthogonalisation step, the information of the filter energy vector is compacted into the first number of components and we can shorten the vector to L components, resulting in the $M \times L$ matrix $\mathbf{Q}$.

$$q_{ij} = \frac{1}{K} \sum_{k=0}^{K-1} p_{ik} \cos\left((k - 0.5)\frac{\pi j}{L}\right), \quad \begin{matrix} j = 0, \ldots, L-1 \\ i = 0, \ldots, M-1 \end{matrix} \tag{3.18}$$

L is chosen to be less than K, usually somewhere between 9 and 15.

3.3.3.9 Incorporation of dynamic features

It has been found that including the first and second derivatives of the log energy vector significantly improves the performance of mel-scaled cepstrum-based systems [93, 92]. These are referred to as the delta and delta-delta cepstra. Since we are dealing with discrete data, it is advisable to calculate the derivatives on a smoothed approximation. This issue is considered in detail in [98]. The first mel-scaled cepstral coefficient represents the mean energy in each frame and is usually dropped. The delta-cepstrum and occasionally the delta-delta cepstrum is concatenated to the mel-scaled cepstrum to form one long vector. This then constitutes a mel-scaled cepstrum feature vector.

3.3.4 Discussion

From an engineering point of view, the mel-scaled cepstrum is an efficient algorithm because it is performed mainly in the frequency domain and can therefore use the FFT. The inverse cosine transform is an efficient dimension reduction technique. From a perceptual point of view, the mel-scaled cepstrum takes into account the non-linear nature of pitch perception (the mel scale) as well as loudness perception (the log operation). It also models critical bandwidth as far as differential pitch sensitivity is concerned (the mel scale). The derivatives serve to incorporate dynamic information. However, the mel-scaled cepstrum does not model (static) masking. It does not model forward (dynamic) masking and there is no feedback between higher level processing (the classifier stage) and feature extraction.

The feature extraction process models human audition up to the point where we become aware of speech. The next section describes one way of casting the infinite range of speech sounds into a manageable set of recognisable symbols. At this point we are probably starting to diverge very far from anything going on in the human brain.

3.4 Vector quantisation

Vector quantisation (VQ) is the process of breaking up (infinite, continuous) vector space into a finite set of chunks or quanta. We can then assign a symbol to each quantum and represent a trajectory through vector space with a sequence of symbols.

In our system vector quantisation is performed with the aid of the Self-Creating and Organising Neural Network (SCONN) algorithm [99]. It is essentially a top-down clustering algorithm. The top-down, as opposed to bottom-up, approach makes it remarkably efficient.

3.4.1 The SCONN algorithm

SCONN finds clusters in feature space and the centroids of these clusters constitute the VQ codebook. The codebook, in turn, is used to translate a set of vectors in feature space into a set of symbols. The algorithm below is adapted from Choi and Park's paper [99]. It has been shown to outperform two well-known algorithms - Kohonen's Self Organising Feature Map (SOFM) [100], and the Linde-Buzo-Gray (LBG) algorithm [101].

The algorithm starts off with a single node in the vector space of the dataset that it attempts to model. This "mother" node has an activation region that stretches across the whole input space. **Activation value** represents the range of the activation region of a node. A node is activated by an input vector when the Euclidean distance to the vector is within this range. The activation region is shrunk with each iteration and new nodes (with activation regions equal in range to the current range of other nodes) are added as they become needed to model data points in vector space that fall outside the shrinking regions of other nodes. In addition, nodes slowly migrate in the direction of high concentrations of data points that fall within their activation regions. **Weights** ($\mathbf{w}_j$) are simply the vector representations of nodes. The result of the learning algorithm is therefore a codebook of vectors (and ranges) that model the input data. The input data is vector quantised by assigning the closest vector from the codebook to each data point. In this way the input data that consists of a series of vectors is reduced to a series of symbols or tokens.

Let $\mathbf{Q}$ be the $M \times L$ feature matrix, containing M feature vectors of size L, i.e. M points in L-dimensional feature space. Use i to iterate over the feature vectors ($\mathbf{q}_i$) and j to iterate over the nodes ($\mathbf{w}_j$). Let t denote the absolute iteration count.

1. Initialise weights. Let $i = 0$ and $t = 0$. Start with one ($N = 1$) node, $\mathbf{w}_0$, and give it a random weight in the neighbourhood of the Euclidean centre of the cluster

off all the feature vectors in **Q**. Set activation level $\rho(0)$ big enough to cover all data points.

2. Present new input (feature) vector $\mathbf{q}_i$.

3. Calculate distance to all nodes.

$$d_j^2 = \sum_{k=0}^{L-1} (q_{ik} - w_{jk}(t))^2, \quad j = 0, 1, \ldots, N-1 \tag{3.19}$$

4. Select winner node.

$$j_{win} = \arg \min_{j=0,1,\ldots,N-1} \left[d_j^2 \right] \tag{3.20}$$

5. Decide whether winner node is active. If winner node is active, i.e. $d_{j_{win}} < \rho(t)$, then go to step 6, else go to step 7.

6. Adapt weights of active winner node.

$$w_{j_{win}k}(t+1) = w_{j_{win}k}(t) + \alpha(q_{ik} - w_{j_{win}k}(t)), \quad k = 0, 1, \ldots, L-1, \tag{3.21}$$

where α is a factor that regulates the tempo at which nodes migrate in feature space. Go to step 8.

7. Create a son node from the inactive winner (mother) node. Increase N by 1 and assign the following weight to the son node:

$$w_{(N-1)k}(t+1) = w_{j_{win}k}(t) + \beta(q_{ik} - w_{j_{win}k}(t)), \quad k = 0, 1, \ldots, L-1, \tag{3.22}$$

where β is a resemblance factor between 0 and 1.

8. Decrease activation values of all nodes.

$$\rho(t) = a\exp(-ct) + b, \tag{3.23}$$

where a, b and c are problem specific constants determined empirically. Halt if stop criterion is met, else increase t by 1, increase i by 1 and go to step 2.

Any one of three stop criteria can be used to halt the algorithm:

1. Iteration count $t >$ maximum number of iterations.
2. Number of nodes, or vector quanta, $N >$ maximum number of nodes.
3. Activation level or range $\rho(t) <$ minimum range.

We use the size of the vector quantisation code book, i.e. the maximum number of nodes allowed, as a stop criterion. Once a codebook is generated in this manner, the series of vectors $\mathbf{q}_i$ can be coded as a series of M tokens, τ_i, that each represent the node closest to the particular feature vector in feature space.

3.4.2 Discussion

We have presented a vector quantisation algorithm that allows us to translate a feature stream into a token stream. Features are extracted from relatively short (~ 10 to 30ms), fixed-length speech frames; tokens consequently represent sub-phonetic units. In the unsupervised approach we attempt to directly utilise the statistical properties of streams of these tokens to identify a language. Sophisticated approaches, however, exploit the higher information content of phones - larger, variable-length speech segments. In the next section we take a look at this approach.

3.5 Phone recognition and hidden Markov models

Phone recognition, or token alignment, is the process of classifying the feature stream into a time-aligned sequence of tokens. Tokens, in this context, represent phones. This process, being continuous speech recognition in essence, is the problem at the heart of spoken language processing. Fortunately, ALI is less demanding of this process than most other spoken language applications. Since only the statistics of the tokens are of importance, rather than the exact transcription, it is robust with regard to the recognition process. On the other hand, every mistake contributes towards inaccuracy in the statistics which will affect system performance.

3.5.1 Phonetic segmentation followed by classification

One way to approach the problem is to pre-segment the feature stream into chunks corresponding to phones, and then proceed to classify those chunks using a classifier like hidden Markov models. Typically the segmentation is done by finding some form of derivative of the feature steam over time and defining phone boundaries at local peaks in the derivative. Unfortunately, it seems to be quite difficult to mimic the auditory system's ability to break a continuous sound stream into chunks without feedback from higher levels of modelling.

3.5.2 Integrated phonetic segmentation and classification

The more sophisticated approach, which has by now become standard, is to adjust the hypothesised phone boundaries with a search algorithm. The phone sequence is reclassified and the probability of the sequence re-estimated. The phone boundaries are adjusted to maximise this probability.

We have briefly sketched two approaches to phone recognition. Both the methods

(can, and mostly do) use hidden Markov models (HMMs) to model individual phones in terms of sub-phonetic feature sequences. We will now proceed to introduce the theory of hidden Markov models. The implementation of HMMs, various architectural issues and specifically the application to spoken language processing opens up a large number of issues that are topics of research on their own. We will not enter into these issues here. The most salient point maybe, is continuous versus discrete HMMs. We experimented with both, but since it does not provide significant additional insight, we will refrain from developing the theory of continuous HMMs. For more information please see [102, 103, 104, 105, 106, 28, 107]. The following is adapted from [108].

3.5.3 Definition of a hidden Markov model

The Hidden Markov model is an extension of the idea of discrete-time Markov processes. Like a finite state machine, an HMM can be in any of a number of discrete states at a given time. Let the states for an N-state HMM be taken from the set $Q = \{1, 2, ..., N\}$. Let q_t denote the active state at time t.

The HMM changes state at each time step. The next state is a probabilistic function of the current state and the $N \times N$ state transition matrix **A**,

$$a_{ij} = P(q_{t+1} = j | q_t = i), \quad \begin{matrix} i = 1, 2, \ldots, N \\ j = 1, 2, \ldots, N. \end{matrix} \tag{3.24}$$

That is, a_{ij} is the probability of moving form state i to state j at any given time.

In addition to **A**, the vector π defines the initial state distribution that determines in which state the HMM starts up. Accordingly

$$\pi_i = P(q_1 = i), \quad i = 1, 2, \ldots, N. \tag{3.25}$$

The state of the system is not directly observable, in stead, at time t the active state emits an observation, o_t, that is one of M possible symbols taken from the set $V = \{v_1, v_2, \ldots, v_M\}$. The emitted symbol is a probabilistic function of the current state and the observation symbol probability distribution matrix **B**, where

$$b_j(k) = P(o_t = v_k | q_t = j), \quad \begin{matrix} j = 1, 2, \ldots, N \\ k = 1, 2, \ldots, M. \end{matrix} \tag{3.26}$$

That is, $b_j(k)$ is the probability of emitting symbol k in state j.

Given N, M, and V, an HMM is uniquely defined by the three sets of probabilities **A**, **B** and π. The parameter set, λ, for a model is denoted as

$$\lambda = (\mathbf{A}, \mathbf{B}, \pi). \tag{3.27}$$

In addition, λ defines $P(\mathbf{o}|\lambda)$, a probability measure for **o**, where

$$\mathbf{o} = o_1, o_2, \ldots, o_T, \tag{3.28}$$

is the observation sequence generated over T time steps by the HMM λ.

3.5.4 Basic hidden Markov model problems

Having defined the HMM in the previous section, we now turn to the three major problems that come to mind when implementing a system that utilises HMMs.

Problem 1

Given an observation sequence $\mathbf{o} = o_1, o_2, \ldots, o_T$ and a model $\lambda = (\mathbf{A}, \mathbf{B}, \pi)$, find $P(\mathbf{o}|\lambda)$ i.e. the probability of the observation sequence given the model. This information is needed in order to classify an observation sequence.

Problem 2

Find the most likely state sequence $\mathbf{q} = q_1, q_2, \ldots, q_T$, given the model λ and an observation sequence $\mathbf{o} = o_1, o_2, \ldots, o_T$. In a speech application this typically allows segmentation of an utterance.

Problem 3

Adjust $\lambda = (\mathbf{A}, \mathbf{B}, \pi)$ to maximise $P(\mathbf{o}|\lambda)$. This represents the training of an HMM.

3.5.5 Solutions to the basic hidden Markov model problems

3.5.5.1 Solution 1 – Probability evaluation

A brute force approach is to enumerate every possible state sequence of length T. Consider one of N^T such possible fixed state sequences

$$\mathbf{q} = q_1, q_2, \ldots, q_T, \tag{3.29}$$

where q_1 is the initial state. The probability of the observation sequence $\mathbf{o}$ given the state sequence in 3.29 is

$$P(\mathbf{o}|\mathbf{q}, \lambda) = \prod_{t=1}^{T} P(o_t|q_t, \lambda), \tag{3.30}$$

if the observations are taken to be statistically independent. From 3.26

$$P(\mathbf{o}|\mathbf{q}, \lambda) = b_{q_1}(o_1) \cdot b_{q_2}(o_2) \cdot \ldots \cdot b_{q_T}(o_T). \tag{3.31}$$

The probability of the state sequence $\mathbf{q}$ can be written as

$$P(\mathbf{q}|\lambda) = \pi_{q_1} \cdot a_{q_1 q_2} \cdot a_{q_2 q_3} \cdot \ldots \cdot a_{q_{T-1} q_T}. \tag{3.32}$$

The product of 3.31 and 3.32 yields the joint probability of **o** and **q**:

$$P(\mathbf{o},\mathbf{q}|\lambda) = P(\mathbf{o}|\mathbf{q},\lambda)P(\mathbf{q}|\lambda). \tag{3.33}$$

Given the model λ, the probability of the observation sequence **o** is obtained by summing the joint probability given in 3.33 over all possible state sequences **q**,

$$\begin{aligned} P(\mathbf{o}|\lambda) &= \sum_{\text{allq}} P(\mathbf{o}|\mathbf{q},\lambda)P(\mathbf{q}|\lambda) \\ &= \sum_{\text{allq}} \pi_{q_1} \cdot b_{q_1}(o_1) \cdot a_{q_1 q_2} \cdot b_{q_2}(o_2) \ldots a_{q_{T-1} q_T} \cdot b_{q_T}(o_T). \end{aligned} \tag{3.34}$$

The above is not computationally feasible and we must resort to a more efficient approach. One such a procedure is the *forward backward algorithm.*

Forward backward algorithm

Forward procedure

Define the forward variable $\alpha_t(i)$ as:

$$\alpha_t(i) = P(o_1, o_2, \ldots, o_t, q_t = i|\lambda). \tag{3.35}$$

The forward variables express the probability of the partial observation sequence, $o_1, o_2, \ldots, o_t$ and state i at time t, given the model λ. Their values are computed by iteratively solving for $\alpha_t(i)$. The procedure is as follows:

1. **Initialisation**

$$\alpha_1(i) = \pi_i \cdot b_i(o_1), \quad i = 1, 2, \ldots, N \tag{3.36}$$

2. **Induction**

$$\alpha_{t+1}(j) = \left[\sum_{i=1}^{N} \alpha_t(i) \cdot a_{ij}\right] \cdot b_j(o_{t+1}), \quad \begin{matrix} j = 1,2,\ldots,N \\ t = 1,2,\ldots,T-1 \end{matrix} \tag{3.37}$$

3. **Termination**

$$P(\mathbf{o}|\lambda) = \sum_{i=1}^{N} \alpha_T(i) \tag{3.38}$$

This provides us with $P(\mathbf{o}|\lambda)$ in a much more efficient manner. The backward procedure is used in conjunction with the forward procedure to solve further problems.

Backward procedure

Define the backward variable $\beta_t(i)$ as follows:

$$\beta_t(i) = P(o_{t+1}, o_{t+2}, \ldots, o_T | q_t = i, \lambda). \tag{3.39}$$

The backward variables express the probability of the partial observation sequence from $t+1$ to the end of the sequence, given the state i at time t and the model λ. These probabilities are calculated by solving for $\beta_t(i)$ iteratively, according to the following strategy:

1. **Initialisation**

$$\beta_T(i) = 1, \quad i = 1,2,\ldots,N \tag{3.40}$$

2. **Induction**

$$\beta_t(i) = \sum_{j=1}^{N} a_{ij} \cdot b_j(o_{t+1}) \cdot \beta_{t+1}(j), \quad \begin{matrix} i = 1, 2, \ldots, N \\ t = T-1, T-2, \ldots, 1 \end{matrix} \tag{3.41}$$

3.5.5.2 Solution 2 – Optimal state sequence

Because there are various different ways to define an 'optimal' state sequence associated with any given observation sequence, no unique solution exists for Problem 2. One possible optimality criterion is to choose the states q_t that are individually most likely at time t. This criterion maximises the expected number of correct individual states. To implement this solution, an *a posteriori* probability variable is defined as follows:

$$\gamma_t(i) = P(q_t = i|\mathbf{o}, \lambda), \tag{3.42}$$

where $\gamma_t(i)$ indicates the probability of being in state i at time t, given the observation sequence $\mathbf{o}$ and the model λ. The value of $\gamma_t(i)$ can be expressed as:

$$\begin{aligned} \gamma_t(i) &= P(q_t = i|\mathbf{o}, \lambda) \\ &= \frac{P(\mathbf{o}, q_t = i|\lambda)}{P(\mathbf{o}|\lambda)} \\ &= \frac{P(\mathbf{o}, q_t = i|\lambda)}{\sum_{i=1}^{N} P(\mathbf{o}, q_t = i|\lambda)}. \end{aligned} \tag{3.43}$$

However, $P(\mathbf{o}, q_t = i|\lambda)$ is equal to $\alpha_t(i)\beta_t(i)$ and therefore $\gamma_t(i)$ can be expressed in terms of the forward and backward variables as:

$$\gamma_t(i) = \frac{\alpha_t(i) \cdot \beta_t(i)}{\sum_{i=1}^{N} \alpha_t(i) \cdot \beta_t(i)}, \tag{3.44}$$

where $\alpha_t(i)$ represents the partial observation sequence $o_1, o_2, \ldots, o_t$ at time t while $\beta_t(i)$ accounts for the rest of the observation sequence i.e. $o_{t+1}, o_{t+2}, \ldots, o_T$, given

state $q_t = i$ at t.

The individually most likely state q_t^* at time t can then be solved for as follows:

$$q_t^* = \arg \min_{1 \leq i \leq N} [\gamma_t(i)], \quad t = 1, 2, \ldots, T. \tag{3.45}$$

The Viterbi algorithm is a formal technique based on dynamic programming methods and it is often used to find the single best state sequence.

The Viterbi algorithm

In order to find the best state sequence, $\mathbf{q} = q_1, q_2, \ldots, q_T$, for the given observation sequence $\mathbf{o} = o_1, o_2, \ldots, o_T$, define the quantity

$$\delta_t(i) = \max_{\text{all}\mathbf{q}} P(q_1, q_2, \ldots, q_{t-1}, q_t = i, o_1, o_2, \ldots, o_t | \lambda), \tag{3.46}$$

where $\delta_t(i)$ is the highest probability along a single path, at time t, that accounts for the first t observations and ends in state i. By induction the value of $\delta_{t+1}(j)$ is

$$\delta_{t+1}(j) = \max_i [\delta_t(i) \cdot a_{ij}] \cdot b_j(o_{t+1}). \tag{3.47}$$

1. **Initialisation**

$$\begin{aligned} \delta_1(i) &= \pi_i \cdot b_i(o_1), \\ \psi_1(i) &= 0, \end{aligned} \qquad i = 1, 2, \ldots, N \tag{3.48}$$

2. **Recursion**

$$\delta_t(j) = \max_{1 \leq i \leq N} [\delta_{t-1}(i) \cdot a_{ij}] \cdot b_j(o_t), \qquad \begin{aligned} t &= 2, 3, \ldots, T \\ j &= 1, 2, \ldots, N \end{aligned} \tag{3.49}$$

$$\psi_t(j) = \arg\max_{1\leq i\leq N}[\delta_{t-1}(i)\cdot a_{ij}], \quad \begin{matrix} t = 2,3,\ldots,T \\ j = 1,2,\ldots,N \end{matrix} \tag{3.50}$$

3. **Termination**

$$\begin{aligned} P^* &= \max_{1\leq i\leq N}[\delta_T(i)] \\ q_T^* &= \arg\max_{1\leq i\leq N}[\delta_T(i)] \end{aligned} \tag{3.51}$$

4. **Reconstruction**

$$q_t^* = \psi_{t+1}(q_{t+1}^*), \quad t = T-1, T-2, \ldots, 1 \tag{3.52}$$

The resulting trajectory, $q_1^*, q_2^*, \ldots, q_T^*$, solves Problem 2.

3.5.5.3 Solution 3 – Parameter estimation

The last, and most difficult problem, is the estimation of a set of model parameters that satisfies a certain optimisation criterion. One solution is the Baum-Welch or expectation maximisation (EM) method.

Baum-Welch algorithm

Define

$$\xi_t(i,j) = P(q_t = i, q_{t+1} = j|\mathbf{o},\lambda). \tag{3.53}$$

Compute these probabilities using the forward backward variables:

$$\xi_t(i,j) = \frac{P(q_t = i, q_{t+1} = j, \mathbf{o}|\lambda)}{P(\mathbf{o}|\lambda)}, \tag{3.54}$$

$$\xi_t(i,j) = \frac{\alpha_t(i) \cdot a_{ij} b_j(o_{t+1}) \cdot \beta_{t+1}(j)}{P(\mathbf{o}|\lambda)}, \tag{3.55}$$

$$\xi_t(i,j) = \frac{\alpha_t(i) \cdot a_{ij} \cdot b_j(o_{t+1}) \cdot \beta_{t+1}(j)}{\sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_t(i) \cdot a_{ij} \cdot b_j(o_{t+1}) \cdot \beta_{t+1}(j)}. \tag{3.56}$$

The probability of being in state i at t given the observation sequence $\mathbf{o}$ and the model, λ was defined in equation 3.42 as

$$\gamma_t(i) = P(q_t = i|\mathbf{o}, \lambda). \tag{3.57}$$

$\gamma_t(i)$ and $\xi_t(i,j)$ can be related by summing over j:

$$\gamma_t(i) = \sum_{j=1}^{N} \xi_t(i,j). \tag{3.58}$$

Note that

$$\sum_{t=1}^{T-1} \gamma_t(i) = \text{expected number of transitions from state } i \text{ in } \mathbf{o}, \tag{3.59}$$

and

$$\sum_{t=1}^{T-1} \xi_t(i,j) = \text{expected number of transitions from state } i \text{ to state } j \text{ in } \mathbf{o}. \tag{3.60}$$

Choose the initial parameters, $\lambda = (\mathbf{A}, \mathbf{B}, \pi)$, arbitrarily.

Reestimate the parameters:

$$\bar{\pi}_j = \gamma_1(i), \tag{3.61}$$

$$\bar{a}_{ij} = \frac{\sum_{t=1}^{T-1} \xi_t(i,j)}{\sum_{t=1}^{T-1} \gamma_t(i)}, \tag{3.62}$$

$$\bar{b}_j(k) = \frac{\sum_{t=1, o_t=v_k}^{T} \gamma_t(j)}{\sum_{t=1}^{T} \gamma_t(j)}. \tag{3.63}$$

Set $\bar{\lambda} = (\bar{\mathbf{A}}, \bar{\mathbf{B}}, \bar{\pi})$.

If $\lambda = \bar{\lambda}$ then quit (having achieved convergence), else set λ to be $\bar{\lambda}$ and repeat procedure.

3.5.6 Discussion

We have presented an introduction to hidden Markov model theory in terms of the three basic problems that one encounters during application of HMMs. These techniques provide us with a powerful way to model dynamic stochastic systems and have with much success been applied to the problem of extracting speech sounds from a speech stream. The last remaining ALI component is a framework that will allow us to model the characteristic probability constraints imposed on token sequences by the source language. The next section considers a solution to this problem.

3.6 Language modelling

With language modelling we aim to capture and recognise the language-specific constraints inherent in a sequence of symbols generated by a certain language. We present N-gram modelling in the next section and subsequently introduce an adapted approach that focuses on discrimination *between*, as opposed to representation *of* different languages.

3.6.1 N-gram modelling

Consider a sequence of symbols $\mathbf{q} = q_1, q_2, \ldots, q_T$ (representing an utterance) generated by language λ_i, where q_t is drawn from the set of tokens $\Psi_i = \{\psi_1, \psi_2, \ldots, \psi_M\}$, associated with λ_i. The exact nature of these tokens are not of immediate importance; it might be phones, phonemes, sub-phonetic units or even words. On all of these levels, the probabilities associated with a sequence of observed tokens are determined by the source language. A sequence of two tokens is known as a **bigram,** three tokens, a **trigram** and in general a sequence of N tokens is called an **N-gram.**

In trying to assign a specific language λ_i to an utterance represented by the symbol sequence $\mathbf{q}$, we are interested in

$$P(\mathbf{q}|\lambda_i) = P(q_1|\lambda_i) \cdot P(q_2|q_1, \lambda_i) \cdot \ldots \cdot P(q_T|q_1 q_2 \ldots q_{T-1}, \lambda_i) \tag{3.64}$$

$$= \prod_{t=1}^{T} P(q_t|q_1 q_2 \ldots q_{t-1}, \lambda_i). \tag{3.65}$$

For a sequence longer than two or three symbols, the estimation of this probability quickly becomes a practical impossibility and we have to resort to an approximation.

The N-gram approximation states that

$$P(q_t|q_1q_2\dots q_{t-1},\lambda_i) \approx P(q_t|q_{t-N+1}q_{t-N+2}\dots q_{t-1},\lambda_i), \tag{3.66}$$

and hence

$$P(\mathbf{q}|\lambda_i) = \prod_{t=1}^{T} P(q_t|q_{t-N+1}q_{t-N+2}\dots q_{t-1},\lambda_i). \tag{3.67}$$

Even this estimation becomes infeasible for N more than two or three. Fortunately, the statistics associated with a sequence of only two phones are already quite distinctive. Still, there are difficulties associated with these estimations and we suggest a slightly different approach in the next section.

3.6.2 Discriminatory vs. representational modelling

In the previous section we presented a simple probabilistic framework for N-gram modelling of a language. This was a *representational* approach, since we attempted to estimate accurate probabilities that describe the way in which a sequence of tokens generated by a language is constrained by that language. However, for a token set Ψ_i of size M, there are M^N possible N-grams. This number can easily grow very large and one is consequently faced with the practical problems of estimating extremely small probabilities, requiring huge amounts of data. Since we are trying to discriminate among languages, an alternative approach is to focus on the *discriminatory* properties of a much smaller set of N-grams. In this way one can capture the information relevant to ALI in a more efficient manner.

Consider the N-gram $q_{t-N+1}\dots q_{t-1}q_t$. Let $C(\cdot|\lambda_i)$ denote the number of times that an N-gram occurs in $H_i = \{\mathbf{q}_{i1}, \mathbf{q}_{i2}, \dots, \mathbf{q}_{iK_i}\}$, the set of training sequences that represent λ_i, where K_i is the number of training sequences for that language. In addition, let $C(\cdot)$

denote the number of times that an N-gram occurs in the union of all such training sets, $H_\Lambda = \bigcup_{i=1}^{S} H_i$, where S is the number of languages. Now define $D(\cdot)$, an N-gram *distinctiveness measure*, as follows:

$$D(q_{t-N+1} \dots q_{t-1} q_t) = \max_i \left[\frac{C(q_{t-N+1} \dots q_{t-1} q_t | \lambda_i)}{C(q_{t-N+1} \dots q_{t-1} q_t)} \right] \tag{3.68}$$

For each language the model-building algorithm constructs a histogram of the N-grams encountered in the training set H_Λ. The histograms are sorted according to distinctiveness as defined in 3.68, using absolute frequency of occurrence as a secondary sort key. The absolute frequency is used to weed out N-grams that might have high information content, but only occur rarely. The L_i most distinctive N-grams are kept for each language. L_i is chosen in such a way as to balance the total number of expected occurrences of the N-grams across all languages, i.e.

$$\sum_{j=1}^{L_i} C(\mathbf{r}_{ij} | \lambda_i) \approx \min_k \sum_{j=1}^{L_k} C(\mathbf{r}_{kj} | \lambda_k), \quad i = 1, 2, \dots, S, \tag{3.69}$$

where $\mathbf{r}_{ij}$ is the jth N-gram in language model i.

A list of L_i N-grams generated in this manner is considered a language model. When classifying an unknown utterance, a point is awarded to a competing language hypothesis for every occurrence of an N-gram present in its model list. The hypothesis with the highest score wins.

This approach has the disadvantage that a model is not uniquely defined for a language, but in return it provides a model that is optimised for discrimination among competing language models.

3.6.3 Discussion

We presented representational and discriminatory N-gram models. The algorithm used to estimate the latter is adaptive in the sense that the most distinctive N-gram model for a language depends on the other languages that are involved in the training process. It provides a way to deal efficiently with that information which is important for classification purposes.

Having described various ALI components, we present the main architectures based on these methods in the next section.

3.7 ALI architectures

From a number of reviews of ALI over the last couple of years a significant trend concerning performance and complexity emerges [4, 36, 9, 7, 10]. The first attempts followed a simple pattern recognition approach with little or no *a priori* information. As large amounts of labelled speech data became available, systems increased in complexity while trying to capture more and more *a priori* information in order to boost performance. While state-of-the-art systems currently operate on about ten languages, future systems will eventually have to handle hundreds of languages in an increasingly global community. From such a point of view, the explosion in complexity of current architectures becomes a potential problem. The following is an overview of ALI architectures that use the components described in this chapter in various ways. The architectures are presented in order of increasing complexity and performance. We implemented those described in Sections 3.7.3 (vector quantisation followed by language modelling) and 3.7.5 (parallel phone recognition followed by language modelling).

3.7.1 Raw speech

At the bottom end of the complexity scale are attempts to directly classify a raw speech signal. This approach assumes that language-specific raw speech wave forms differ in a trivial way on a very low level and/or that a sufficiently complex classifier, like a neural network, can "discover" and adequately model higher level building blocks through self-organisation. Though the former assumption does not seem to hold very well, the latter has much merit. Unfortunately it seems that the problem is simply too complex to allow this avenue of attack with current understanding and technology. An example of this approach is the work of Kwasny *et al.* [23, 24, 25] who used raw speech data with neural network and recurrent neural network classifiers. Unfortunately, their data sets were much too small for the experiments to deliver any meaningful results.

3.7.2 Gaussian mixture model classification

A bit more sophisticated is the Gaussian mixture model (GMM) approach [27, 22, 7]. GMM ALI is motivated by the assumption that languages differ on the acoustic level (say, in spectral content) and that these differences can be captured directly by appropriate features. The dynamic nature of the speech signal, contextual information and higher order organisation are neglected. The speech signal is broken into overlapping frames by moving a fixed-length analysis window over the signal in fixed increments. The frames are represented as points in feature space by extracting acoustic features for each frame. The assumption is that each language can be represented by a set of N multivariate Gaussian distributions in feature space. During the training phase of such a system, vector quantisation is used to find the initial clusters, which are refined through an estimation-maximisation (EM) procedure. A test utterance is classified by finding that model which has produced it with maximum likelihood. The

latest study of this type of system was done by Zissman [7], who achieved 50% and 53% for 5s and 45s utterances respectively on a 10-language task using the OGI_TS corpus. These results are significantly worse than those of more complex systems reported in Chapter 2.

3.7.3 Vector quantisation followed by language modelling

The next level of complexity in ALI architectures arises from acknowledging the symbolical (implying discrete units), contextual and hierarchical nature of speech, i.e. the fact that information resides in the way that higher level units are arranged with regard to each other, rather than simply the presence or absence of certain sounds. If we are to work with units of some kind, we require a discretisation process. As in the previous section, the speech stream is broken into frames which are represented as points in feature space. Feature space is then discretised by means of vector quantisation. This allows us to translate the continuous speech stream into (sub-phonetic) units, by assigning an entry from the VQ codebook to each frame. Having represented an utterance as a sequence of symbols, one can use N-gram analysis to create a language model that characterises a language in terms of statistics describing the probabilities of various combinations of units occurring in close proximity to each other.

This approach still assumes that unlabelled, fixed-length, sub-phonetic units are good enough and attempts to model speech at a level below that which humans are consciously aware of. In defence, one must say that it is about as far as one can go before requiring human expertise in the form of hand-labelled speech. It therefore maintains a very important advantage in being much cheaper than more complex systems that require hand-labelled speech. In addition, it scales better than more complex systems. Sugiyama [3, 8] has examined an approach loosely belonging to this class. He achieved 80% on a 20-language task (64s utterances) as described in Section 2.3. One of our systems also falls in this class; the experiments and results for this system are detailed

in Section 4.6.

The next architecture moves over the threshold that distinguishes supervised from unsupervised systems.

3.7.4 Phone recognition followed by language modelling

Zissman describes a system embodying the next level of complexity as "Monolingual Phone Recognition followed by language dependent N-gram Language Modelling" (PRLM) [7]. The system uses hand-labelled speech data from one language to train a single monolingual phone recogniser. The phone recogniser transcribes a test utterance into a phone string, which is processed using a number of N-gram language models – one for each language hypothesis. The language models produce probabilities or system-specific scores which are classified with a neural network, linear classifier, voting scheme or some equivalent technology. A variation on this system uses a single multilingual front-end, trained on the data from a number of different languages [48, 49]. All major research efforts have explored this class of systems [41, 42, 43, 37, 38, 39, 40, 7]. Zissman [7], achieved 54% and 72% with a PRLM system for 5s and 45s utterances respectively, on a 10-language task using the OGI_TS corpus.

3.7.5 Parallel phone recognition followed by language modelling

A variation on the previous architecture, "Parallel Monolingual Phone Recognition followed by language dependent N-gram Language Modelling" (PPRLM), defines state-of-the-art ALI. It again goes one step further by introducing multiple language-specific phone recognisers operating in parallel. For an N language task, the system shown in Figure 3.4 consists of M language dependent phone recognisers, followed by an array of $N \times M$ language models. The language models each produce a likelihood score. The likelihood scores are averaged over M models for each language and are classified as

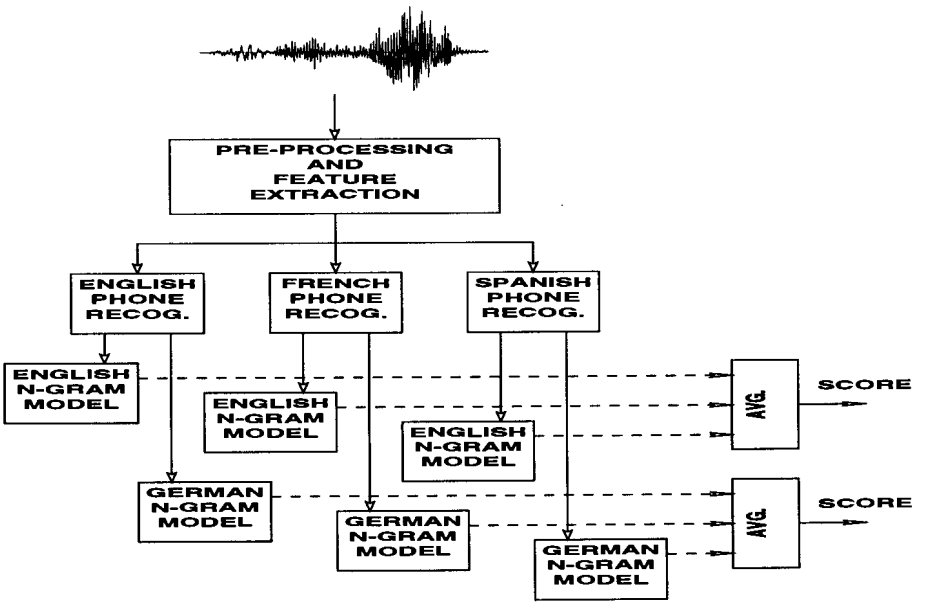

Figure 3.4: Parallel Phone Recognition followed by Language Modelling

described in the previous section. The languages used for the phone recognisers need not be the same as all, or even any of those recognised [45, 7, 47, 46]. In the same study mentioned in the previous section, Zissman's PPRLM system achieved 63% and 79% on the 5s and 45s utterance tests [7].

We have also implemented such a system. Section 4.5 details the experiments and results for this system.

3.7.6 Language dependent parallel phone recognition

Another variation on the previous two architectures is language dependent Parallel Phone Recognition. Here, the language model for a specific language is incorporated into the phone recogniser, improving its performance. One now essentially has a bank of fully-fledged, phone-level continuous speech recognition systems in parallel. Each recogniser produces a phone string hypothesis and a probability score. The language with the highest score wins. Note that in this case one needs labelled speech for every

language to be recognised (this not being the case for any of the previous systems). Zissman [7] has implemented and tested such a system. It performed slightly worse than the PPRLM system on a 3-language task.

3.7.7 Large vocabulary continuous speech recognition

At the top end of the complexity scale are Large Vocabulary Continuous Speech Recognition (LVCSR) systems [60, 61, 62]. The rationale behind this approach is that, once it is known *what* a person is saying, it becomes trivial to determine the language. LVCSR systems use a number of fully-fledged continuous speech recognition systems in parallel – one for each language under consideration. Each system produces a word level transcription hypothesis for a test utterance, together with a likelihood score. The language of the system that produces the hypothesis with the highest likelihood is taken to be the language of the test utterance. Hieronymus and Kadambe [60] tested their system on a 5-language task (English, German, Spanish, Japanese and Mandarin) using the OGI_TS corpus. The best recognition rates were 93% for 10s utterances and 98% for 45s utterances.

3.7.8 Discussion

We have briefly discussed the various classes of ALI architectures that have evolved during the history of ALI research and how they relate to each other in terms of complexity and performance. We now provide a summary of this chapter.

3.8 Summary

Spoken language. We presented a condensed introduction to relevant linguistic concepts and expressed our opinion that linguistic knowledge is important in developing high-performance spoken language systems.

Auditory perception and feature extraction. We discussed the mel-scaled cepstrum feature extraction algorithm in the context of significant human auditory phenomena and found it to be a good engineering solution compared to other standard approaches. However, as processing power increases, we may find it rewarding to move to more advanced auditory models.

Vector quantisation. This section introduced vector quantisation and presented the details of the Self-Creating and Organising Neural Network algorithm used in one of our systems.

Phone recognition and hidden Markov models. The structure and theoretical foundations of HMMs were presented and we explained its use as a phone recogniser in spoken language systems.

Language modelling. We formulated the N-gram modelling approach and introduced an N-gram distinctiveness measure used in our systems.

ALI architectures. In conclusion we showed how the above building blocks can be used to create ALI systems.

In the next chapter we present our own ALI system implementations, experiments and results.

Chapter 4

Experiments and results

4.1 Introduction

In this chapter we describe our experimental framework, the experiments that were performed and relevant results. As previously explained, we aim to explore the performance-cost-complexity relationship between a fully-fledged, state-of-the-art ALI system and a much simpler data-driven alternative.

4.2 Data

We used fine-phonetically labelled data from the OGI Multi-language Telephone Speech corpus, provided by the Centre for Spoken Language Understanding of the Oregon Graduate Institute [4, 6].

4.2.1 OGI telephone speech corpus

The OGI Multi-language Telephone Speech corpus (designated OGI_TS in literature) contains telephone speech from 11 languages. The data was recorded over various telephone channels and sampled at 8kHz. There are roughly two times as many male as female speakers. The initial collection, collected by Yeshwant Muthusamy for his Ph.D. dissertation research [4], included 900 calls - 90 in each of 10 languages: English, Farsi, French, German, Japanese, Korean, Mandarin, Spanish, Tamil and Vietnamese. It is from this initial set that Muthusamy established training (50 calls), development (20) and test (20) sets for his work. The National Institute of Standards and Technology (NIST) used the same sets in their annual evaluation of ALI systems [109]. Amongst others, each speaker provides about 60 seconds of speech that consists of a monologue on a subject of the speaker's choice. A tone was played at the 50s mark while recording these monologues. The speech samples before and after the tone were labelled as *story-bt* and *story-at* respectively. We used the *story-bt* data. The speech files are accompanied by time-aligned, fine-phonetic transcriptions for six of the languages: English (148 calls), German (100), Japanese (64), Mandarin (70), Spanish (102) and Hindi (68). Since the Hindi data was added at a later time under slightly different conditions, we only used data from the first five languages.

4.2.2 Data statistics

Table 4.1 provides statistics on the data that we used. The columns indicate the various languages. The rows show the number of speakers (male, female and total), as well as the amount of data measured in minutes, for the various data sets. Initially we used Muthusamy's original training/development/test division, however, because some of the utterances in the development sets had corrupted, incomplete or no label files, this division was revised. Appendix A details the final data sets on a file-by-file

Data set	Subset	EN	GE	JA	MA	SP	TOT
Train	Male	30	22	27	30	32	141
Train	Female	14	24	19	15	16	88
Train	Total	44	46	46	45	48	229
Train	Time (min.)	35	36	36	29	38	174
Develop	Male	12	11	14	9	15	61
Develop	Female	6	8	4	9	4	31
Develop	Total	18	19	18	18	19	92
Develop	Time (min.)	15	15	14	13	14	71
Test	Male	15	15	11	13	10	64
Test	Female	4	5	9	6	7	31
Test	Total	19	20	20	19	17	95
Test	Time (min.)	15	16	15	15	13	74
All	Male	57	48	52	52	57	266
All	Female	24	37	32	30	27	150
All	Total	81	85	84	82	84	416
All	Time (min.)	65	67	65	57	65	319

Table 4.1: Data set breakdown.

basis.

4.2.3 Discussion

The OGI_TS allowed for the first time direct comparison of independently developed ALI systems. Using it allows us to reliably benchmark our own systems and compare it with other published results. We continue in the next section to describe the experimental framework in terms of the hardware and software that we used.

4.3 Hardware and software

4.3.1 Hardware and operating system platform

All the software was developed in, and for, a UNIX environment. Although development and testing took place on a number of different platforms, it was mostly Linux running on various Intel x86 machines. The final experiments were done on a number of 350MHz Intel Pentium II' s with 128MB of RAM, running Linux.

4.3.2 Software

Since the project documented in this dissertation coincided with the birth of our local spoken language research group, there was no previous infrastructure on which to build. Unaware of the large investment in software that this type of research requires, we started developing our own software. This resulted in the SPoken Language Analysis Toolkit (SPLAT), a set of software tools for manipulating speech and label files, as well as applying various feature extraction schemes and performing discrete hidden Markov modelling. In addition we used the Hidden Markov model Toolkit for Speech Recognition (HMTSR) to experiment with continuous density hidden Markov models. HMTSR was developed by other group members, concurrently with our research.

4.3.2.1 SPLAT

The SPoken Language Analysis Toolkit was developed over a period of three years. It consists of a number of processing blocks that were implemented in as general a way as possible to allow experimental freedom. Writing the software was a learning experience, so efficiency came second many times to ease, elegance and consistency of implementation. Integrity always had highest priority. Where possible, results

were tested against examples and common sense. Sometimes (as with the mel-scaled cepstrum features) this was not really possible and we had to be content with something that 'looked right'. Much care has been taken to ensure robustness and to keep the system modular and generic. Unfortunately, again, these qualities were traded for reduced efficiency. SPLAT only implemented discrete hidden Markov modelling, which is a sub-optimal solution for large spoken language systems. Though not a very practical solution for building spoken language systems, it did serve us well as a tool for initial exploration of the necessary concepts. The VQLM system was implemented using SPLAT. Relevant experiments are documented in Section 4.6. Please see Appendix C for a description of SPLAT components.

4.3.2.2 HMTSR

The Hidden Markov model Toolkit for Speech Recognition, developed by Darryl Purnell and Christoph Nieuwoudt[1], provides software components for training and testing continuous hidden Markov models from speech data. We used it to build phone models in the phone recognition front-ends. Our PPRLM system is designed around this software. The experiments detailed in Section 4.5 were all performed using the HMTSR system.

4.3.3 Discussion

All the software that we used was developed locally, amounting to thousands of man hours. Neither of our systems scale well with the amount of data used and is inefficient when faced with the large corpora that have become standard. It would appear that speech processing systems are inherently complex and that such systems need a

[1]Pattern Recognition Group, Department of Electrical and Electronic Engineering, University of Pretoria.

very large initial development investment in order to produce potentially competitive results. We have, however, gained much practical experience during the process.

The following section deals with our first experiments, exploring the feasibility of automatic language identification.

4.4 Text-based test of language modelling back-end

We performed this experiment early in our research in order to test a claim by House and Neuberg [29] about the distinctiveness of phoneme N-grams in a language identification context. Having only access to text data at the time, we assumed that the distinctiveness of letter N-grams are roughly comparable to those of phoneme or phone N-grams.

4.4.1 System description

The system consists of a text pre-processor, a training module and a recognition module. Raw ASCII text files were pre-processed by converting all upper case to lower case letters and then filtering out everything in the file except the twenty-six lower case letters and spaces; every file contains at most twenty-seven symbols. Some languages contain symbols that are not part of the standard ASCII character set. These were either converted to one of the twenty-seven symbols, or just omitted. This is not strictly necessary (we have tested the system with the raw files and it still works well), but serves to increase the signal to noise ratio in most cases. Where language specific characters are concerned, it might seem that one is throwing away valuable information. Unfortunately these symbols are in many cases associated with a specific character set, making it useless when documents from many different sources are compared. The letter filters can of course be adapted to allow a greater variety of symbols if one

uses the system in a well-defined environment.

During the training stage a discriminative N-gram language model is constructed for each training set as detailed in Section 3.6.2. When tested, the system is presented with a previously unseen piece of text. The text is scanned for occurrences of the N-grams representative of the different language models. In this way the system builds a histogram for each language of the number of hits for the various N-grams as described in Section 3.6.2. The test text is classified as belonging to the language model with the highest score.

4.4.2 Data and experiments

The system was tested on a text corpus of the following twelve languages: Afrikaans, English, Sepedi, Xhosa, Zulu, Tswana, Swazi, German, Italian, French, Spanish and Portuguese. The data for the training and test sets came from (different parts of) a single text in the following cases: French, Italian, Portuguese, Sepedi and Xhosa. We chose English as an example and measured classification performance as a function of the training set size, the size of the representative model of the language and the test set size.

4.4.3 Results and interpretation

The system obtained 100 percent correct classification of all the test texts. In order to measure differences in performance meaningfully while achieving perfect language recognition, we used a *relative performance* measure. It is defined as the ratio of the histogram score attained by the language model representing the correct language, to that of the largest of the other models. Accordingly a value of "2" indicates that the test instance was correctly classified with the output being twice as high as that of the runner up. Similarly a figure smaller than one would represent a misclassification. In

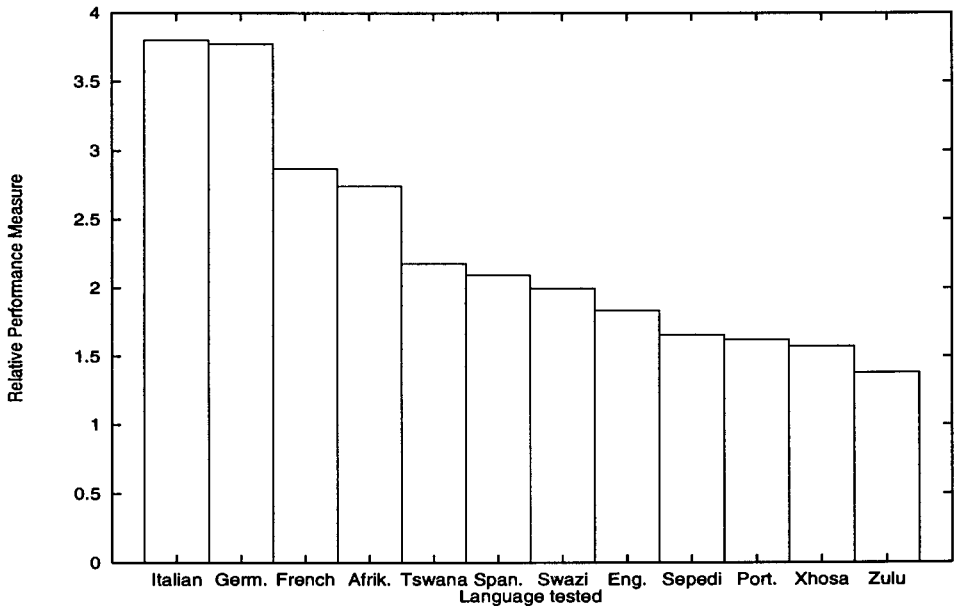

Figure 4.1: Relative performance of different languages.

Figure 4.1 the performance of the system in terms of the relative performance measure is shown for each of the test languages. The results were obtained by making four test runs for each language and averaging the results. All of the individual test runs also achieved 100 percent classification.

4.4.3.1 Effect of training set size on classification performance

Figure 4.2 is a graph of the relative performance measure as a function of the number of characters per language in the training set. The system is trained using texts of equal length to prevent *a priori* bias of the language models. The performance degrades sharply for training sets exceeding 1400 characters on average for this specific mix of languages. The result may appear counter-intuitive at first, as one might expect the system to perform better when trained with more data. Since even 5000 characters still fall within a single text for most of the languages, this behaviour may be ascribed to over-fitting of the model to the idiosyncrasies of the specific training set at the

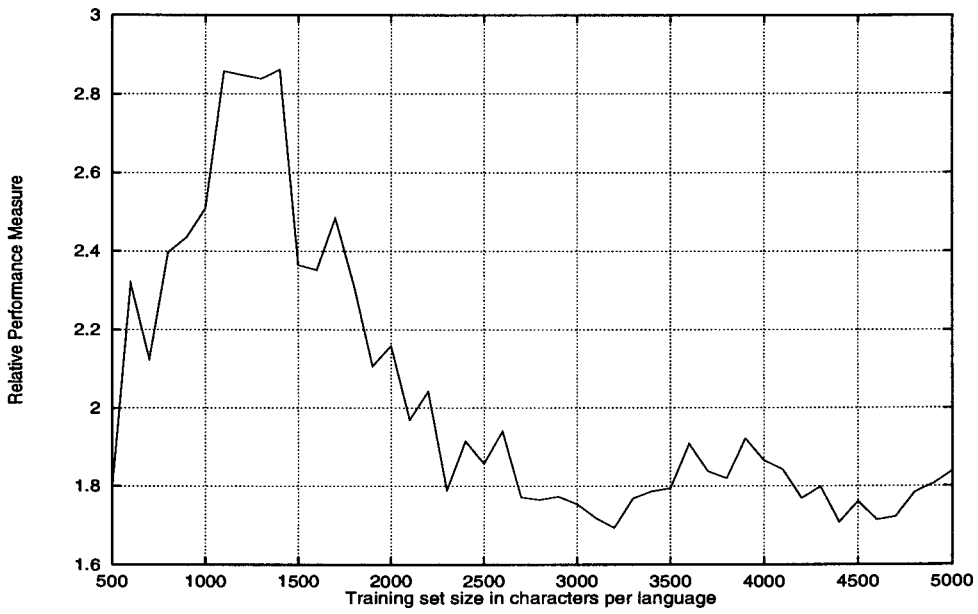

Figure 4.2: Relative performance as a function of training set size.

cost of failing to capture the general characteristics of the language. Nevertheless, it does seem to be a surprisingly small text sample when one keeps the complexity of a whole language in mind. If one were to extend the training set over a (large) number of different texts for each language, one would expect performance to increase again with training set size.

4.4.3.2 Effect of language model size on classification performance

Since the language models are normalised to equalise the *a priori* probabilities of the languages, the models do not necessarily contain the same number of N-grams. The size used in the graph is that of the smallest model. Figure 4.3 suggests that there exists an optimum point for model size. Though the system seems to work best for extremely small models, its operation is unstable and very sensitive to model size in that region. This is an artifact of the relative performance measure that is used. Since this value is calculated as a ratio of occurrence frequencies, it does not have statistical

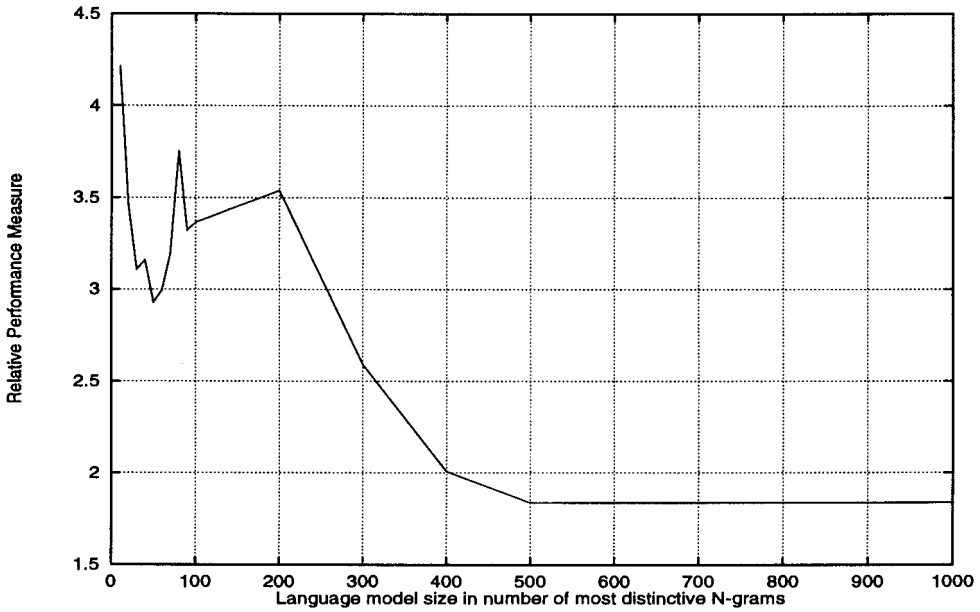

Figure 4.3: Relative performance as a function of grammar size.

significance for small numbers. Still, it is clear that system performance is adversely affected by large models. When allowed to grow too large the system incorporates N-grams with low discriminatory value, that is letter combinations that are not specific enough to a certain language and consequently give rise to false alarms.

4.4.3.3 Effect of test set size on classification performance

The classification performance of the system is obviously a function of the length of the piece of text that it analyses. Similar to Figure 4.3, Figure 4.4 seems to suggest that the best results are obtained with very short texts, but again the system is unstable when confronted with too little data. The result can be seen to stabilise for larger test sets as the statistical significance increases for larger numbers.

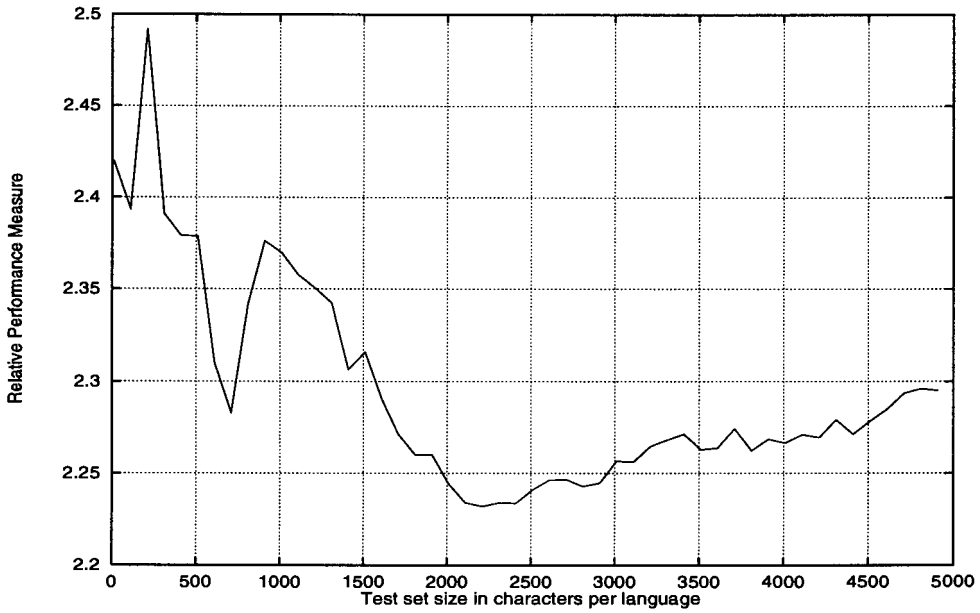

Figure 4.4: Relative performance as a function of test set size.

4.4.4 Discussion

We have presented a text-based automatic language identification system and investigated the effect of a number of parameters on system performance. Performance was measured in terms of a relative performance measure. We have shown that there exist optimum sizes for the training set as well as the language model and that the test set should be large enough for the classifier to stabilise. It is clear that letter N-grams are in fact very distinctive as far as language is concerned and therefore supports the results of House and Neuberg [29].

4.5 PPRLM ALI system

4.5.1 System description

Our *Parallel monolingual Phone Recognition followed by language dependent N-gram Language Modelling* system consists of two language-specific phone recognisers operating in parallel, followed by an array of 2×2 language models for the various 2-language tasks. During training, labelled speech data from a specific language is used to estimate the parameters for a number of CDHMMs - each modelling a single phone. Such a set of HMMs constitute a language-dependent phone recogniser. When applied to a test utterance, the system uses a search algorithm in conjunction with the HMMs to find the most likely phone boundaries and phone transcriptions. The phone transcriptions are analysed using a set of discriminatively trained, phone bigram language models which provide a set of language scores - one for each hypothesis. The score for a specific language is summed over the various models and the hypothesis with the highest score wins. Although this need not be the case, the languages used for the phone recognisers are the same as those recognised. An important parameter in the language model is the number of adjacent phones, N, that are used to estimate N-gram statistics. Distinctiveness implies that the frequency of an N-gram for a given language should be high relative to the frequency of occurrence in other languages. In addition, the absolute frequency of occurrence of a feature should be high enough to have statistical significance. The more tokens in the feature, the higher its discriminatory value, but the lower its probability of occurrence. These two criteria constitute a trade-off that is a strong function of the length of N-grams used for features. In preliminary experiments we found $N = 2$ to be a practical compromise.

4.5.2 Experiments

The performance of the phone recognition system is sensitive with regard to the number of states and Gaussian mixtures that are used in the HMMs. We performed a series of experiments to find the optimal number of states and Gaussian mixtures for our problem. We assume that the two parameters affect performance independently (although this is not quite true). Given the optimal values for these two parameters, we further investigated the effect of the language model size and utterance duration on system performance. No assumption regarding the independence of these two parameters were made. All the above experiments were performed on the development set. The final experiment was an overall system performance evaluation on the test set using the optimal values for various parameters as suggested by the results of the relevant experiments.

The training set contained approximately 35 minutes of speech data per language and the development and test sets each about 15 minutes per language. A single 2-language task experiment (training and testing phase) processed therefore 100 minutes of speech and used ± 60 hours of computer time on a 350MHz Pentium II with 128MB of RAM. Though this figure can be substantially reduced in various ways, searching for optimal system performance through parameter space is obviously very expensive and we had to limit such a search to, and assume independence between, a few salient parameters.

4.5.3 Results and interpretation

We present the various PPRLM experiments and results, providing us with quasi-optimal parameters settings and an indication of the performance of the PPRLM system on 2-language tasks. A note on the operation of the N-gram language model classifier is in order here: Since the score that it generates for each language is an in-

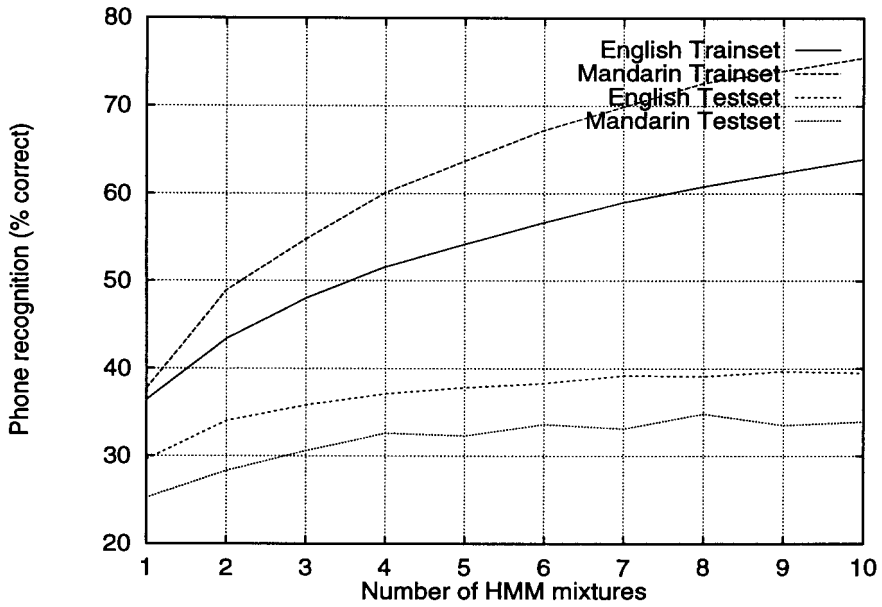

Figure 4.5: Phone recognition rate as a function of number of HMM mixtures.

teger value directly related to the number of occurrences of language-specific N-grams, it may happen that the system produces identical scores for two language hypotheses. In such a case there is no ground for forcing a classification and the result is undecided. We have treated all such cases as a "wrong" classification, rather than a "rejection". System performance stated in terms of "percent correct" is therefore calculated throughout in the stricter sense as *percentage of utterances correctly classified out of the total number of utterances* presented to the system, as opposed to the number utterances correctly classified out of those not "rejected".

4.5.3.1 Effect of number of HMM Gaussian mixtures on phone recognition performance

We varied the number of HMM mixtures from 1 to 10 for two phone recognition systems – one English and one Mandarin. The recognition rates were reported both as *percentage correct* and *accuracy* scores. *Accuracy* is defined as percentage correct

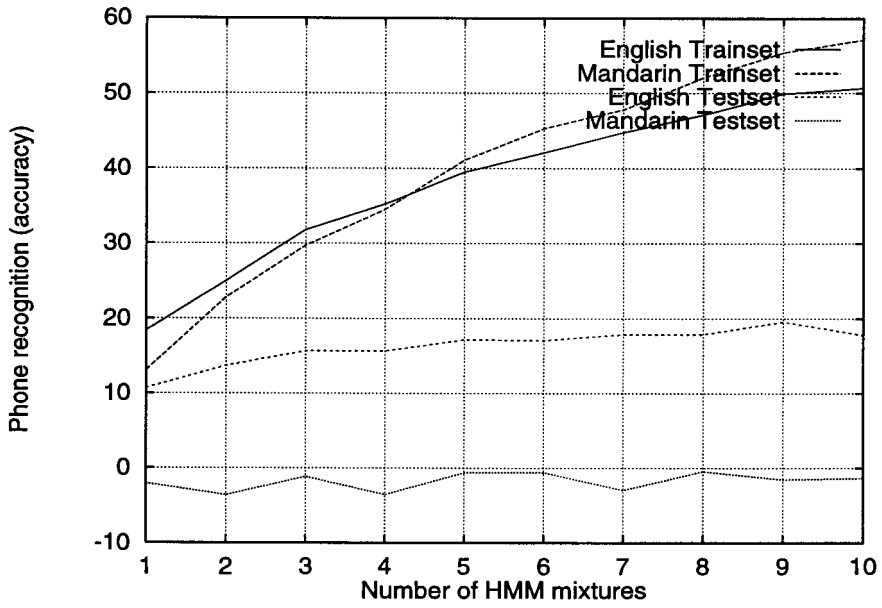

Figure 4.6: Phone recognition accuracy as a function of number of HMM mixtures.

minus insertions, deletions and substitutions (and can hence be a negative number). It is a better measure of performance than simply taking the percentage of correct phones calculated by pairing the test result and correct phone string phone-by-phone. Figure 4.5 shows the percentage correct results for the training and development sets and Figure 4.6, the accuracy results. Performance seemed to stabilise at six mixtures and we used this figure in further experiments.

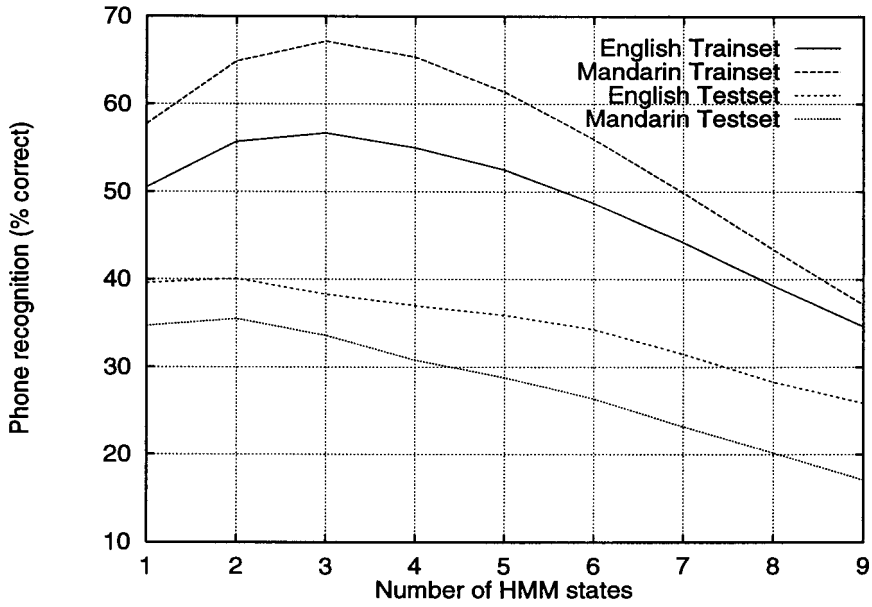

Figure 4.7: Phone recognition rate as a function of number of HMM states.

4.5.3.2 Effect of number of HMM states on phone recognition performance

In this experiment we varied the number of HMM states from 1 to 9 - again for the English and Mandarin phone recognition systems. The results of the previous experiment suggested that the number of HMM mixtures be set to six for these tests. Figure 4.7 show the percentage correct results for the training and development sets and Figure 4.8, the accuracy results. From the latter we chose the number of HMM states to be six.

4.5.3.3 Effect of language model size on language identification performance

The next parameter that we experimented with is the size of the language model, i.e. the number of significantly discriminative phone bigrams used in the language scoring process. Since this number may vary widely depending on how closely related the languages under consideration is, we expressed this quantity implicitly in another parameter - the *minimum distinctiveness measure value.* During training of the language

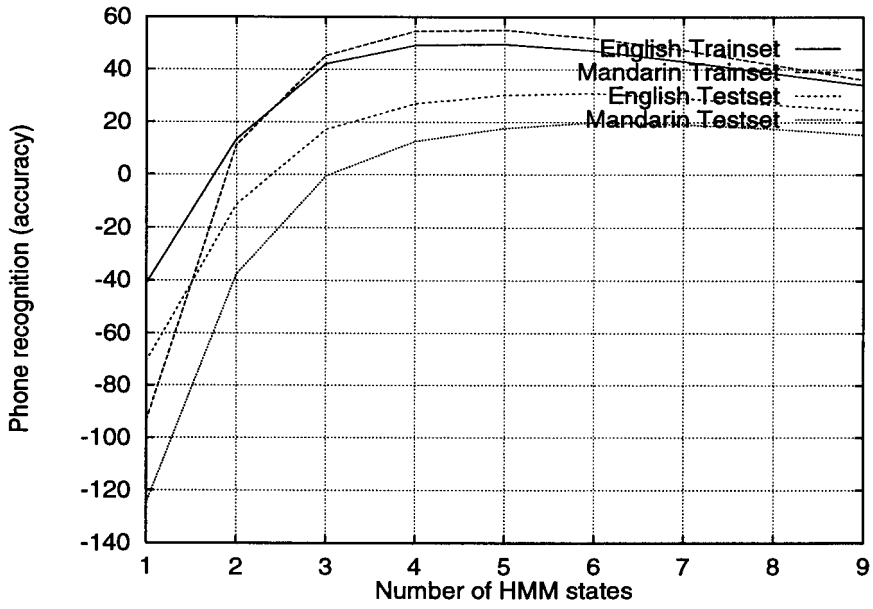

Figure 4.8: Phone recognition accuracy as a function of number of HMM states.

model, only bigrams with a distinctiveness measure value greater than this parameter are accepted into the language model. This constraint is applied before equation 3.69. The higher the threshold, the smaller the language model and the chance of overfitting. In an English-Spanish task, with a minimum distinctiveness measure of 1.0, for example, the English model contains 390 bigrams and the Spanish model 562 bigrams in terms of the English VQ codebook and 529 and 665 bigrams respectively in terms of the Spanish VQ codebook.

The distinctiveness measure (equation 3.68) is only defined in the range 0.5 ($\frac{1}{N}$, for N languages) to 1.0. In this set of experiments the minimum distinctiveness measure value was swept from 0.5 to 1.0 in 0.1 increments for all ten pair-wise language permutations. The mean performance of the ten systems on the training and development sets are shown in Figure 4.9 and Figure 4.10 respectively. Figure 4.10 suggests an optimal value of 0.6. The results for individual language pairs are documented in Sections B.2 and B.3.

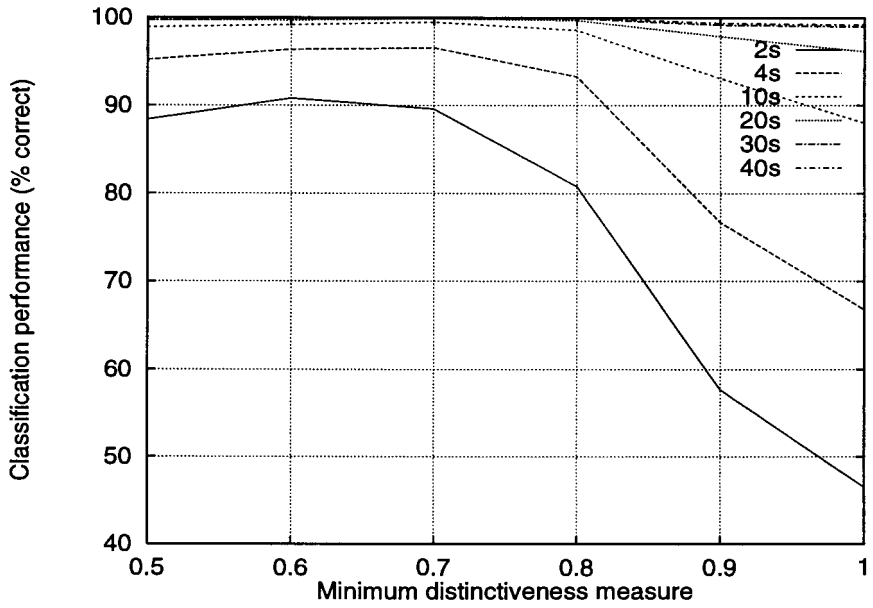

Figure 4.9: Mean language classification performance on the training set as a function of the minimum distinctiveness measure value and utterance duration.

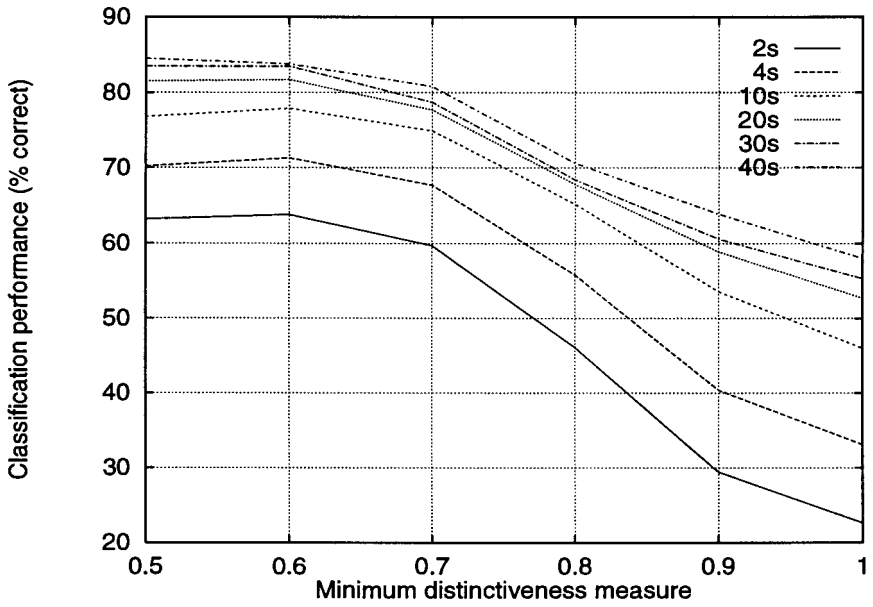

Figure 4.10: Mean language classification performance on the development set as a function of the minimum distinctiveness measure value and utterance duration.

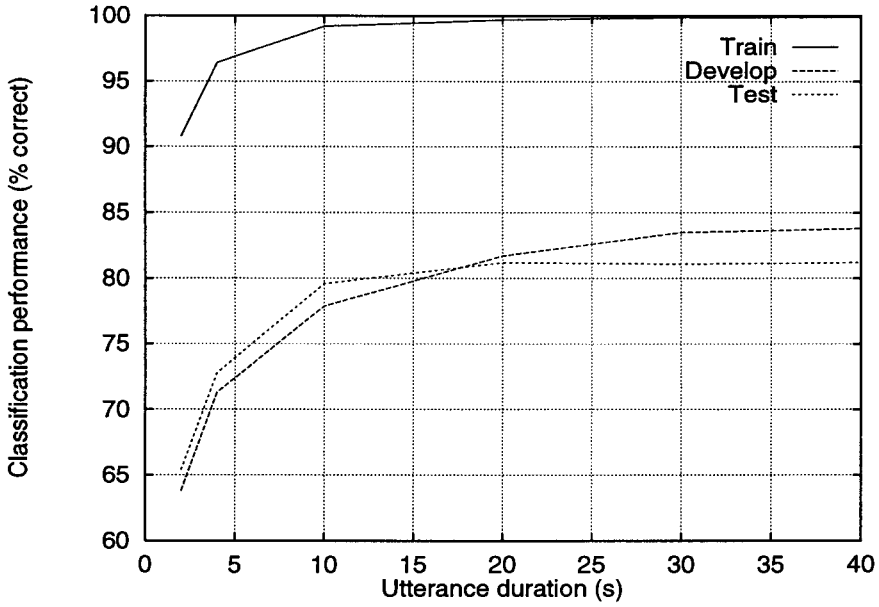

Figure 4.11: Mean language classification performance on the training, development and final test sets as a function of utterance duration.

4.5.3.4 Effect of utterance duration on language identification performance

The last parameters that we considered experimentally was utterance duration. Obviously the system should perform better with more information, but one would also expect performance to level off after a certain length of time. One of the goals of ALI is to achieve adequate performance on as little speech as possible. Figure 4.11 show system behaviour as a function of utterance duration. As expected, performance asymptotes with increasing utterance duration.

4.5.3.5 Evaluation of system performance on test set

Finally, the system was configured with optimal parameters as suggested by the previous experiments (summarised in Table 4.2) and tested on the OGI_TS test sets. The results are shown in Table 4.3 and Figure 4.11. The numbers in parentheses in Table 4.3 are the number of "undecided" utterances (see Section 4.5.3) and were taken as

Parameter	Value
HMM Gaussian mixtures	6
HMM states	6
Min. distinctiveness measure	0.6

Table 4.2: Quasi-optimal parameter set for PPRLM system.

incorrectly classified. The largest number (seven utterances for English vs. Mandarin) accounts for approximately 4 percent of the 10s utterances classified in that task.

4.5.4 Discussion

We presented four sets of experiments to determine good values for certain PPRLM system parameters. In the final set the system was configured with these values and tested on test data, presented for the first time to the system. It achieved average recognition rates of 79.6% (10s utterances) and 81.2% (40s utterances) over the ten pair-wise language recognition tasks.

Language Pair	10s utterance	40s utterance
English - German	85% (4)	86%
English - Japanese	92% (1)	97%
English - Mandarin	84% (7)	83% (1)
English - Spanish	80% (5)	80%
German - Japanese	86% (4)	89%
German - Mandarin	79% (4)	76%
German - Spanish	80% (6)	86%
Japanese - Mandarin	77% (4)	78%
Japanese - Spanish	58% (9)	63%
Mandarin - Spanish	75% (3)	74%

Table 4.3: PPRLM final results on two-language tasks.

4.6 VQLM ALI system

4.6.1 System description

As with the PPRLM system, the VQLM system is based on the fact that the sequential organisation of discrete units in spoken language provide language-specific information that can be used to identify that language. During training, the speech signal is broken into overlapping frames by moving a fixed-length analysis window over the signal in fixed increments. The frames are represented as points in feature space by extracting acoustic features for each frame. Feature space is vector quantised. This results in a VQ codebook for each language which represents a set of sub-phonetic tokens for that language. The speech stream (frame-based features) is then coded using the VQ codebook. The resulting sequence of tokens is used to train an array of 2×2 language models in the same way as with the PPRLM system. A test utterance is quantised using both VQ codebooks and the token sequences are analysed using a set of discriminatively trained, phone bigram language models which provide a set of language scores - one for each hypothesis. The score for a specific language is summed over the various models and the hypothesis with the highest score wins.

4.6.2 Experiments

The most prominent parameters in the VQLM system are the VQ codebook size, language model size and utterance duration. Accordingly we performed three sets of experiments to evaluate the effect of these quantities on system performance as well as a set of experiments on the test set as a final benchmark.

The training set contained approximately 35 minutes of speech data per language and the development and test sets each about 15 minutes per language. A single 2-language task experiment (training and testing phase) processed therefore 100 min-

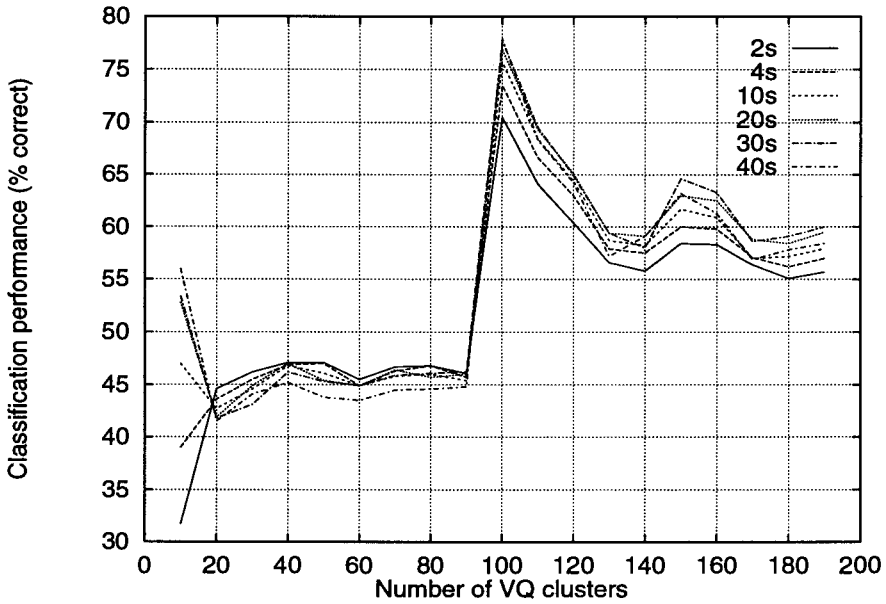

Figure 4.12: Mean language classification performance on the training set as a function of VQ codebook size and utterance duration.

utes of speech and used ± 1 hour of computer time on a 350MHz Pentium II with 128MB of RAM. This is sixty times faster than the PPRLM system.

4.6.3 Results and interpretation

We present the various VQLM experiments and results, providing us with quasi-optimal parameters settings and an indication of the performance of the VQLM system on a set of 2-language tasks.

4.6.3.1 Effect of vector quantisation codebook size on language identification performance

The first set of experiments is aimed at determining an optimal codebook size. On the one hand a large codebook is preferable, since it provides finer resolution in dis-

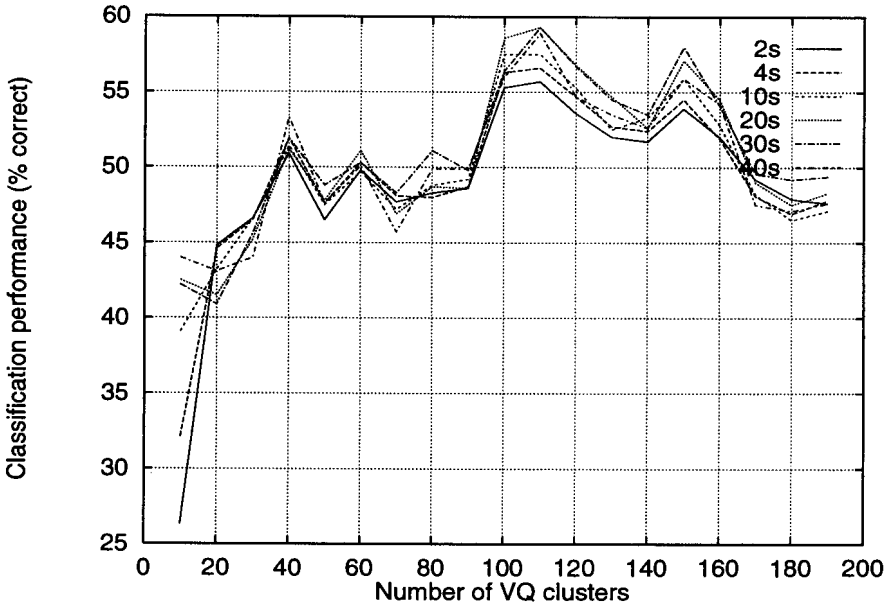

Figure 4.13: Mean language classification performance on the development set as a function of VQ codebook size and utterance duration.

tinguishing between potentially different sub-phonetic units. On the other hand, the number of possible bigram probabilities grows quadratically with codebook size and manifests as complexity in the language model that needs more data to train satisfactorily. The system was tested with codebook sizes ranging from 10 to 190 on all ten pair-wise language permutations. The mean performance of the ten systems on the training and development sets are shown in Figure 4.12 and Figure 4.13 respectively. The latter suggests a codebook size of 110.

The discontinuity and sharp peak at 100 in Figure 4.12 invites discussion. It is possible that this point represents the mean total number of (sub) phone-like units naturally occurring in two languages. Below this point significantly different sounds would be forced together, degrading performance, while above it the statistics belonging to a single unit would be diffused across multiple units with a resulting higher confusion rate. In addition it may appear that the classifier is performing worse than chance (50%) to the left of the peak. Note, however, that since the classifier recognises an

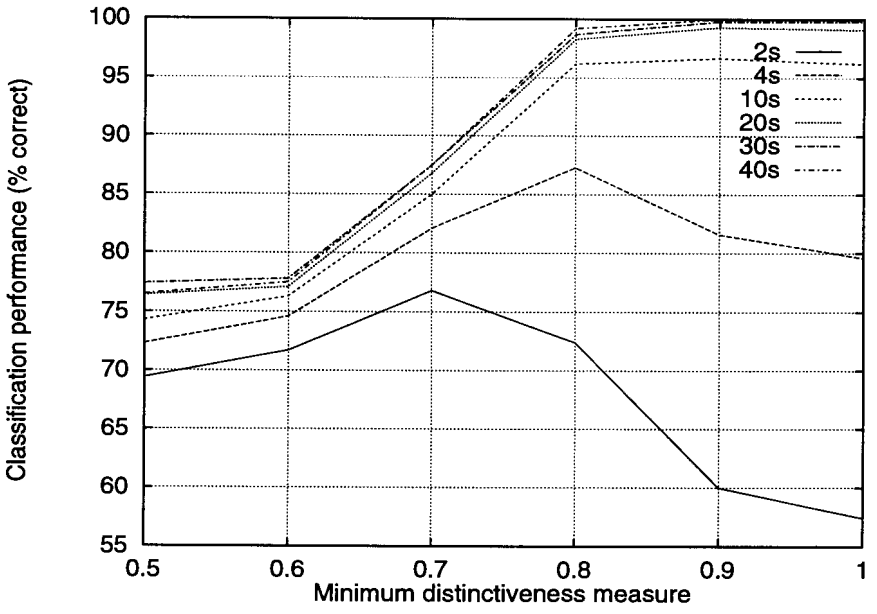

Figure 4.14: Mean language classification performance on the training set as a function of the minimum distinctiveness measure value and utterance duration.

"undecided" class in addition to the two language hypotheses, as explained in Section 4.5.3, chance performance should be calculated for a three-class, rather than a two-class system. The exact number is smaller than 50.0%, but larger than 33.3% and is a non-trivial function of the length of the test utterance. The number of undecided utterance are nevertheless very small - less than one percent over nearly the entire region represented in Figure 4.12. (The figure does become significantly larger in some regions of parameter space.) The results for individual language pairs are documented in Sections B.4 and B.5.

4.6.3.2 Effect of language model size on language identification performance

This set of experiments is identical in nature to those in Section 4.5.3.3, performed for the PPRLM system. Again the minimum distinctiveness measure value was swept from 0.5 to 1.0 in 0.1 increments for all ten pair-wise language permutations. The mean performance of the ten systems on the training and development sets are shown

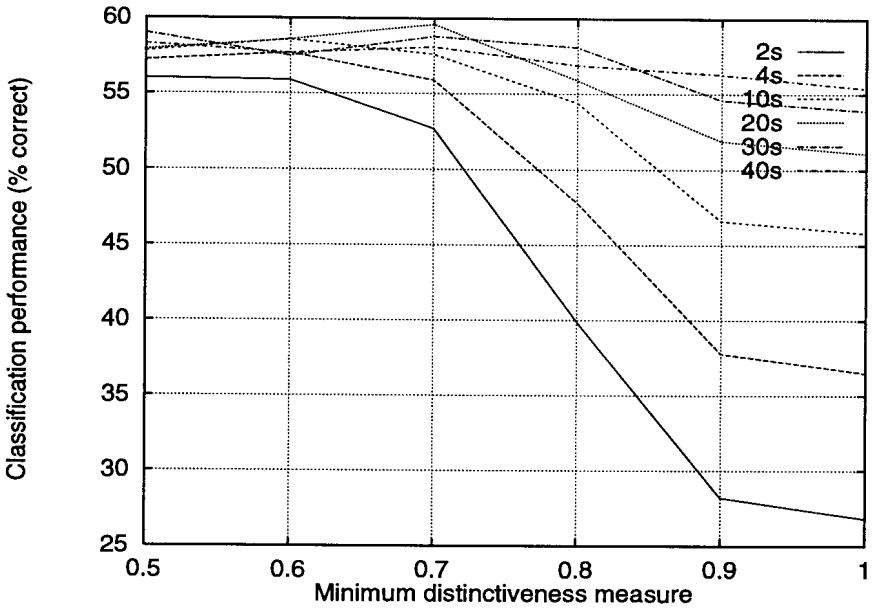

Figure 4.15: Mean language classification performance on the development set as a function of the minimum distinctiveness measure value and utterance duration.

in Figure 4.14 and Figure 4.15 respectively. Figure 4.15 suggests an optimal value of 0.6. The results for individual language pairs are documented in Sections B.6 and B.7.

4.6.3.3 Effect of utterance duration on language identification performance

The last parameter that we considered experimentally for the VQLM system was utterance duration. As with the PPRLM system, VQLM should perform better with more information and one would expect performance to level off after a certain length of time. Figure 4.16 shows system behaviour as a function of utterance duration. The results for different utterance durations are quite close to each other in the regions where the system performs well.

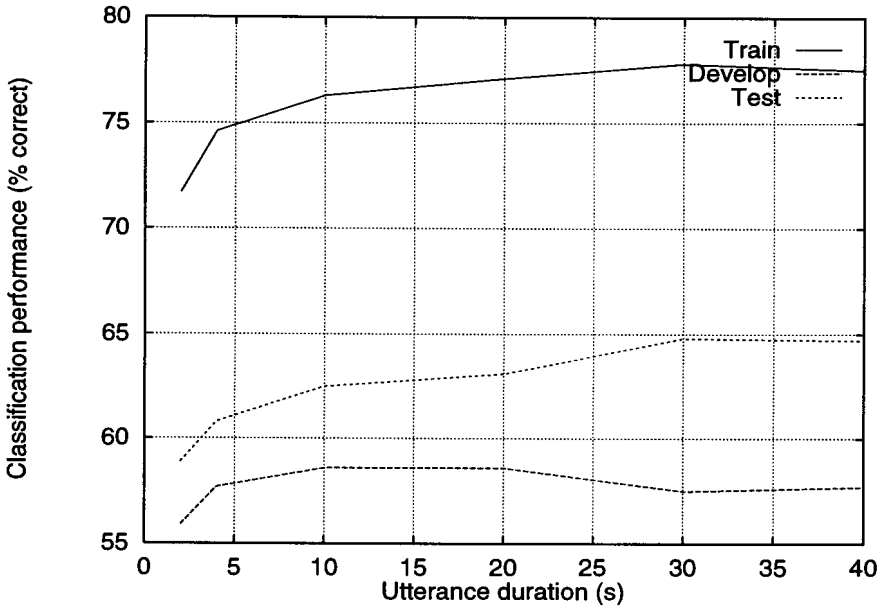

Figure 4.16: Mean language classification performance on the training, development and final test sets as a function of utterance duration.

4.6.3.4 Evaluation of system performance on test set

Finally, the system was configured with optimal parameters suggested by the previous experiments as summarised in Table 4.4 and tested on the OGI_TS test sets. The results are shown in Table 4.5 and Figure 4.16. The numbers in parentheses in Table 4.5 are the number of "undecided" utterances (see Section 4.5.3) and were taken as incorrectly classified.

Parameter	Value
VQ codebook size	110
Min. distinctiveness measure	0.6

Table 4.4: Quasi-optimal parameter set for VQLM system.

Language Pair	10s utterance	40s utterance
English - German	58% (1)	58%
English - Japanese	66%	68%
English - Mandarin	74% (1)	81%
English - Spanish	58% (1)	58%
German - Japanese	54%	60% (1)
German - Mandarin	73% (1)	78%
German - Spanish	59% (1)	61%
Japanese - Mandarin	62% (1)	58%
Japanese - Spanish	53%	54%
Mandarin - Spanish	68%	71%

Table 4.5: VQLM final results on two-language tasks.

4.6.4 Discussion

We presented four sets of experiments to determine good values for certain VQLM system parameters. In the final set the system was configured with these values and tested on test data, presented for the first time to the system. It achieved average recognition rates of 62.5% (10s utterances) and 64.7% (40s utterances) over the ten pairwise language recognition tasks. The system shows inferior performance compared to that of the PPRLM system (79.6% and 81.2% respectively). This is expected, given the simpler nature of the system and the fact that it incorporates much less *a priori* information. One should however bear in mind that these results are achieved with unlabelled data, implying large savings in development cost.

4.7 Summary

We have described our experimental environment and the data used, the two ALI systems that we have implemented, as well as various experiments and results. In the final chapter we present various conclusions concerning automatic language identifi-

cation, our work and the nature of spoken language systems in general.

Chapter 5

Conclusion

5.1 Introduction

In Chapter 1 we introduced automatic language identification as a concept and sketched the nature and context of the problem, as well as the outlines of a solution. Chapter 2 continued with a foray into the existing body of literature on ALI systems. It became clear that non-withstanding the bewildering variety in implementation detail, there are a number of standard components to such systems. Following a brief overview of relevant linguistic terms, these components were introduced and described in some detail in Chapter 3. In addition, we presented the more prominent ALI architecture families. Chapter 4 described our experimental framework, the experiments that we conducted and relevant results.

In this, the final chapter, we provide a retrospective overview of our work. In the next section we summarise our results and draw conclusions regarding the issue of performance vs. complexity in automatic language identification systems. Section 5.3 follows with a discussion of our research experience in a wider sense. Section 5.4 presents ideas for future work flowing from our increased understanding of the prob-

System	10s utterance	40s utterance
PPRLM	80%	81%
VQLM	63%	65%
State-of-the-art	90%	94%

Table 5.1: Comparison of final results.

lem and Section 5.5 concludes the dissertation.

5.2 Performance vs. complexity in automatic language identification systems

We implemented, and experimented with, two systems. A state-of-the-art architecture and our own much simpler, cheaper alternative. The overall language classification rates on a set of ten pair-wise language identification tasks are compared in Table 5.1. It is clear that the PPRLM system significantly outperformed the VQLM system. The former was, however, trained on about three hours of fine-phonetically hand-labelled speech data, requiring a vast amount of expensive skilled labour. It was also more complex, conceptually as well as implementation-wise. We had hoped that the simpler system could approach state-of-the-art performance, but it seems that it remains a clear trade-off between performance on the one hand and cost and complexity on the other.

Neither of our systems performed as well as the best reported in literature on a similar set of tasks [7] (confirm Table 5.1). We attribute this to the fact that both our systems are first-generation systems and cannot really be fairly compared to systems in literature that have the benefit of years of fine-tuning and dedicated research effort. As a matter of fact, the large discrepancy between the performance of our PPRLM system and state-of-the-art results shown in Table 5.1, would seem to suggest that both our

systems can benefit from additional refinement. If the difference between the PPRLM and state-of-the-art results is taken as measure, VQLM system performance might very well be raised to a level that is adequate for some environments.

5.3 Challenges, issues and insights

We have gained some valuable experience and insight into the nature of spoken language systems as well as research and development resulting in technically involved software systems. We share some conclusions:

5.3.1 Speech corpus development

Statement: The development of a high quality speech corpus is an activity worthy of project status, requiring expert knowledge, careful planning, a solid infrastructure and many hours of skilled labour.

We did initially intend to collect and label data for a local speech corpus, but found it beyond our immediate capabilities. Since different types of speech (conversational, read, prompted, monologue, etc.) has an impact on the performance of the system, one should be clear beforehand about the exact nature of the data required. Depending on the nature of the data, it might be necessary to have ready access to expert knowledge on phonetics and/or linguistics. In addition, the recording environment should be stable, well-tested and user-friendly, since the environment affects the subjects in ways that alters the nature and quality of the data. The same criteria applies to the labelling environment. The labelling process is tedious and time-consuming and ideally requires a small army of computer literate, mother-language phonetic experts that received additional training as demanded by the specific goals of the research project.

Corpus development is *not* something that one does quickly before starting with the development of speech technology.

5.3.2 Expert knowledge

Statement: Spoken language research is a multi-disciplinary activity that spans non-trivial concepts and skills over various expert domains and requires ready access to domain experts that are comfortable with multi-disciplinary interaction, as well as a wide range of applicable books and journals.

We found spoken language processing to be surprisingly resistant to the pattern recognition approach which attempts to view the speech signal as simply another signal that contains patterns that can be extracted without any additional knowledge about the origin, nature and context of the signal. Successful spoken language systems operate across various domains and levels of organisation (acoustic properties, perceptual issues, phonetic units, words, sentences, language structures, etc.) that each require expert knowledge. Maybe owing to the rapid growth of, specifically, automatic language identification, we found that research was mostly published in journals and technical conference proceedings (focusing on reporting results and conceptual advances), without the much-needed counterpart of tutorial papers and books that give attention to finer detail and implementation issues.

5.3.3 System complexity

Statement: Effective spoken language systems are complex.

Spoken language have proven to be remarkably resistant to simple solutions. It would seem that human spoken communication is an activity worthy of the most advanced form of computation known to us and that it will continue to demand cutting edge

technology for a long time to come. This fact has implications in all aspects of spoken language systems development.

5.3.4 Well-defined, limited problem specification

Statement: Because of the inherent complex nature of spoken language systems, it is vital to very carefully delimit the goals of any specific research project.

Automatic language identification, from a state-of-the-art point of view, is a high-level application, i.e. it requires certain components that are sophisticated systems which, in turn, touch on significant development issues of their own. Current systems require a continuous speech phone recogniser, which in turn requires a well-implemented hidden Markov modelling environment and a solid feature extraction component. Hidden Markov modelling and speech feature extraction are active areas of research themselves. When one focuses on a high-level task like automatic language identification, these components should be available on a state-of-the-art level.

5.3.5 Computing infrastructure

Statement: Spoken language systems are resource-intensive.

The explosive advance in spoken language systems over the past few years seems to be largely because the equally impressive increase in raw processing power has pushed spoken language system over the edge of feasibility. Software should be developed with a high premium on efficiency and even then, especially in an experimental environment, will require much processing power. Speech data also tend to require large amounts of storage space.

5.3.6 Discussion

It would seem that spoken language systems require a high degree of complexity that translates into a large initial development investment, before achieving any amount of success. In our opinion this fact can be traced back to the inherent complexity of human speech as a communication medium, as pointed out in Section 1.1. It is also reflected in the rather involved human auditory system (Section 3.3). Despite the fact that spoken language systems function reasonably well and are continually improving, they still lag behind human performance, especially under adverse conditions. At the same time though, paradoxically, current spoken language systems perform surprisingly well when one keeps in mind the involved nature of the articulatory and auditory systems and the formidable processing ability of the human mind.

5.4 Future work

As a general comment one might note that since state-of-the-art ALI systems rely on continuous speech phone recognition as a first processing step, improved continuous speech recognition holds obvious performance benefits for ALI systems. We mention a number of more specific issues:

5.4.1 Feature extraction

It would seem that the most basic aspects of a spoken language system are those of **feature extraction**, **knowledge representation** and **search** (optimisation over a discrete, non-ordered space). The last two are very general technologies and does not concern us directly here. In our view, however, the features used in spoken language systems need more attention because they can potentially affect the performance of such systems significantly [38, 7]. In addition it would seem that the effectiveness of

features depends on the language(s) under consideration [57].

Current systems largely ignore the dynamic nature of the auditory system. Specifically, masking and forward masking are important concepts that need to be integrated in an efficient way into spoken language feature extraction systems [110]. As processing power increase we may find it rewarding to move to more advanced auditory models.

5.4.2 N-gram language modelling

5.4.2.1 Vector quantisation

The vector quantisation algorithm (SCONN), used in our work, has a number of parameters that affect its performance. Amongst others, one has to choose the number of clusters or codebook size. It is preferable to rather let this number flow naturally from the structure of the data in feature space. One way of doing this might be by using the minimum description length principle that takes into account both the information contained in the quantised data and in the model and attempts to minimise the complete description of the original data, aiming for better generalisation [111].

5.4.2.2 Distinctiveness measure

The measure that is currently used to evaluate the distinctiveness of N-grams is a heuristic. It evaluates the N-grams primarily according to their distinctiveness and secondarily according to their frequency of occurrence. While it takes the absolute frequency of occurrence into account, it only does so marginally. The ideal distinctiveness measure should blend the effect of relative and total frequency of occurrence in an elegant and effective way. We suggest a closer look at the mutual information measure which has a much firmer basis in information theory [112], as well as the minimum description length principle [111] and other methods that can be used to

select features that are optimal for classification [113]. Another approach would be to use minimum classification error (MCE) training [114] to optimise the parameters of a function that combines the information implied by the relative and absolute occurrence frequency of N-grams. In addition one can attempt to find N-grams that are both maximally distinctive between languages and *least distinctive among same-language utterances*. Although the various measures might perform comparably under favourable circumstances, one should ask how their performance degrade with degrading accuracy of the token string (phone transcription) that is analysed.

5.4.2.3 N-gram weighting

The information gained by the more advanced distinctiveness measures suggested in the previous section can also be used more efficiently if retained in the N-gram language model. Each N-gram can be accompanied by a weight that represents its distinctiveness. Such a weight will allow the system to affect the classification score in a manner proportional to the amount of information that an N-gram it carries, rather than simply noting its presence in a way that implies all N-grams to be equal.

5.4.3 Alternative approaches to ALI

5.4.3.1 Phone clustering in HMM space

We have considered an interesting compromise between the two system that we implemented. If one could retain the discriminative power arising from the high information content of phones and phonotactics without the burden of having to manually label speech data, it should allow for a powerful ALI system. One could start with approximate phone boundaries (automatically segmented) and build a large number of quasi-phone HMMs from the training data. Clustering in HMM space may then lead to

a set of HMMs that closely approximate phones. Rabiner and Juang suggest a number of distance measures in HMM space [108] that can be useful in this regard.

5.4.3.2 The comprehensive speech signal information extractor

We mentioned in Section 1.1 that human speech, apart from being massively redundant, contains information about the sex, age, unique vocal tract, socio-economical background, native geographical location, emotional status, nationality and language of the speaker. In addition, the signal is altered by background noise as well as distortion introduced by the communication channel. As a matter of fact, if one works with the whole audio band accessible to the human auditory system, the intended symbolic message constitutes only about 0.1% of the total information content of a speech signal [1].

Now, various spoken language systems attempt to apprehend some of the information in the speech signal, while going to great lengths to ignore the rest. Speaker identification and verification systems are interested in speaker-dependent features, while most other systems implement elaborate schemes to work around the unfortunate presence of this information. There are systems that attempt to gauge the emotional status of the subject, again this is unwanted information in most other systems. Monolingual systems tend to depend on certain speech characteristics that may be absent in other languages.

And so research focused on the various members of spoken language systems family tend to go their own way with the occasional attempt at cross-pollination. Why not integrate all the systems and use the abundance of information to iteratively refine the decisions of various sub-systems? Large vocabulary continuous speech recognition-based ALI (Section 3.7.7) and the work of Li [59], using ideas from speaker identification in ALI, are examples of some attempts in this direction.

5.4.4 System optimisation

5.4.4.1 General comments

A major remaining challenge is to reduce the complexity of ALI systems and to find ways to optimise the large sets of parameters that are associated with existing systems. Our PPRLM system has over 30 parameters and alternative configurations that can potentially affect performance. These parameters are interlinked in a complex, non-linear way and affect system performance synergistically. Finding the parameters that are critical to system performance as well as optimal values for them, is a rather formidable non-linear optimisation problem. Maybe judicious, well-motivated pruning of the parameter space augmented by a genetic algorithm search through the remaining space will be a good angle of attack.

Alternatively, or, in addition, one can attempt to create a model of system behaviour as a function of various parameters and perform the optimisation process on the model, rather than the real system. This would allow for an increase in speed of a couple of orders of magnitude.

5.4.4.2 Phone recognition

A single parameter that may still improve the performance of the PPRLM system significantly when properly optimised, is a weight that represents the transition probability between phones during phone recognition. The parameter will effectively adjust the balance between phone insertions and deletions. We suggest the difference between the number of phones in the phone string returned by the system, and the true number in the test utterance as a simple, yet effective cost function.

5.4.5 Speech corpus management

Typically, spoken language corpora are distributed as a set of audio files in one of many formats and a set of accompanying label files in some custom format. It has occurred to us that research might benefit from an effort to make the data available (if only locally) as a multimedia relational database. Such a database can span different corpora and contain with every utterance various labelling efforts, as well as information regarding the speaker (sex, age, nationality, etc.) and recording conditions.

5.4.6 Discussion

From a state-of-the-art point of view it seems that the basic problems of ALI are now well understood and that relatively successful strategies exist [7, 10]. These will have to be incrementally improved and will be aided by increasing processing power.

5.5 And finally

This then represents our research, findings and views on automatic language identification. In the hope that I have conveyed it clearly and concisely, I thank you for your interest and attention. Live long and prosper.

Appendix A

Data sets

A.1 Introduction

This appendix contains a list of the data files distributed with the OGI_TS corpus that we used. The first column is simply a count of the number of files, the second is the file name of the speech and label files without extensions, the third is the duration of the utterance in seconds and the last column indicates the sex of the speaker. For a few files the sex of the speaker was indeterminate – these were assumed to be female. Please note that some of the files were incompletely labelled and thus provided less data than indicated by the durations given here.

A.2 Training Set

A.2.1 English

1	ENcall-3-G.story-bt	48.8	m
2	ENcall-4-G.story-bt	49.1	m
3	ENcall-5-G.story-bt	48.4	m
4	ENcall-6-G.story-bt	50.0	f
5	ENcall-8-G.story-bt	49.0	m
6	ENcall-9-G.story-bt	48.4	m
7	ENcall-11-G.story-bt	48.8	f
8	ENcall-12-G.story-bt	43.4	f
9	ENcall-13-G.story-bt	49.0	m
10	ENcall-18-G.story-bt	49.1	m
11	ENcall-19-G.story-bt	48.8	f
12	ENcall-20-G.story-bt	47.8	f
13	ENcall-22-G.story-bt	45.1	m
14	ENcall-24-G.story-bt	48.7	m
15	ENcall-27-G.story-bt	42.2	m
16	ENcall-28-G.story-bt	49.4	m
17	ENcall-29-G.story-bt	49.1	m
18	ENcall-30-G.story-bt	50.0	m
19	ENcall-31-G.story-bt	49.2	f
20	ENcall-32-G.story-bt	49.1	f
21	ENcall-33-G.story-bt	48.2	f
22	ENcall-34-G.story-bt	50.0	m
23	ENcall-35-G.story-bt	48.5	m
24	ENcall-37-G.story-bt	48.9	m
25	ENcall-38-G.story-bt	48.3	m
26	ENcall-40-G.story-bt	48.3	m
27	ENcall-41-G.story-bt	48.8	f
28	ENcall-42-G.story-bt	48.9	m
29	ENcall-43-G.story-bt	48.1	m
30	ENcall-44-G.story-bt	48.5	m
31	ENcall-45-G.story-bt	47.9	m
32	ENcall-47-G.story-bt	45.5	f
33	ENcall-48-G.story-bt	47.7	f
34	ENcall-50-G.story-bt	48.5	f
35	ENcall-51-G.story-bt	47.8	f
36	ENcall-52-G.story-bt	49.2	m
37	ENcall-53-G.story-bt	47.8	m
38	ENcall-54-G.story-bt	46.6	m
39	ENcall-56-G.story-bt	49.5	f
40	ENcall-57-G.story-bt	46.8	m
41	ENcall-58-G.story-bt	47.0	m

42	ENcall-59-G.story-bt	49.2	m
43	ENcall-60-G.story-bt	49.4	m
44	ENcall-61-G.story-bt	48.4	m

A.2.2 German

1	GEcall-1-G.story-bt	49.2	f
2	GEcall-2-G.story-bt	48.2	m
3	GEcall-3-G.story-bt	49.0	m
4	GEcall-4-G.story-bt	49.4	f
5	GEcall-5-G.story-bt	48.8	f
6	GEcall-6-G.story-bt	49.3	f
7	GEcall-7-G.story-bt	48.8	m
8	GEcall-9-G.story-bt	45.2	f
9	GEcall-10-G.story-bt	45.3	f
10	GEcall-11-G.story-bt	48.7	f
11	GEcall-12-G.story-bt	49.0	f
12	GEcall-14-G.story-bt	48.9	m
13	GEcall-15-G.story-bt	47.4	f
14	GEcall-16-G.story-bt	46.2	f
15	GEcall-18-G.story-bt	48.4	m
16	GEcall-19-G.story-bt	47.4	f
17	GEcall-22-G.story-bt	47.1	f
18	GEcall-23-G.story-bt	47.0	f
19	GEcall-24-G.story-bt	46.6	m
20	GEcall-26-G.story-bt	48.4	m
21	GEcall-27-G.story-bt	49.8	m
22	GEcall-28-G.story-bt	48.8	m
23	GEcall-31-G.story-bt	48.1	m
24	GEcall-33-G.story-bt	49.4	m
25	GEcall-34-G.story-bt	46.0	m
26	GEcall-36-G.story-bt	47.1	f
27	GEcall-37-G.story-bt	49.4	m
28	GEcall-38-G.story-bt	47.9	m
29	GEcall-39-G.story-bt	48.5	m
30	GEcall-40-G.story-bt	48.9	m
31	GEcall-41-G.story-bt	44.5	m
32	GEcall-42-G.story-bt	48.5	f
33	GEcall-44-G.story-bt	31.9	f
34	GEcall-45-G.story-bt	47.7	f
35	GEcall-46-G.story-bt	48.3	f
36	GEcall-47-G.story-bt	48.6	m
37	GEcall-50-G.story-bt	48.3	f
38	GEcall-51-G.story-bt	46.1	m

39	GEcall-53-G.story-bt	48.9	f
40	GEcall-56-G.story-bt	48.7	f
41	GEcall-57-G.story-bt	44.3	f
42	GEcall-58-G.story-bt	46.6	m
43	GEcall-59-G.story-bt	48.1	f
44	GEcall-60-G.story-bt	48.4	m
45	GEcall-61-G.story-bt	48.7	m
46	GEcall-63-G.story-bt	48.1	f

A.2.3 Japanese

1	JAcall-1-G.story-bt	48.7	m
2	JAcall-2-G.story-bt	48.5	f
3	JAcall-3-G.story-bt	47.0	m
4	JAcall-7-G.story-bt	47.6	m
5	JAcall-13-G.story-bt	48.3	m
6	JAcall-15-G.story-bt	49.6	m
7	JAcall-17-G.story-bt	46.8	f
8	JAcall-19-G.story-bt	48.8	f
9	JAcall-20-G.story-bt	48.4	m
10	JAcall-22-G.story-bt	46.8	m
11	JAcall-23-G.story-bt	48.9	m
12	JAcall-24-G.story-bt	49.2	m
13	JAcall-25-G.story-bt	19.6	f
14	JAcall-27-G.story-bt	48.0	m
15	JAcall-29-G.story-bt	49.1	m
16	JAcall-35-G.story-bt	48.7	f
17	JAcall-36-G.story-bt	48.0	m
18	JAcall-40-G.story-bt	44.0	m
19	JAcall-47-G.story-bt	48.0	f
20	JAcall-48-G.story-bt	49.2	m
21	JAcall-50-G.story-bt	47.5	f
22	JAcall-51-G.story-bt	48.4	m
23	JAcall-53-G.story-bt	47.8	f
24	JAcall-54-G.story-bt	47.3	m
25	JAcall-55-G.story-bt	48.8	f
26	JAcall-57-G.story-bt	48.0	m
27	JAcall-58-G.story-bt	47.1	m
28	JAcall-60-G.story-bt	48.5	f
29	JAcall-61-G.story-bt	46.5	m
30	JAcall-62-G.story-bt	48.0	m
31	JAcall-65-G.story-bt	49.2	m
32	JAcall-66-G.story-bt	48.8	m
33	JAcall-67-G.story-bt	48.5	f

34	JAcall-68-G.story-bt	48.6	f
35	JAcall-69-G.story-bt	23.3	m
36	JAcall-71-G.story-bt	48.4	f
37	JAcall-72-G.story-bt	46.4	f
38	JAcall-73-G.story-bt	47.9	f
39	JAcall-75-G.story-bt	41.4	m
40	JAcall-80-G.story-bt	48.7	m
41	JAcall-82-G.story-bt	47.5	f
42	JAcall-83-G.story-bt	36.1	m
43	JAcall-85-G.story-bt	48.5	f
44	JAcall-86-G.story-bt	48.7	f
45	JAcall-88-G.story-bt	48.9	m
46	JAcall-90-G.story-bt	43.4	f

A.2.4 Mandarin

1	MAcall-1-G.story-bt	36.0	m
2	MAcall-9-G.story-bt	41.1	m
3	MAcall-11-G.story-bt	36.5	m
4	MAcall-13-G.story-bt	48.6	m
5	MAcall-14-G.story-bt	45.4	f
6	MAcall-15-G.story-bt	48.5	m
7	MAcall-16-G.story-bt	47.9	f
8	MAcall-18-G.story-bt	48.3	m
9	MAcall-21-G.story-bt	43.6	m
10	MAcall-23-G.story-bt	29.9	m
11	MAcall-24-G.story-bt	41.1	m
12	MAcall-27-G.story-bt	20.4	m
13	MAcall-30-G.story-bt	46.9	f
14	MAcall-31-G.story-bt	45.9	f
15	MAcall-33-G.story-bt	49.2	f
16	MAcall-34-G.story-bt	3.3	m
17	MAcall-35-G.story-bt	42.7	m
18	MAcall-36-G.story-bt	48.3	m
19	MAcall-37-G.story-bt	47.8	f
20	MAcall-39-G.story-bt	47.1	m
21	MAcall-40-G.story-bt	32.9	u
22	MAcall-41-G.story-bt	48.1	m
23	MAcall-42-G.story-bt	16.1	f
24	MAcall-43-G.story-bt	46.5	m
25	MAcall-44-G.story-bt	46.1	f
26	MAcall-46-G.story-bt	48.5	m
27	MAcall-48-G.story-bt	46.8	m
28	MAcall-49-G.story-bt	49.4	m

29	MAcall-51-G.story-bt	47.6	f
30	MAcall-52-G.story-bt	21.1	m
31	MAcall-53-G.story-bt	15.9	m
32	MAcall-55-G.story-bt	29.5	m
33	MAcall-56-G.story-bt	47.4	m
34	MAcall-57-G.story-bt	43.4	m
35	MAcall-58-G.story-bt	49.0	m
36	MAcall-59-G.story-bt	27.2	f
37	MAcall-60-G.story-bt	10.2	m
38	MAcall-65-G.story-bt	34.6	f
39	MAcall-67-G.story-bt	47.9	f
40	MAcall-68-G.story-bt	48.6	f
41	MAcall-69-G.story-bt	45.7	m
42	MAcall-73-G.story-bt	16.4	f
43	MAcall-76-G.story-bt	46.9	m
44	MAcall-77-G.story-bt	48.2	m
45	MAcall-78-G.story-bt	21.6	m

A.2.5 Spanish

1	SPcall-1-G.story-bt	46.7	m
2	SPcall-2-G.story-bt	48.9	f
3	SPcall-3-G.story-bt	46.9	m
4	SPcall-4-G.story-bt	46.7	m
5	SPcall-5-G.story-bt	48.8	m
6	SPcall-6-G.story-bt	48.6	m
7	SPcall-8-G.story-bt	49.5	m
8	SPcall-10-G.story-bt	49.2	f
9	SPcall-12-G.story-bt	43.3	m
10	SPcall-13-G.story-bt	49.0	m
11	SPcall-14-G.story-bt	48.1	m
12	SPcall-15-G.story-bt	44.4	f
13	SPcall-16-G.story-bt	49.0	f
14	SPcall-17-G.story-bt	47.9	m
15	SPcall-18-G.story-bt	48.2	m
16	SPcall-19-G.story-bt	49.2	m
17	SPcall-20-G.story-bt	45.8	m
18	SPcall-22-G.story-bt	47.9	m
19	SPcall-23-G.story-bt	38.5	m
20	SPcall-24-G.story-bt	26.1	m
21	SPcall-25-G.story-bt	49.3	f
22	SPcall-26-G.story-bt	49.1	m
23	SPcall-27-G.story-bt	46.0	m
24	SPcall-28-G.story-bt	49.4	m

25	SPcall-29-G.story-bt	47.7	m
26	SPcall-30-G.story-bt	48.0	f
27	SPcall-31-G.story-bt	49.0	m
28	SPcall-32-G.story-bt	47.8	f
29	SPcall-33-G.story-bt	48.5	f
30	SPcall-35-G.story-bt	47.6	f
31	SPcall-36-G.story-bt	48.7	m
32	SPcall-37-G.story-bt	48.4	f
33	SPcall-38-G.story-bt	47.8	f
34	SPcall-40-G.story-bt	45.1	f
35	SPcall-41-G.story-bt	49.1	m
36	SPcall-44-G.story-bt	48.4	f
37	SPcall-46-G.story-bt	48.6	m
38	SPcall-47-G.story-bt	48.2	f
39	SPcall-48-G.story-bt	49.4	m
40	SPcall-49-G.story-bt	47.8	m
41	SPcall-50-G.story-bt	48.7	m
42	SPcall-51-G.story-bt	48.2	m
43	SPcall-52-G.story-bt	42.9	f
44	SPcall-53-G.story-bt	47.3	m
45	SPcall-54-G.story-bt	49.3	m
46	SPcall-55-G.story-bt	47.0	m
47	SPcall-56-G.story-bt	48.9	f
48	SPcall-57-G.story-bt	44.5	m

A.3 Development Set

A.3.1 English

1	ENcall-84-G.story-bt	48.5	m
2	ENcall-86-G.story-bt	48.9	m
3	ENcall-87-G.story-bt	47.9	m
4	ENcall-88-G.story-bt	48.5	f
5	ENcall-90-G.story-bt	48.0	f
6	ENcall-92-G.story-bt	49.0	m
7	ENcall-93-G.story-bt	49.0	m
8	ENcall-94-G.story-bt	49.2	m
9	ENcall-96-G.story-bt	48.9	m
10	ENcall-97-G.story-bt	47.5	m
11	ENcall-98-G.story-bt	49.2	m
12	ENcall-99-G.story-bt	49.9	f
13	ENcall-100-G.story-bt	49.1	m
14	ENcall-103-G.story-bt	47.8	f

15	ENcall-105-G.story-bt	46.8	m
16	ENcall-106-G.story-bt	48.6	f
17	ENcall-107-G.story-bt	48.9	m
18	ENcall-108-G.story-bt	48.5	f

A.3.2 German

1	GEcall-95-G.story-bt	49.0	m
2	GEcall-97-G.story-bt	48.0	f
3	GEcall-99-G.story-bt	46.0	m
4	GEcall-100-G.story-bt	48.9	f
5	GEcall-101-G.story-bt	48.8	m
6	GEcall-102-G.story-bt	48.8	m
7	GEcall-106-G.story-bt	48.0	m
8	GEcall-109-G.story-bt	46.9	m
9	GEcall-113-G.story-bt	41.8	f
10	GEcall-114-G.story-bt	49.0	m
11	GEcall-116-G.story-bt	48.0	f
12	GEcall-118-G.story-bt	48.8	f
13	GEcall-120-G.story-bt	47.7	f
14	GEcall-123-G.story-bt	48.7	m
15	GEcall-124-G.story-bt	46.3	f
16	GEcall-125-G.story-bt	46.9	m
17	GEcall-127-G.story-bt	16.0	m
18	GEcall-129-G.story-bt	44.5	f
19	GEcall-130-G.story-bt	49.2	m

A.3.3 Japanese

1	JAcall-100-G.story-bt	48.1	m
2	JAcall-118-G.story-bt	47.6	m
3	JAcall-120-G.story-bt	47.7	m
4	JAcall-121-G.story-bt	48.7	f
5	JAcall-122-G.story-bt	48.9	m
6	JAcall-124-G.story-bt	47.8	f
7	JAcall-126-G.story-bt	47.9	m
8	JAcall-127-G.story-bt	48.1	m
9	JAcall-129-G.story-bt	48.5	m
10	JAcall-131-G.story-bt	48.6	m
11	JAcall-133-G.story-bt	48.5	m
12	JAcall-135-G.story-bt	49.4	m
13	JAcall-136-G.story-bt	46.9	m
14	JAcall-137-G.story-bt	48.2	f

15	JAcall-138-G.story-bt	49.0	m
16	JAcall-139-G.story-bt	48.2	m
17	JAcall-140-G.story-bt	47.1	f
18	JAcall-141-G.story-bt	48.2	m

A.3.4 Mandarin

1	MAcall-79-G.story-bt	48.9	m
2	MAcall-83-G.story-bt	41.1	f
3	MAcall-86-G.story-bt	45.5	f
4	MAcall-90-G.story-bt	48.1	f
5	MAcall-92-G.story-bt	45.6	f
6	MAcall-93-G.story-bt	49.0	m
7	MAcall-97-G.story-bt	48.1	f
8	MAcall-98-G.story-bt	44.9	f
9	MAcall-100-G.story-bt	48.3	f
10	MAcall-101-G.story-bt	45.4	f
11	MAcall-105-G.story-bt	48.5	m
12	MAcall-106-G.story-bt	21.5	m
13	MAcall-107-G.story-bt	44.0	m
14	MAcall-108-G.story-bt	44.9	m
15	MAcall-109-G.story-bt	41.4	m
16	MAcall-113-G.story-bt	48.1	m
17	MAcall-118-G.story-bt	43.7	f
18	MAcall-119-G.story-bt	48.6	m

A.3.5 Spanish

1	SPcall-81-G.story-bt	47.0	m
2	SPcall-82-G.story-bt	47.5	m
3	SPcall-83-G.story-bt	48.0	m
4	SPcall-84-G.story-bt	48.7	m
5	SPcall-85-G.story-bt	40.8	m
6	SPcall-87-G.story-bt	48.4	m
7	SPcall-88-G.story-bt	46.9	m
8	SPcall-89-G.story-bt	45.9	m
9	SPcall-90-G.story-bt	48.0	m
10	SPcall-91-G.story-bt	48.3	m
11	SPcall-93-G.story-bt	15.3	m
12	SPcall-94-G.story-bt	19.1	f
13	SPcall-95-G.story-bt	47.8	m
14	SPcall-96-G.story-bt	48.4	f
15	SPcall-97-G.story-bt	49.0	m

16	SPcall-98-G.story-bt	43.9	m
17	SPcall-99-G.story-bt	45.1	f
18	SPcall-100-G.story-bt	46.5	m
19	SPcall-102-G.story-bt	46.3	f

A.4 Test Set

A.4.1 English

1	ENcall-62-G.story-bt	48.3	m
2	ENcall-63-G.story-bt	48.0	m
3	ENcall-64-G.story-bt	47.2	m
4	ENcall-65-G.story-bt	48.6	m
5	ENcall-66-G.story-bt	49.0	m
6	ENcall-68-G.story-bt	48.9	f
7	ENcall-69-G.story-bt	48.8	m
8	ENcall-70-G.story-bt	48.6	m
9	ENcall-71-G.story-bt	49.4	m
10	ENcall-72-G.story-bt	49.2	m
11	ENcall-73-G.story-bt	48.9	m
12	ENcall-74-G.story-bt	49.2	f
13	ENcall-76-G.story-bt	48.8	m
14	ENcall-77-G.story-bt	49.3	f
15	ENcall-78-G.story-bt	48.6	m
16	ENcall-79-G.story-bt	49.3	m
17	ENcall-81-G.story-bt	49.3	f
18	ENcall-82-G.story-bt	49.2	m
19	ENcall-83-G.story-bt	48.9	m

A.4.2 German

1	GEcall-69-G.story-bt	48.4	f
2	GEcall-70-G.story-bt	48.7	m
3	GEcall-72-G.story-bt	48.6	m
4	GEcall-74-G.story-bt	47.4	m
5	GEcall-75-G.story-bt	48.6	f
6	GEcall-77-G.story-bt	48.3	f
7	GEcall-78-G.story-bt	48.5	m
8	GEcall-79-G.story-bt	22.3	m
9	GEcall-80-G.story-bt	47.9	m
10	GEcall-81-G.story-bt	48.0	m
11	GEcall-83-G.story-bt	47.6	m

12	GEcall-85-G.story-bt	46.7	f
13	GEcall-86-G.story-bt	48.2	m
14	GEcall-87-G.story-bt	47.3	m
15	GEcall-88-G.story-bt	44.3	m
16	GEcall-89-G.story-bt	48.8	m
17	GEcall-90-G.story-bt	48.5	m
18	GEcall-91-G.story-bt	48.4	m
19	GEcall-93-G.story-bt	48.4	m
20	GEcall-94-G.story-bt	47.2	f

A.4.3 Japanese

1	JAcall-26-G.story-bt	48.1	f
2	JAcall-38-G.story-bt	23.1	m
3	JAcall-91-G.story-bt	47.6	f
4	JAcall-92-G.story-bt	47.3	f
5	JAcall-94-G.story-bt	47.1	f
6	JAcall-96-G.story-bt	47.5	m
7	JAcall-97-G.story-bt	47.0	f
8	JAcall-101-G.story-bt	46.3	f
9	JAcall-102-G.story-bt	49.0	m
10	JAcall-104-G.story-bt	42.5	m
11	JAcall-105-G.story-bt	48.6	f
12	JAcall-106-G.story-bt	48.2	m
13	JAcall-107-G.story-bt	48.8	m
14	JAcall-109-G.story-bt	34.0	f
15	JAcall-110-G.story-bt	48.5	f
16	JAcall-112-G.story-bt	44.6	m
17	JAcall-113-G.story-bt	48.6	m
18	JAcall-116-G.story-bt	49.0	m
19	JAcall-117-G.story-bt	44.4	m
20	JAcall-142-G.story-bt	47.5	m

A.4.4 Mandarin

1	MAcall-121-G.story-bt	48.6	f
2	MAcall-122-G.story-bt	46.2	m
3	MAcall-123-G.story-bt	48.9	f
4	MAcall-124-G.story-bt	47.8	f
5	MAcall-126-G.story-bt	36.7	m
6	MAcall-127-G.story-bt	31.4	m
7	MAcall-129-G.story-bt	49.6	f
8	MAcall-134-G.story-bt	45.8	m
9	MAcall-135-G.story-bt	48.2	m
10	MAcall-136-G.story-bt	47.4	m
11	MAcall-137-G.story-bt	48.0	f
12	MAcall-138-G.story-bt	48.2	m
13	MAcall-140-G.story-bt	43.2	m
14	MAcall-142-G.story-bt	48.2	m
15	MAcall-143-G.story-bt	47.0	f
16	MAcall-146-G.story-bt	45.6	m
17	MAcall-147-G.story-bt	49.1	m
18	MAcall-148-G.story-bt	44.7	m
19	MAcall-149-G.story-bt	48.1	m

A.4.5 Spanish

1	SPcall-60-G.story-bt	41.3	m
2	SPcall-62-G.story-bt	49.3	m
3	SPcall-63-G.story-bt	48.1	m
4	SPcall-64-G.story-bt	47.6	f
5	SPcall-65-G.story-bt	49.1	m
6	SPcall-67-G.story-bt	49.3	m
7	SPcall-68-G.story-bt	49.0	m
8	SPcall-69-G.story-bt	44.7	f
9	SPcall-70-G.story-bt	47.6	m
10	SPcall-71-G.story-bt	36.6	f
11	SPcall-72-G.story-bt	48.9	f
12	SPcall-73-G.story-bt	48.0	m
13	SPcall-76-G.story-bt	46.0	m
14	SPcall-77-G.story-bt	49.0	f
15	SPcall-78-G.story-bt	49.1	m
16	SPcall-79-G.story-bt	45.4	f
17	SPcall-80-G.story-bt	49.3	f

Appendix B

Results

B.1 Introduction

This appendix contains graphs of some of the results of Chapter 4 on a per language-pair basis.

B.2 PPRLM system, minimum distinctiveness measure value, training set

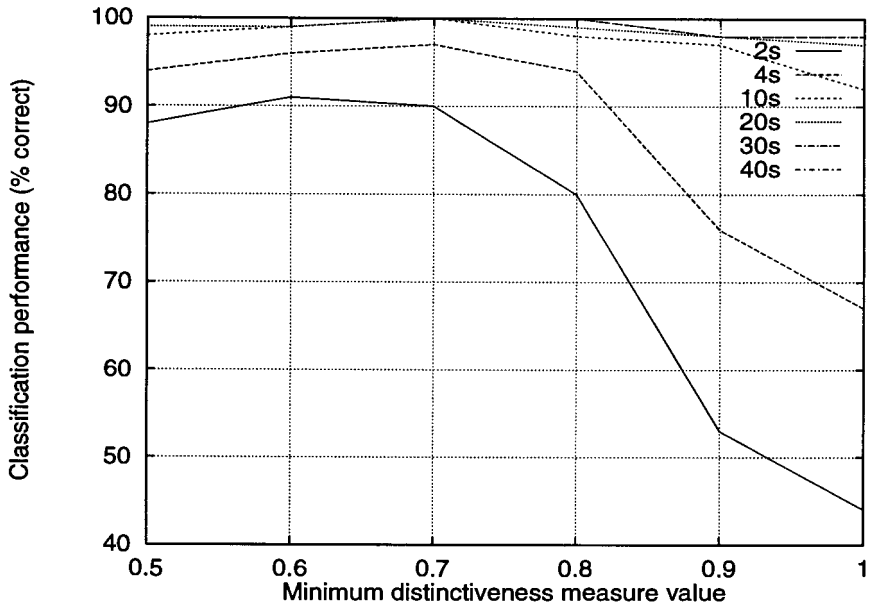

Figure B.1: English - German language classification performance on the training set as a function of the minimum distinctiveness measure value and utterance duration.

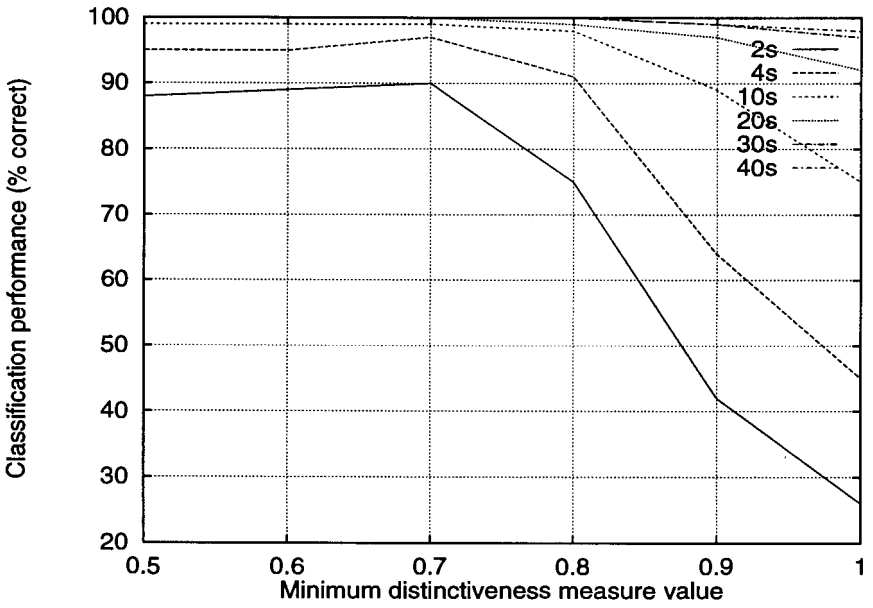

Figure B.2: English - Japanese language classification performance on the training set as a function of the minimum distinctiveness measure value and utterance duration.

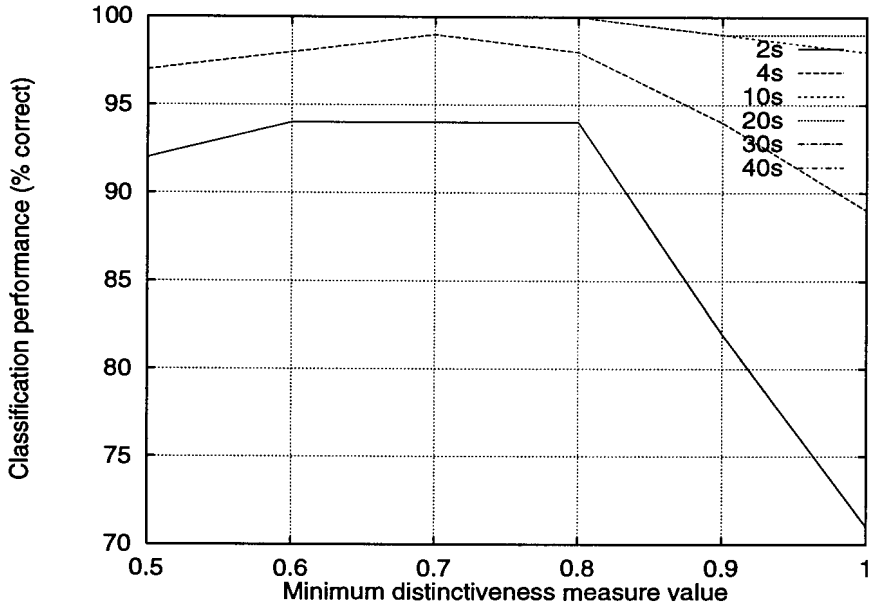

Figure B.3: English - Mandarin language classification performance on the training set as a function of the minimum distinctiveness measure value and utterance duration.

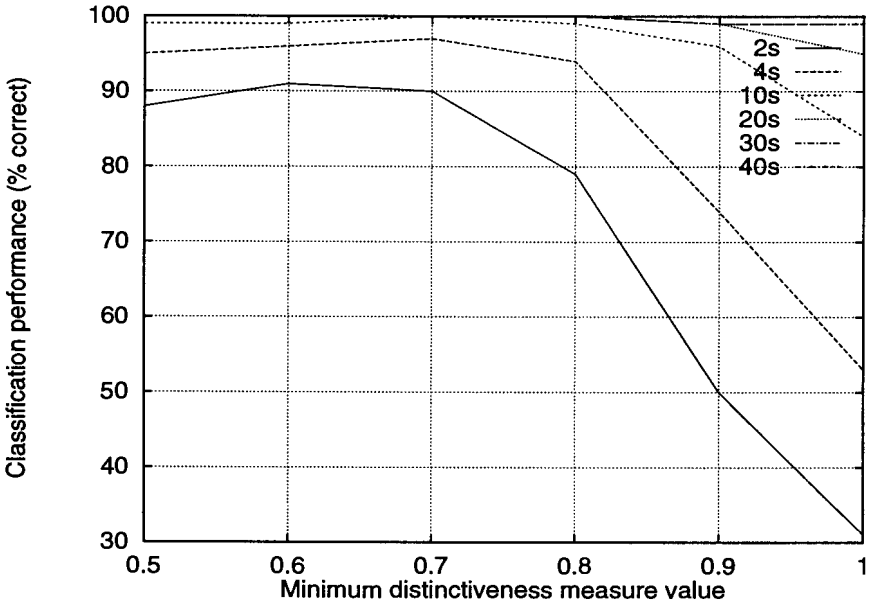

Figure B.4: English - Spanish language classification performance on the training set as a function of the minimum distinctiveness measure value and utterance duration.

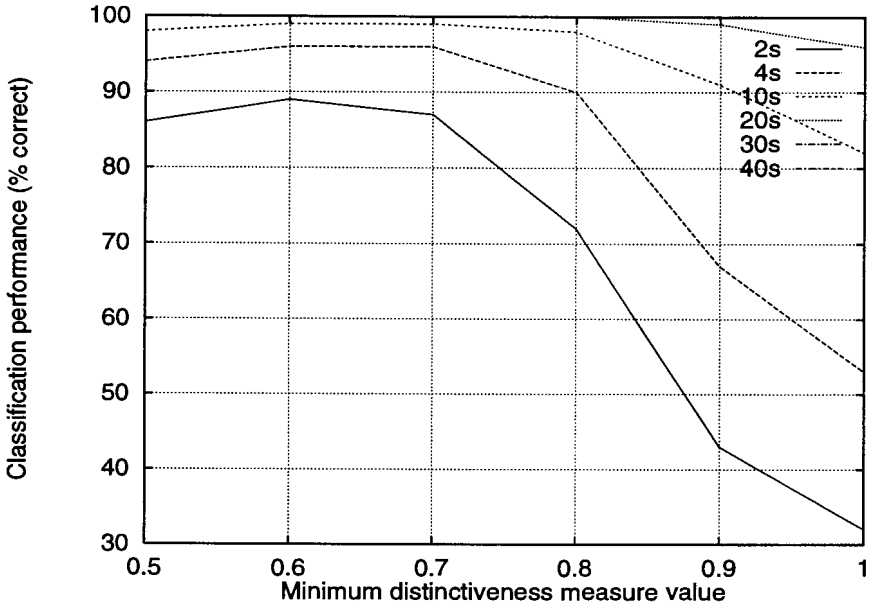

Figure B.5: German - Japanese language classification performance on the training set as a function of the minimum distinctiveness measure value and utterance duration.

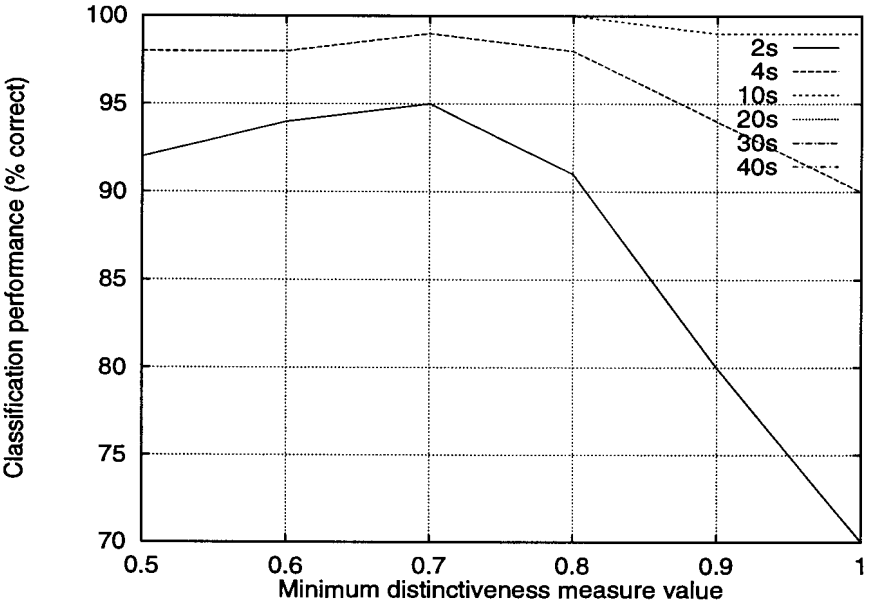

Figure B.6: German - Mandarin language classification performance on the training set as a function of the minimum distinctiveness measure value and utterance duration.

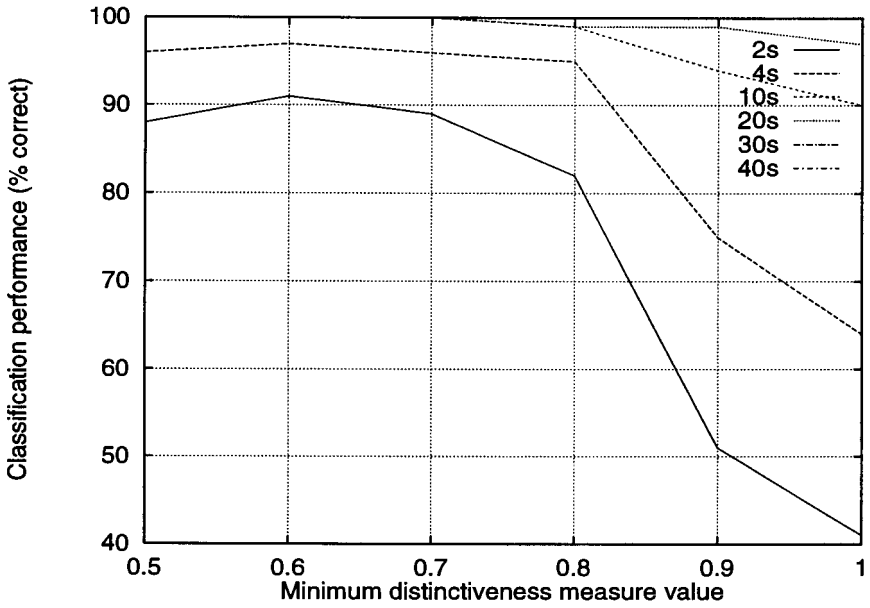

Figure B.7: German - Spanish language classification performance on the training set as a function of the minimum distinctiveness measure value and utterance duration.

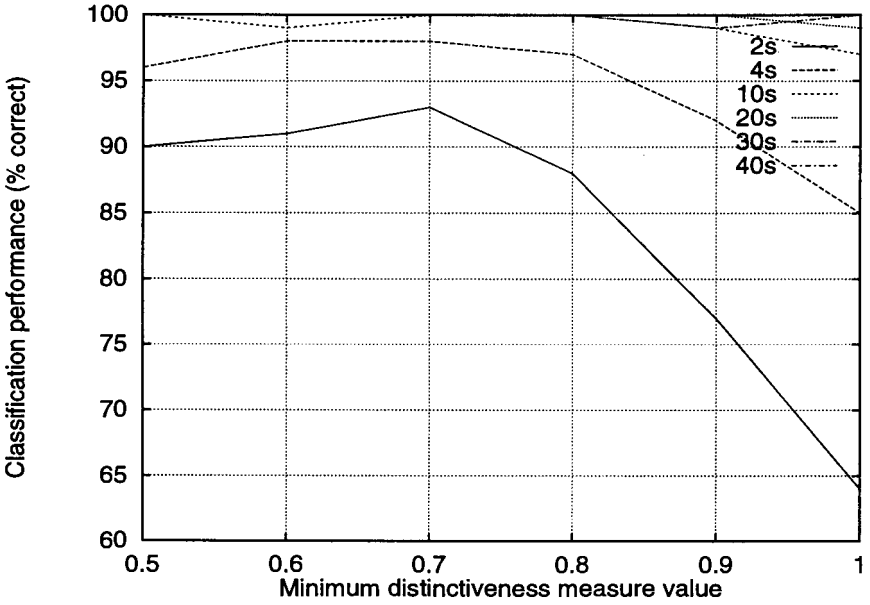

Figure B.8: Japanese - Mandarin language classification performance on the training set as a function of the minimum distinctiveness measure value and utterance duration.

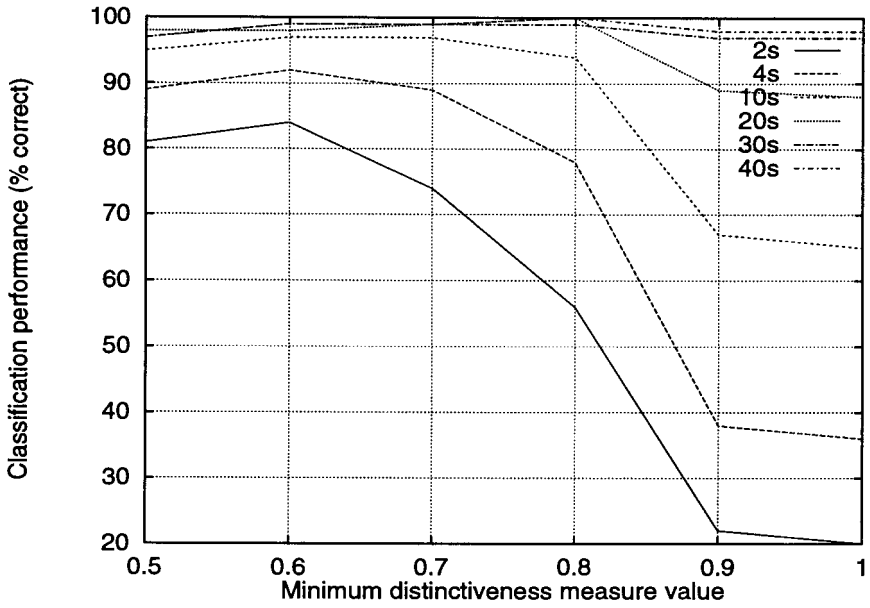

Figure B.9: Japanese - Spanish language classification performance on the training set as a function of the minimum distinctiveness measure value and utterance duration.

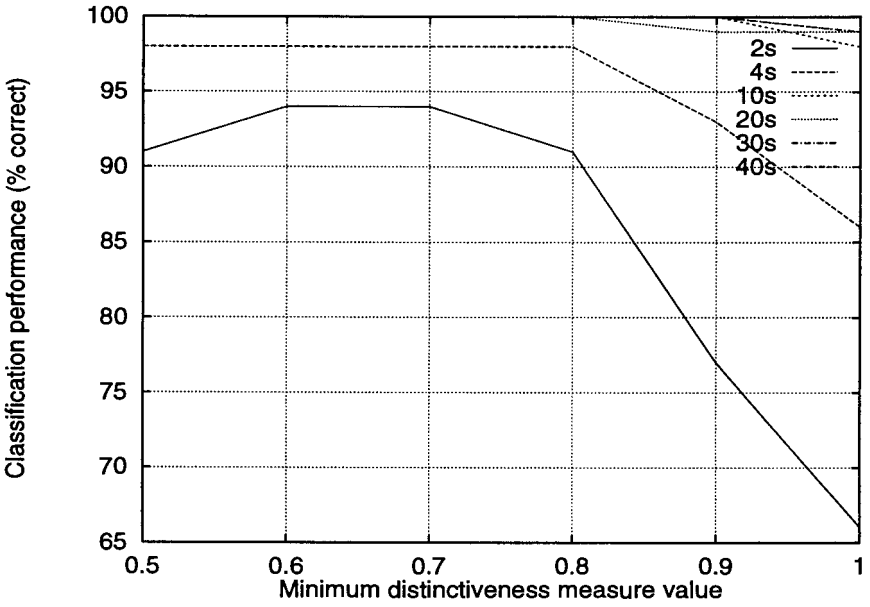

Figure B.10: Mandarin - Spanish language classification performance on the training set as a function of the minimum distinctiveness measure value and utterance duration.

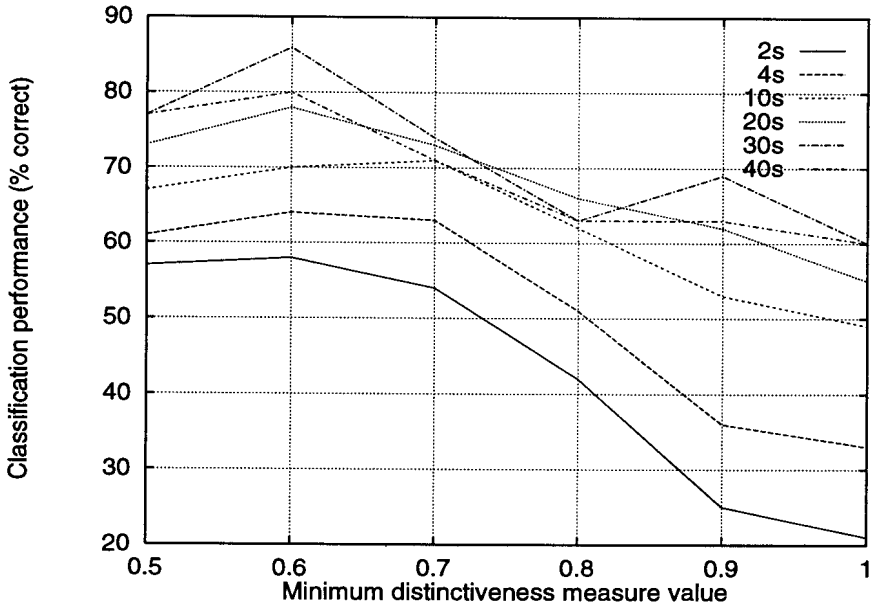

Figure B.11: English - German language classification performance on development set as a function of minimum distinctiveness measure value and utterance duration.

B.3 PPRLM system, minimum distinctiveness measure value, development set

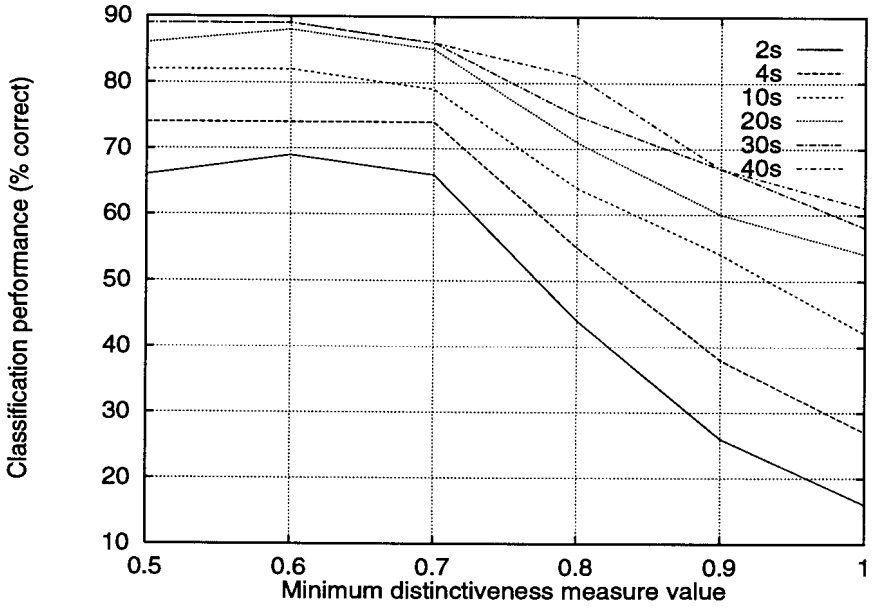

Figure B.12: English - Japanese language classification performance on development set as a function of minimum distinctiveness measure value and utterance duration.

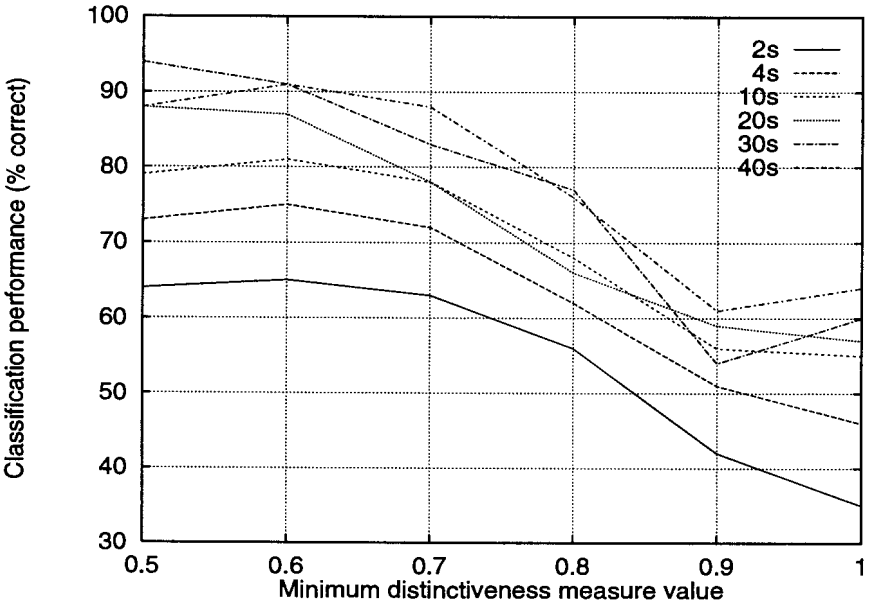

Figure B.13: English - Mandarin language classification performance on development set as a function of minimum distinctiveness measure value and utterance duration.

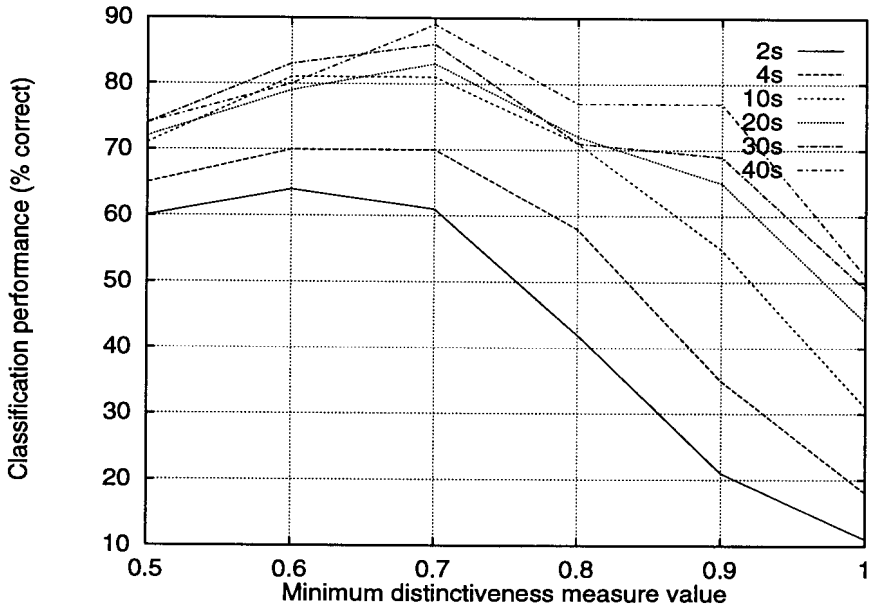

Figure B.14: English - Spanish language classification performance on development set as a function of minimum distinctiveness measure value and utterance duration.

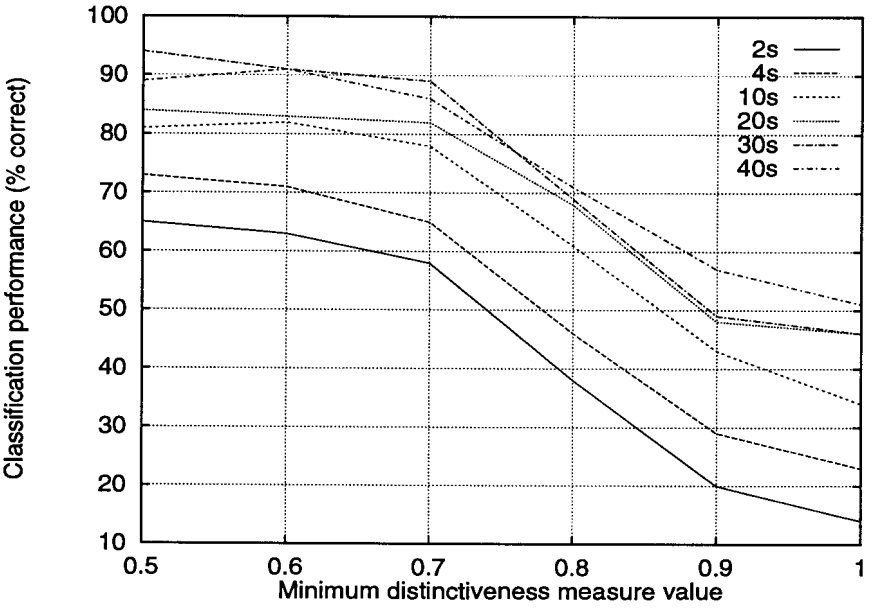

Figure B.15: German - Japanese language classification performance on development set as a function of minimum distinctiveness measure value and utterance duration.

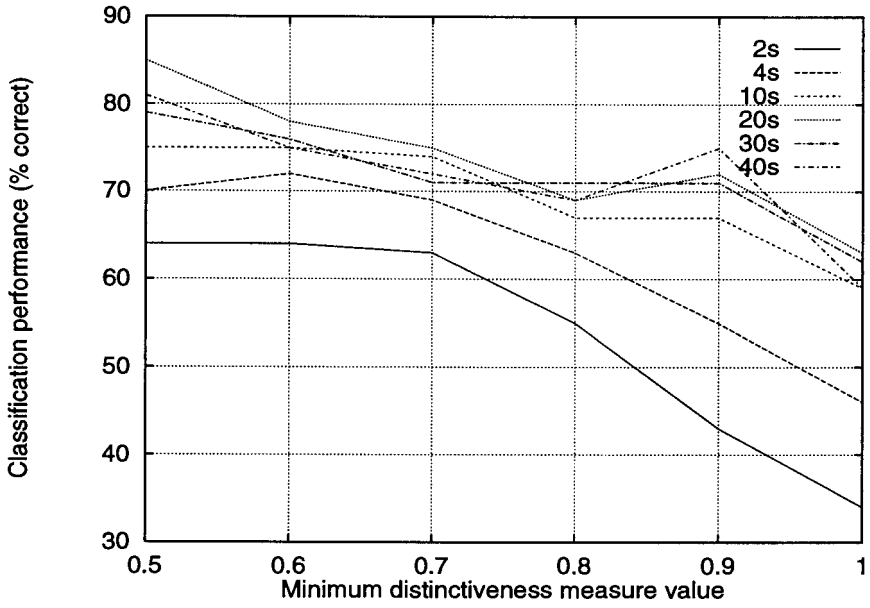

Figure B.16: German - Mandarin language classification performance on development set as a function of minimum distinctiveness measure value and utterance duration.

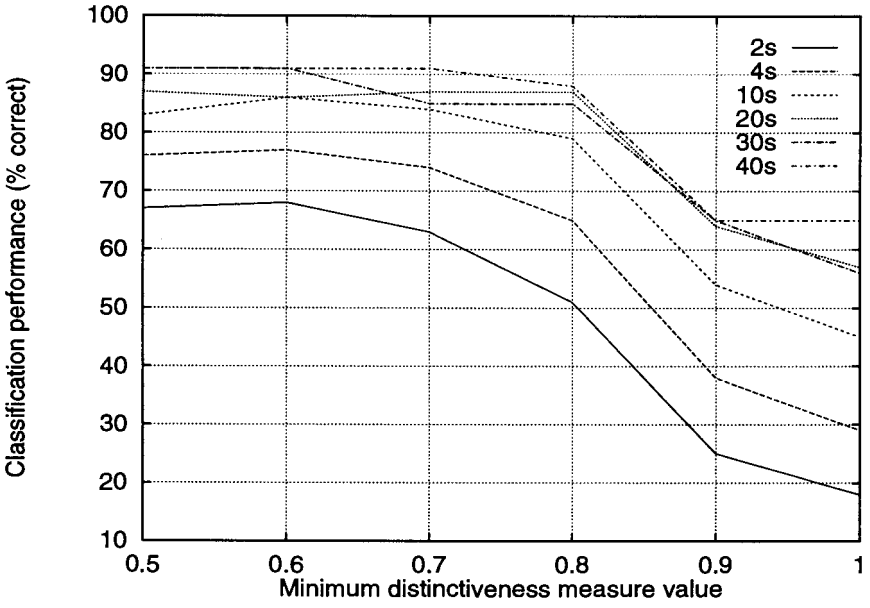

Figure B.17: German - Spanish language classification performance on development set as a function of minimum distinctiveness measure value and utterance duration.

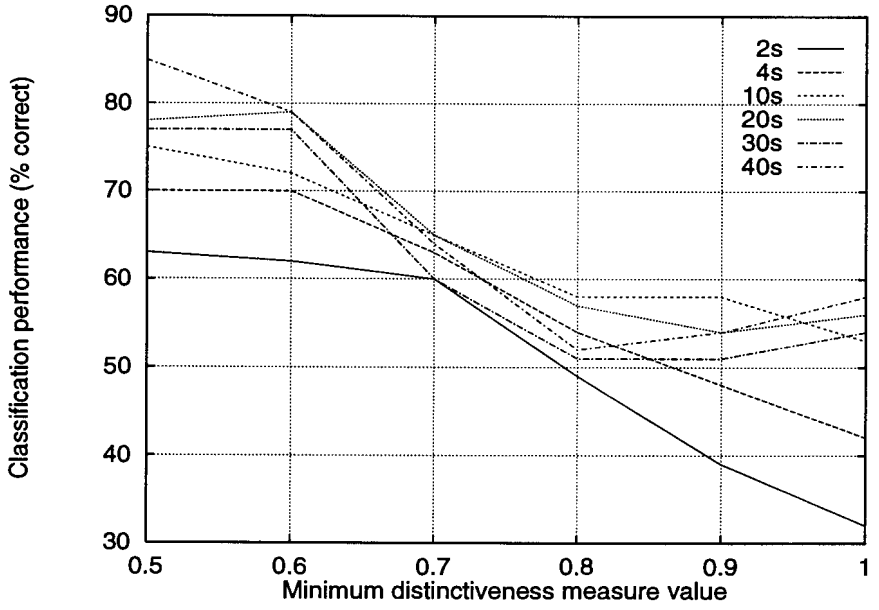

Figure B.18: Japanese - Mandarin language classification performance on development set as a function of minimum distinctiveness measure value and utterance duration.

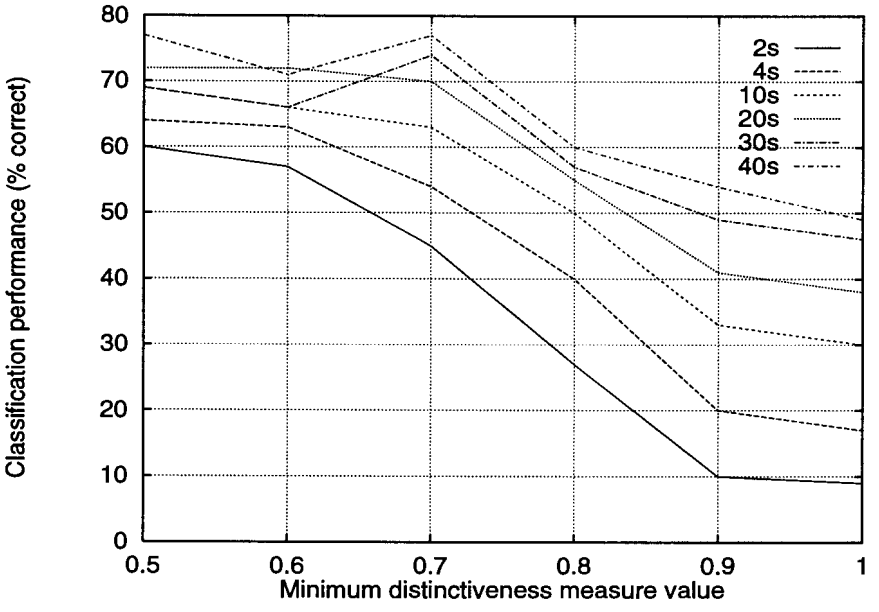

Figure B.19: Japanese - Spanish language classification performance on development set as a function of minimum distinctiveness measure value and utterance duration.

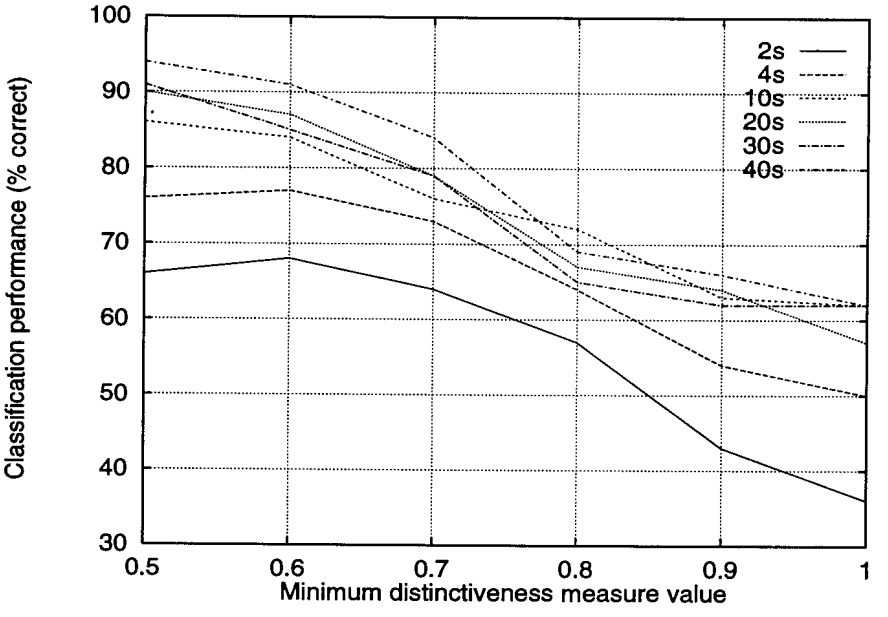

Figure B.20: Mandarin - Spanish language classification performance on development set as a function of minimum distinctiveness measure value and utterance duration.

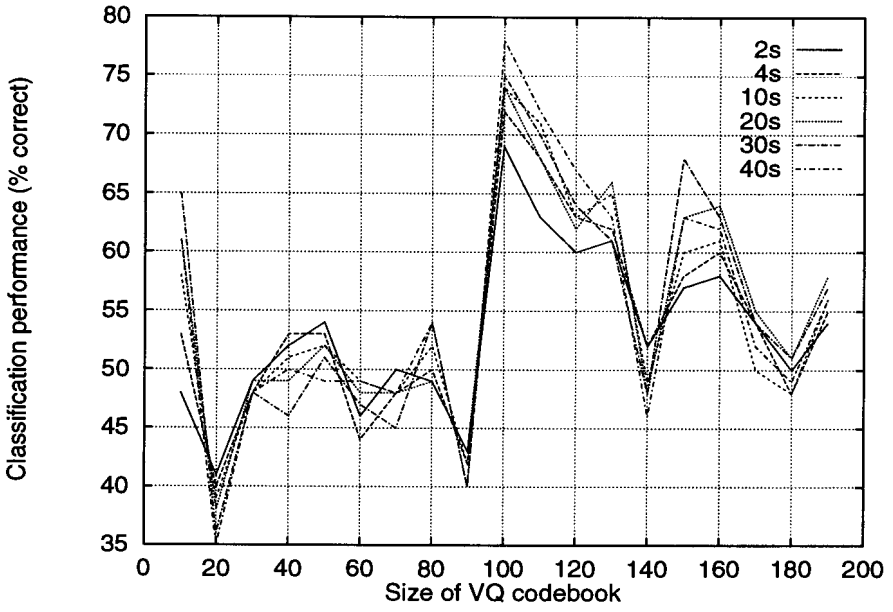

Figure B.21: English - German language classification performance on the training set as a function of VQ codebook size and utterance duration.

B.4 VQLM system, VQ codebook size, train set

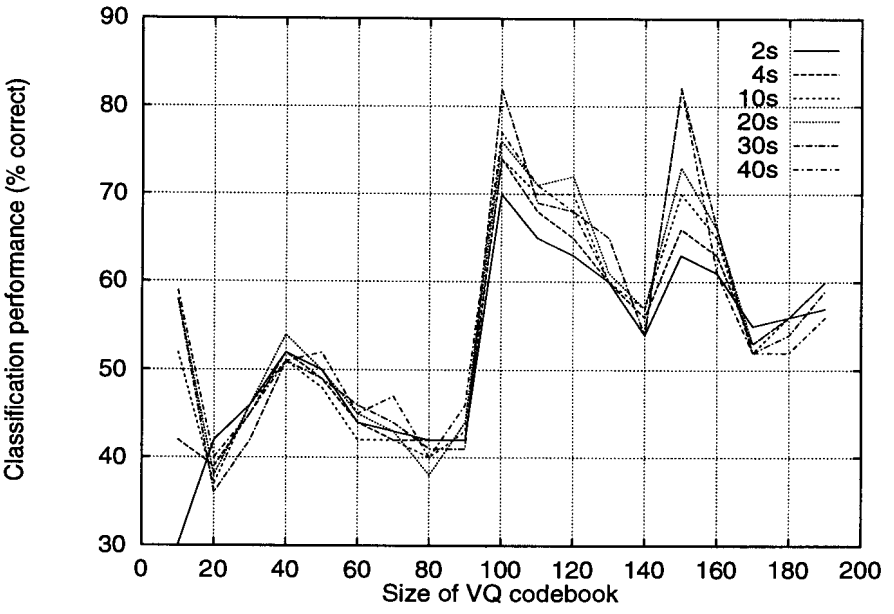

Figure B.22: English - Japanese language classification performance on the training set as a function of VQ codebook size and utterance duration.

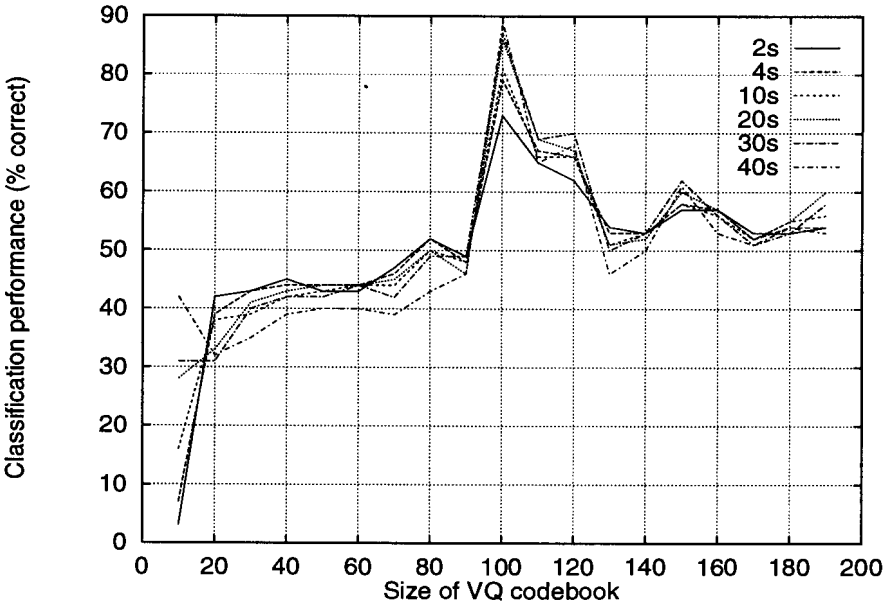

Figure B.23: English - Mandarin language classification performance on the training set as a function of VQ codebook size and utterance duration.

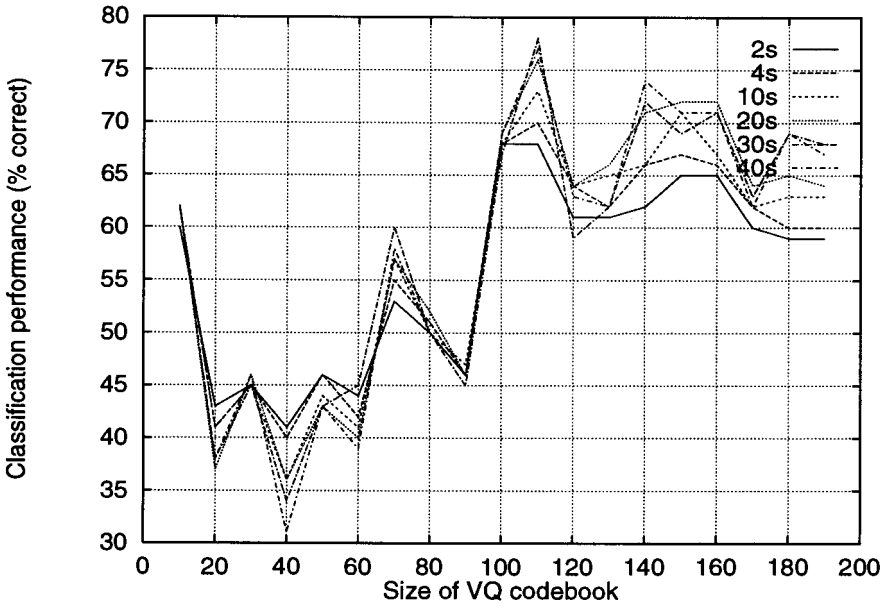

Figure B.24: English - Spanish language classification performance on the training set as a function of VQ codebook size and utterance duration.

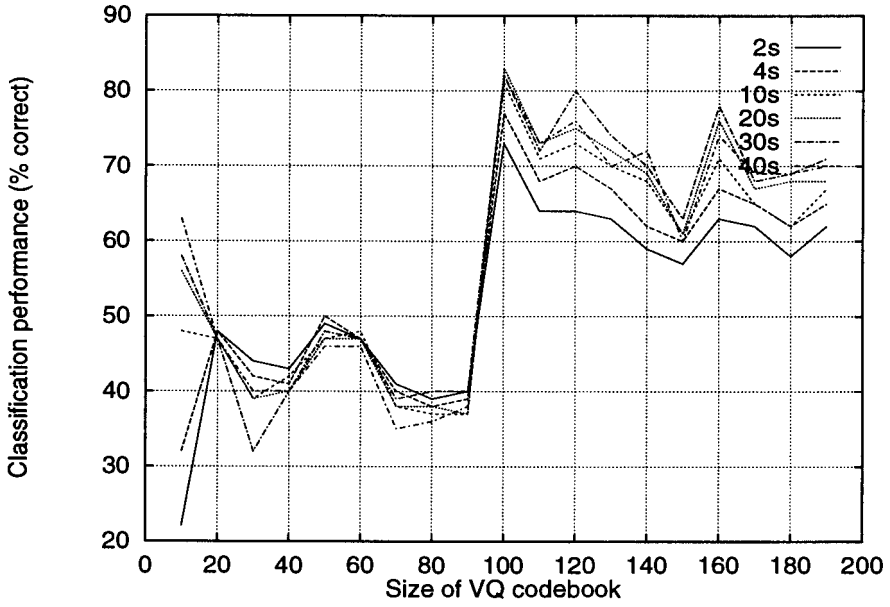

Figure B.25: German - Japanese language classification performance on the training set as a function of VQ codebook size and utterance duration.

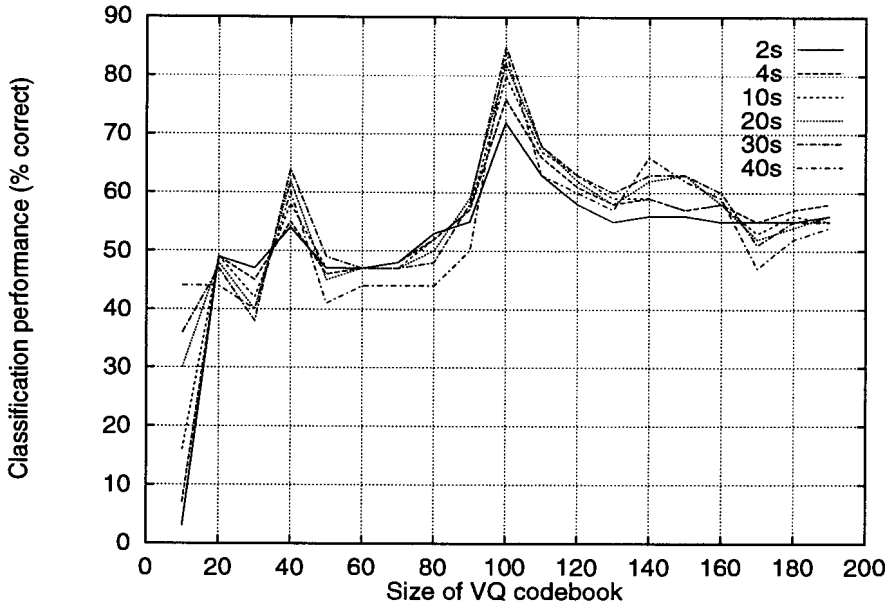

Figure B.26: German - Mandarin language classification performance on the training set as a function of VQ codebook size and utterance duration.

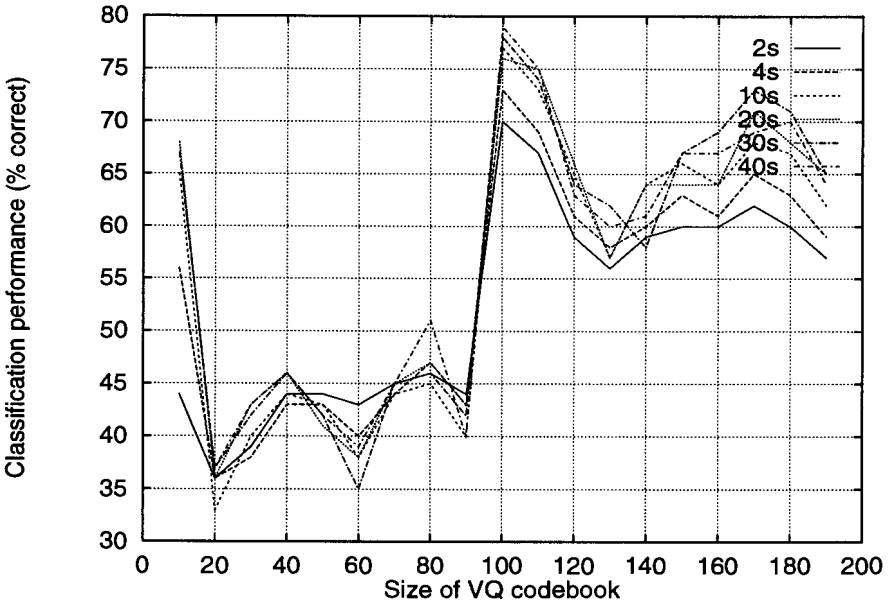

Figure B.27: German - Spanish language classification performance on the training set as a function of VQ codebook size and utterance duration.

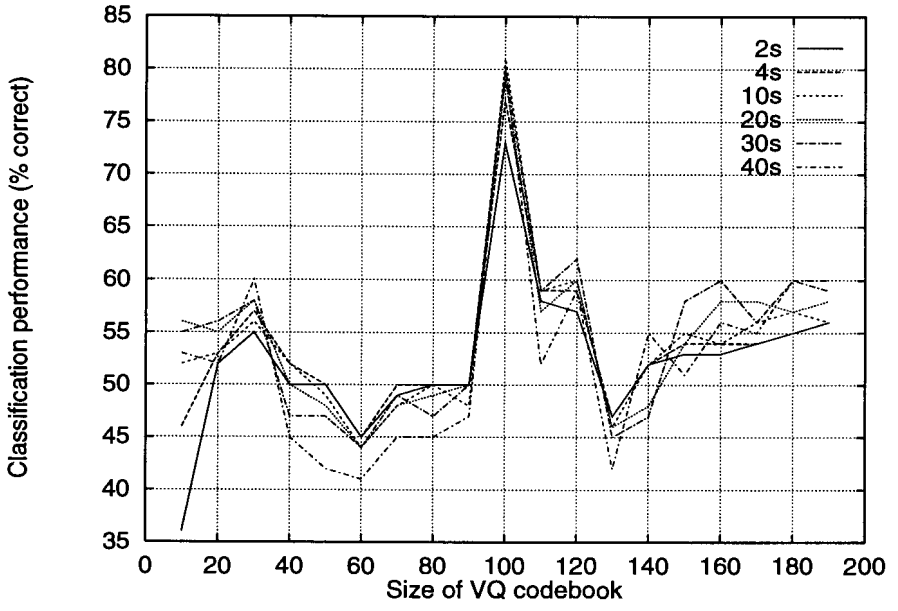

Figure B.28: Japanese - Mandarin language classification performance on the training set as a function of VQ codebook size and utterance duration.

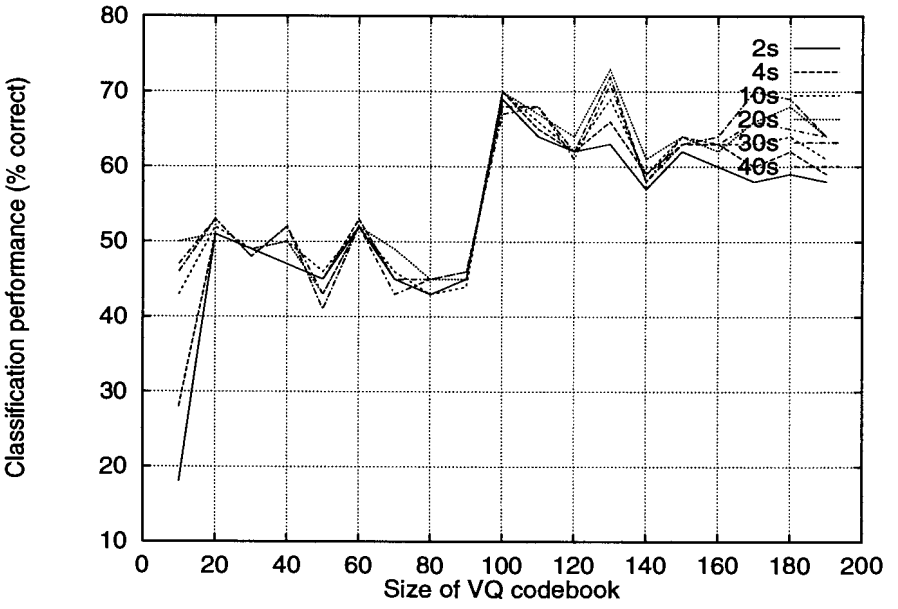

Figure B.29: Japanese - Spanish language classification performance on the training set as a function of VQ codebook size and utterance duration.

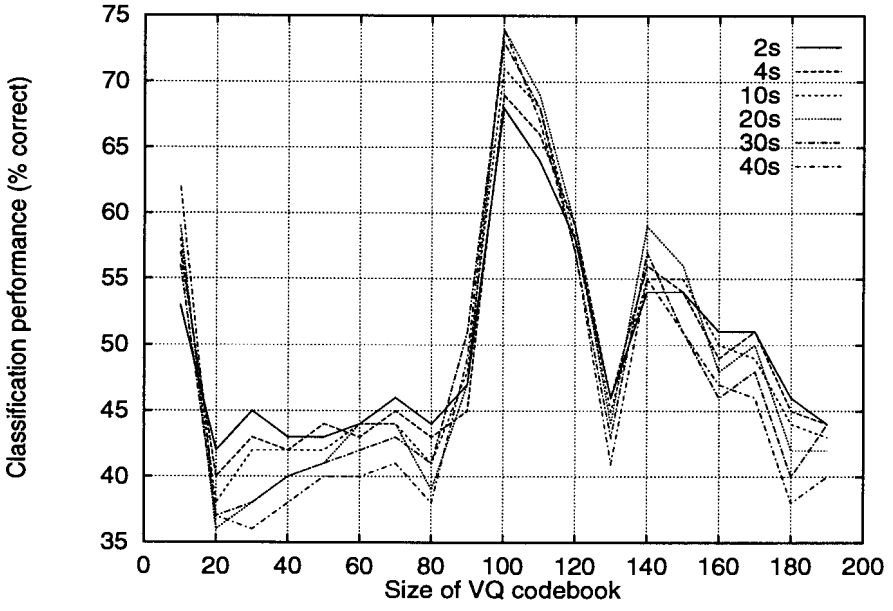

Figure B.30: Mandarin - Spanish language classification performance on the training set as a function of VQ codebook size and utterance duration.

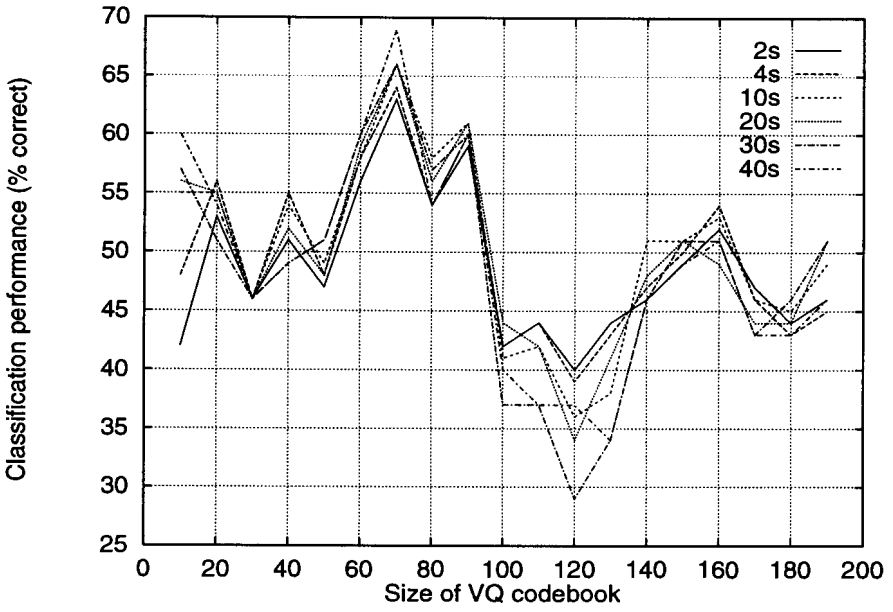

Figure B.31: English - German language classification performance on the development set as a function of VQ codebook size and utterance duration.

B.5 VQLM system, VQ codebook size, development set

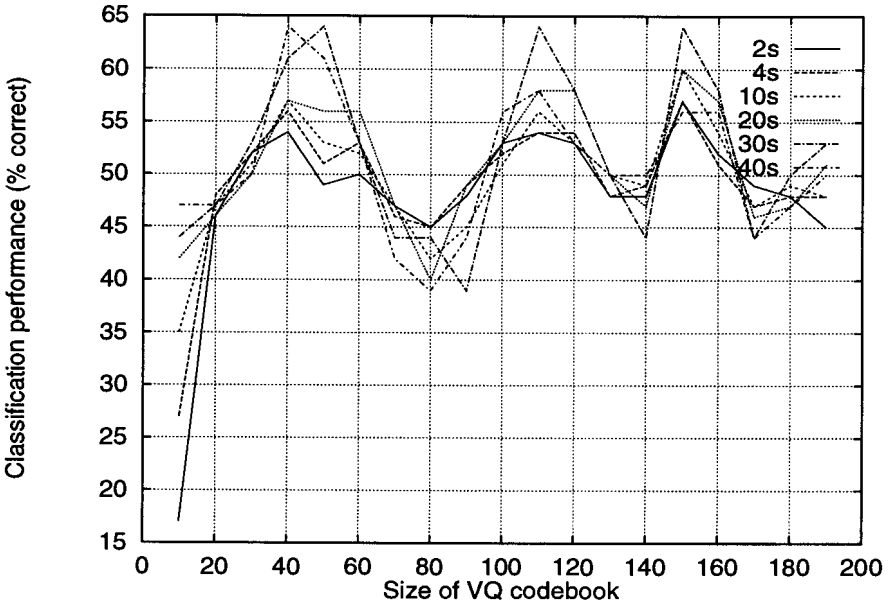

Figure B.32: English - Japanese language classification performance on the development set as a function of VQ codebook size and utterance duration.

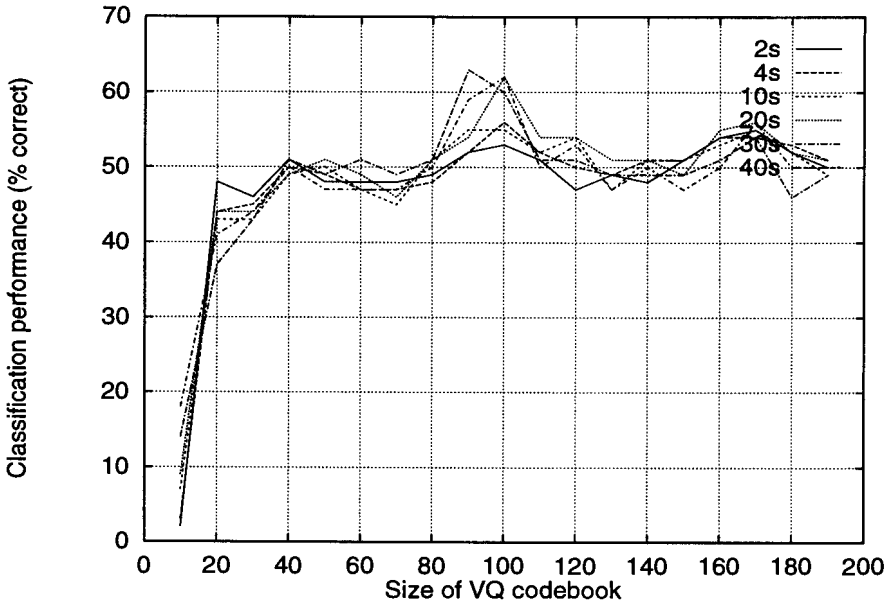

Figure B.33: English - Mandarin language classification performance on the development set as a function of VQ codebook size and utterance duration.

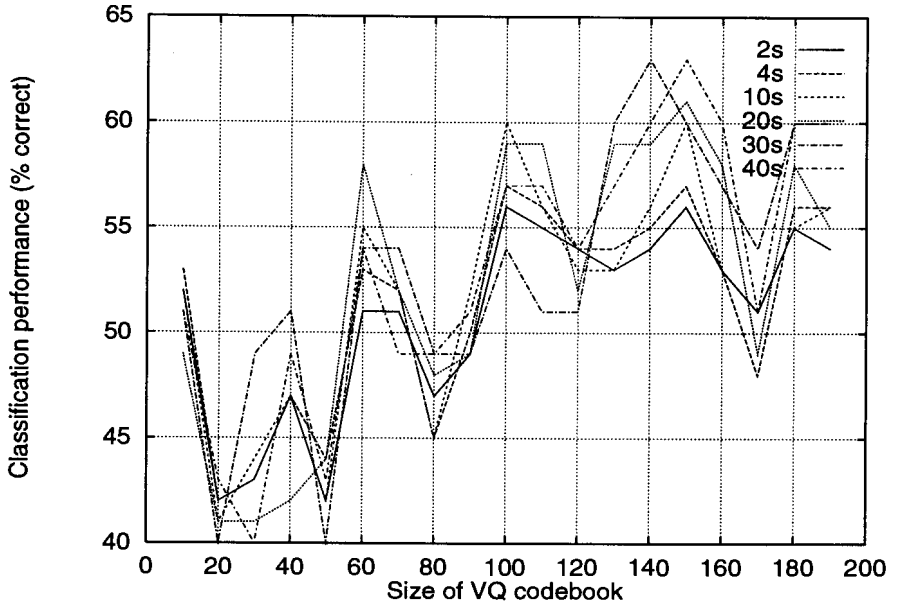

Figure B.34: English - Spanish language classification performance on the development set as a function of VQ codebook size and utterance duration.

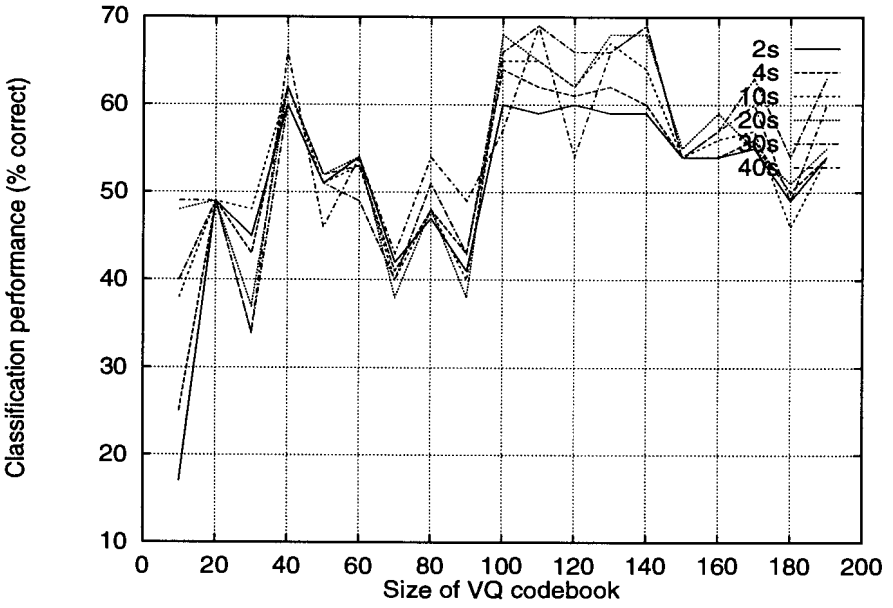

Figure B.35: German - Japanese language classification performance on the development set as a function of VQ codebook size and utterance duration.

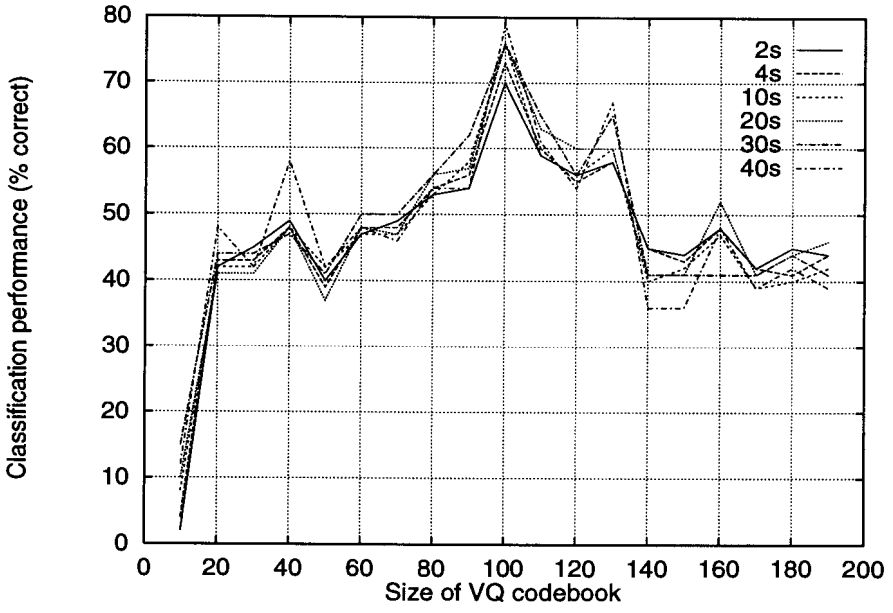

Figure B.36: German - Mandarin language classification performance on the development set as a function of VQ codebook size and utterance duration.

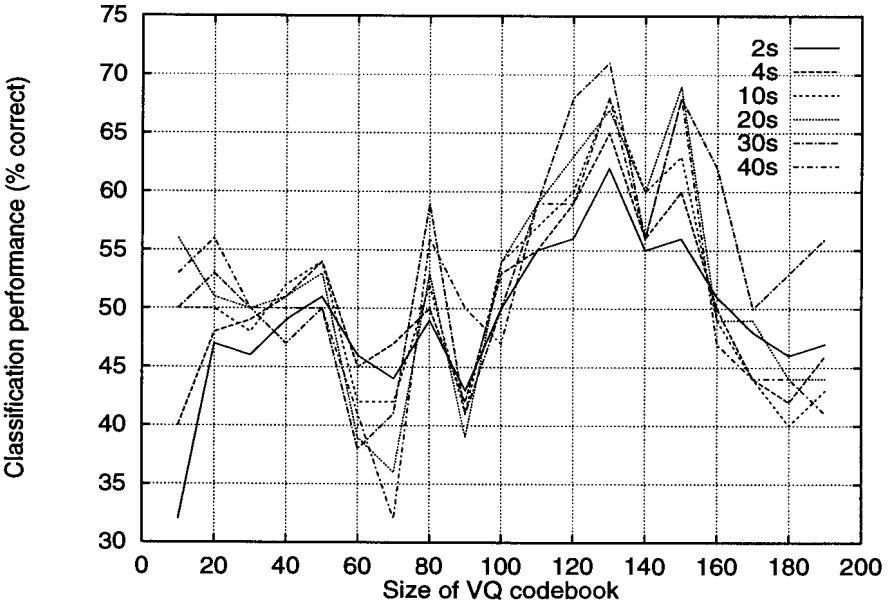

Figure B.37: German - Spanish language classification performance on the development set as a function of VQ codebook size and utterance duration.

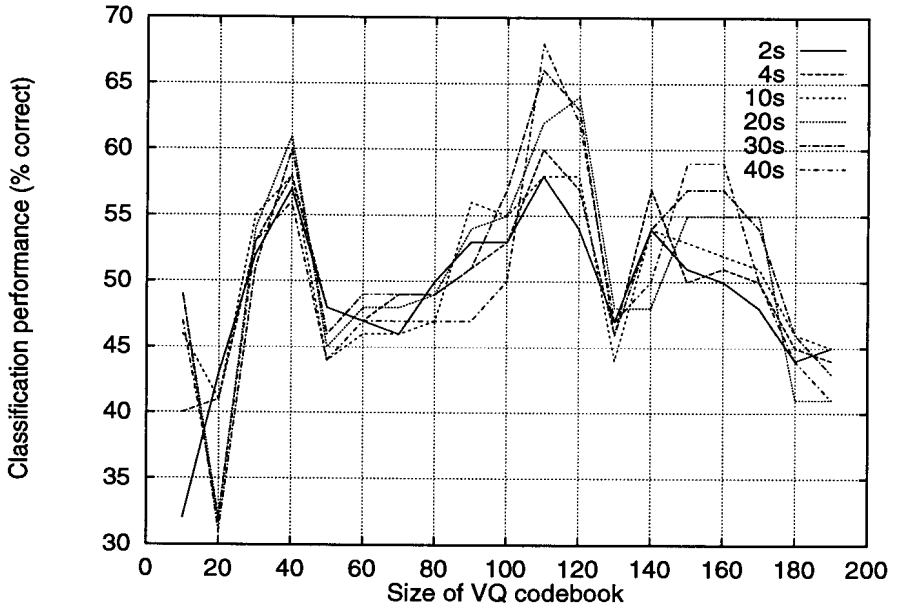

Figure B.38: Japanese - Mandarin language classification performance on the development set as a function of VQ codebook size and utterance duration.

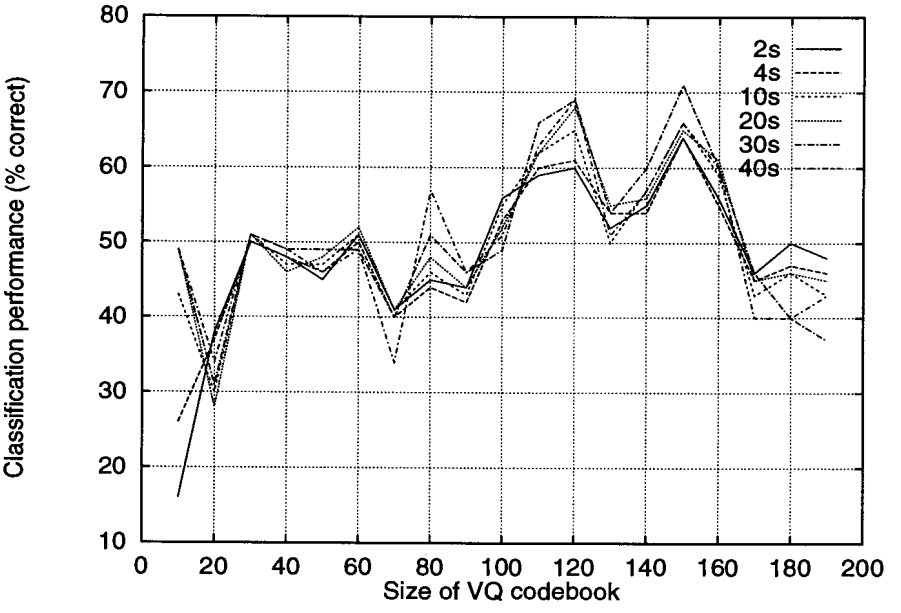

Figure B.39: Japanese - Spanish language classification performance on the development set as a function of VQ codebook size and utterance duration.

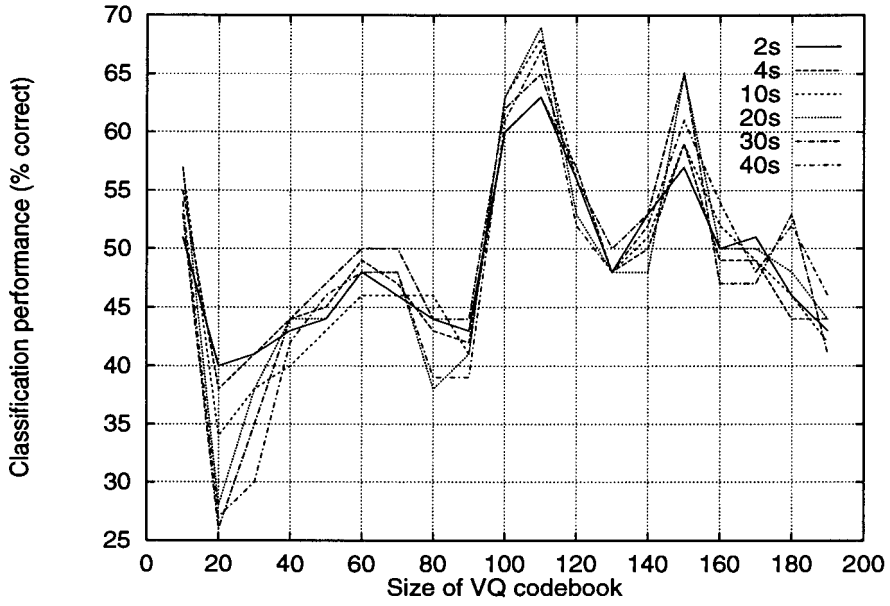

Figure B.40: Mandarin - Spanish language classification performance on the development set as a function of VQ codebook size and utterance duration.

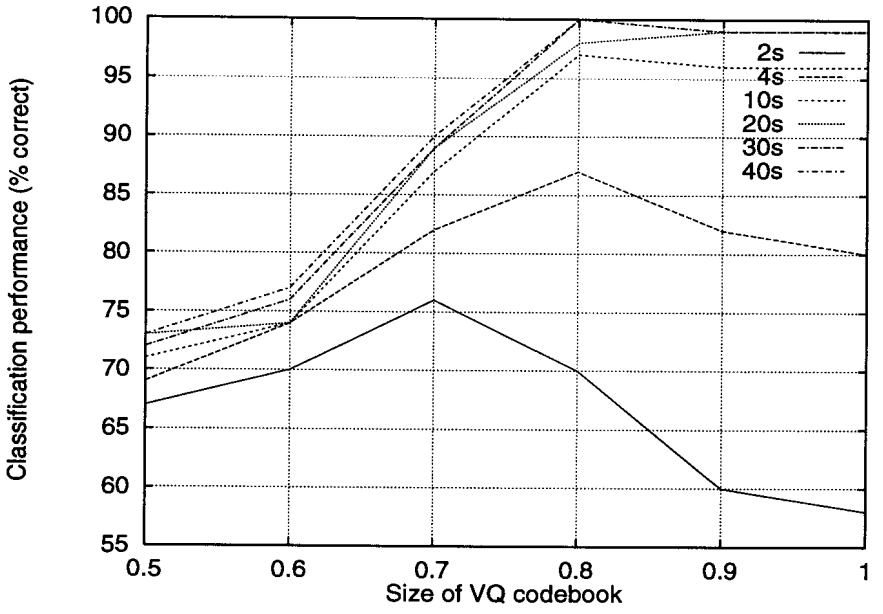

Figure B.41: English - German language classification performance on the training set as a function of the minimum distinctiveness measure value and utterance duration.

B.6 VQLM system, minimum distinctiveness measure value, train set

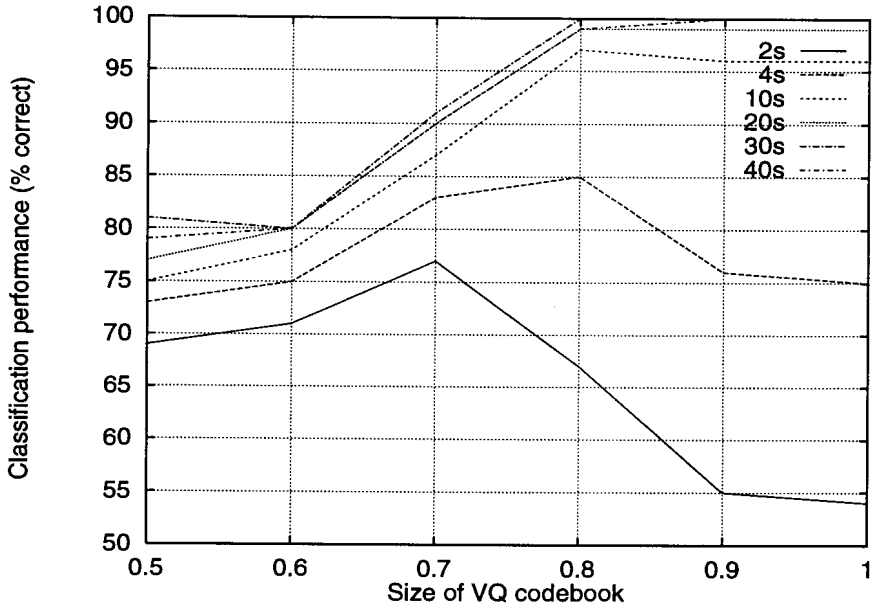

Figure B.42: English - Japanese language classification performance on the training set as a function of the minimum distinctiveness measure value and utterance duration.

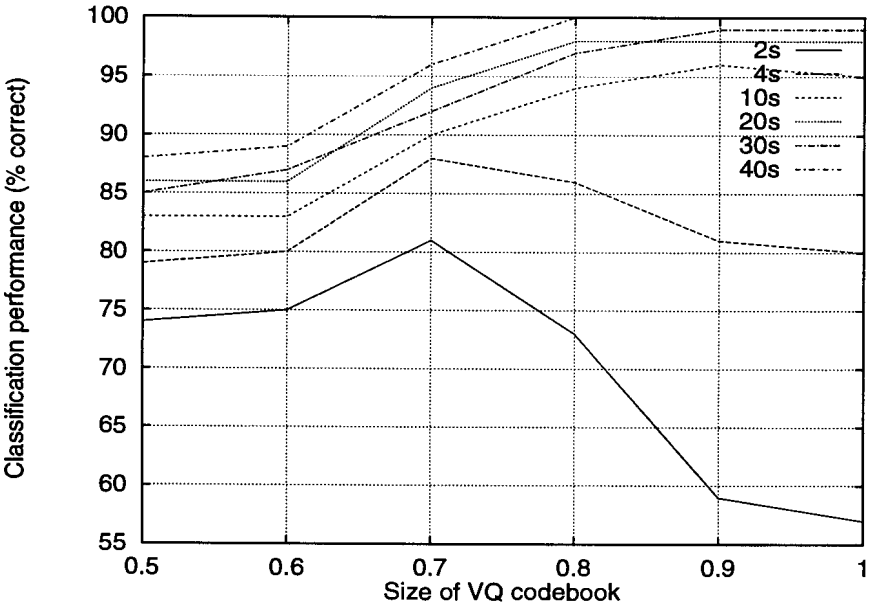

Figure B.43: English - Mandarin language classification performance on the training set as a function of the minimum distinctiveness measure value and utterance duration.

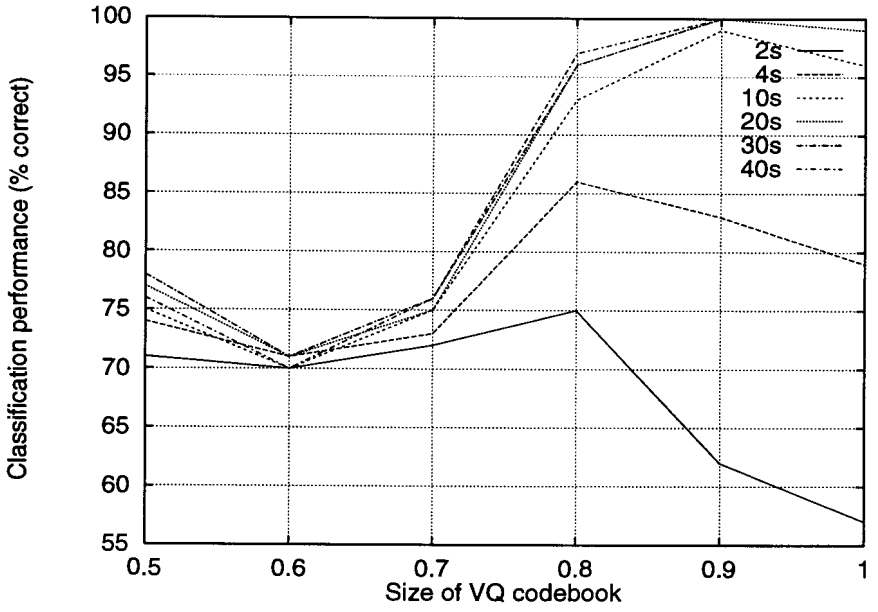

Figure B.44: English - Spanish language classification performance on the training set as a function of the minimum distinctiveness measure value and utterance duration.

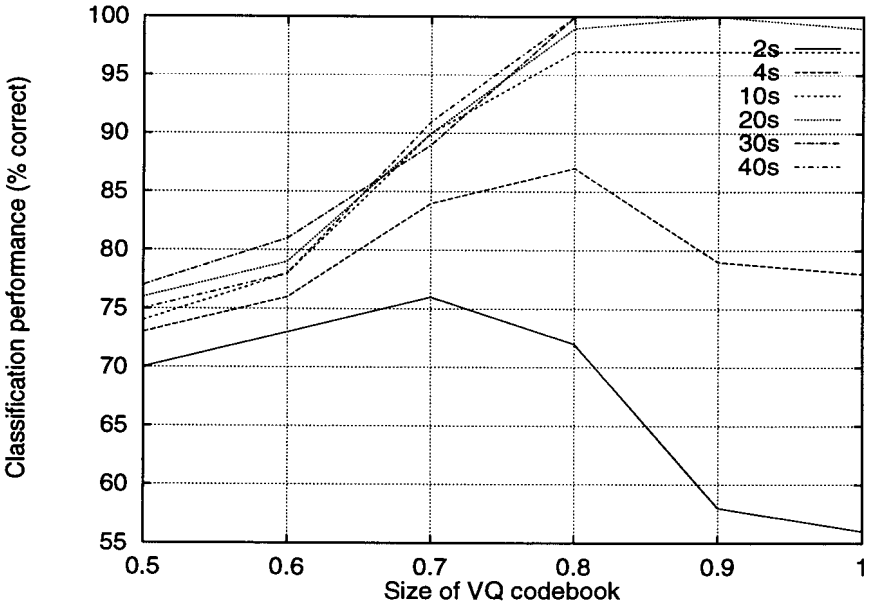

Figure B.45: German - Japanese language classification performance on the training set as a function of the minimum distinctiveness measure value and utterance duration.

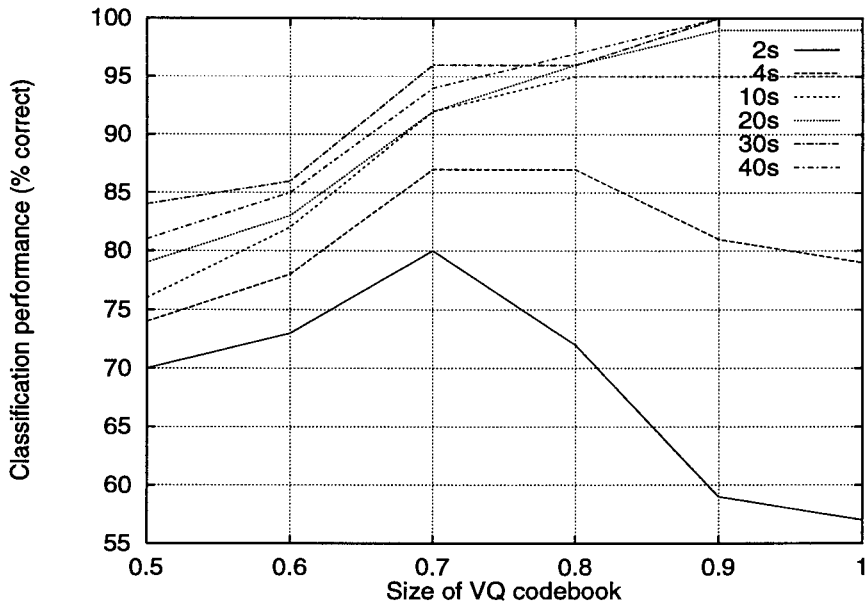

Figure B.46: German - Mandarin language classification performance on the training set as a function of the minimum distinctiveness measure value and utterance duration.

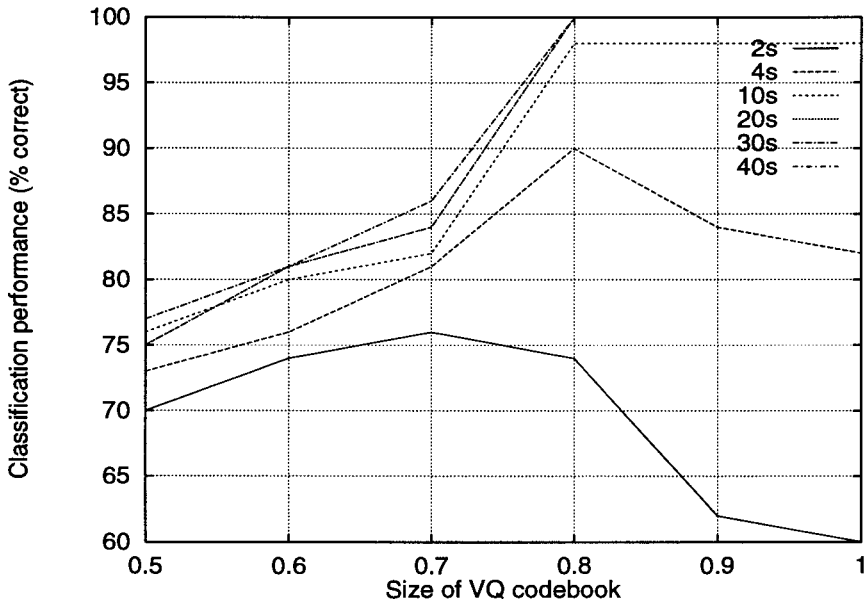

Figure B.47: German - Spanish language classification performance on the training set as a function of the minimum distinctiveness measure value and utterance duration.

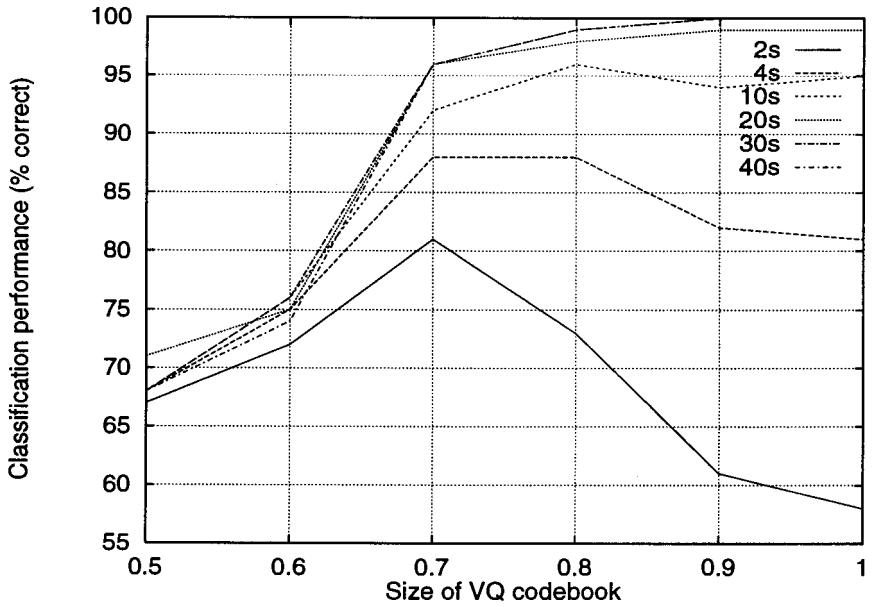

Figure B.48: Japanese - Mandarin language classification performance on the training set as a function of the minimum distinctiveness measure value and utterance duration.

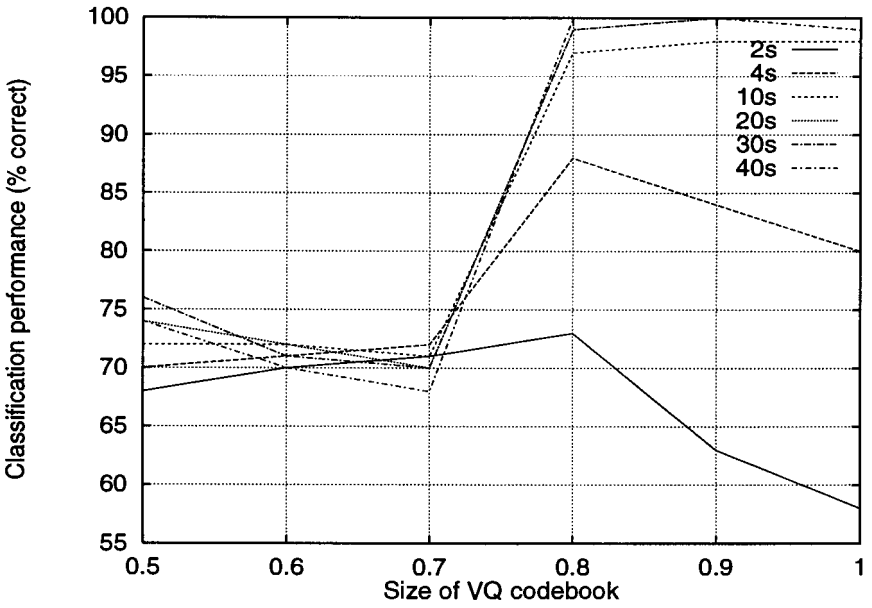

Figure B.49: Japanese - Spanish language classification performance on the training set as a function of the minimum distinctiveness measure value and utterance duration.

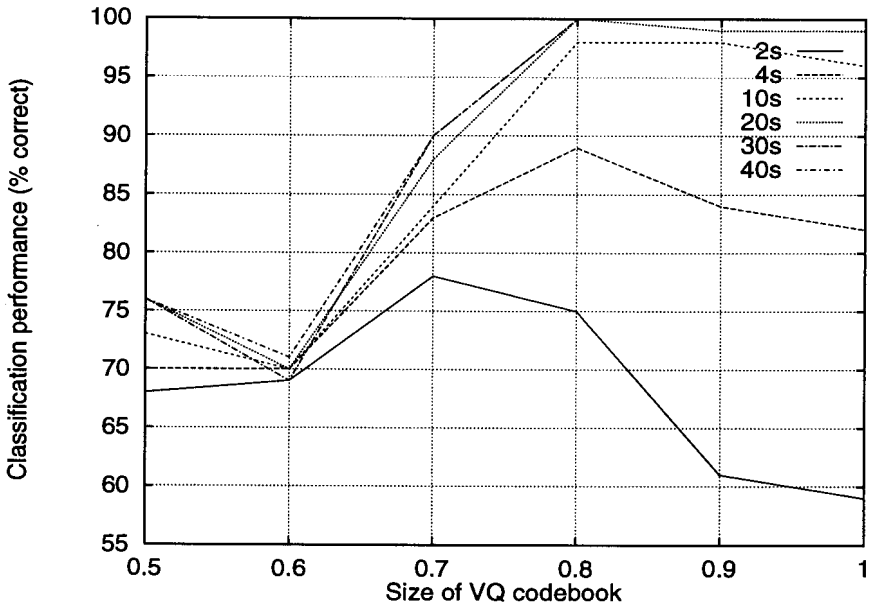

Figure B.50: Mandarin - Spanish language classification performance on the training set as a function of the minimum distinctiveness measure value and utterance duration.

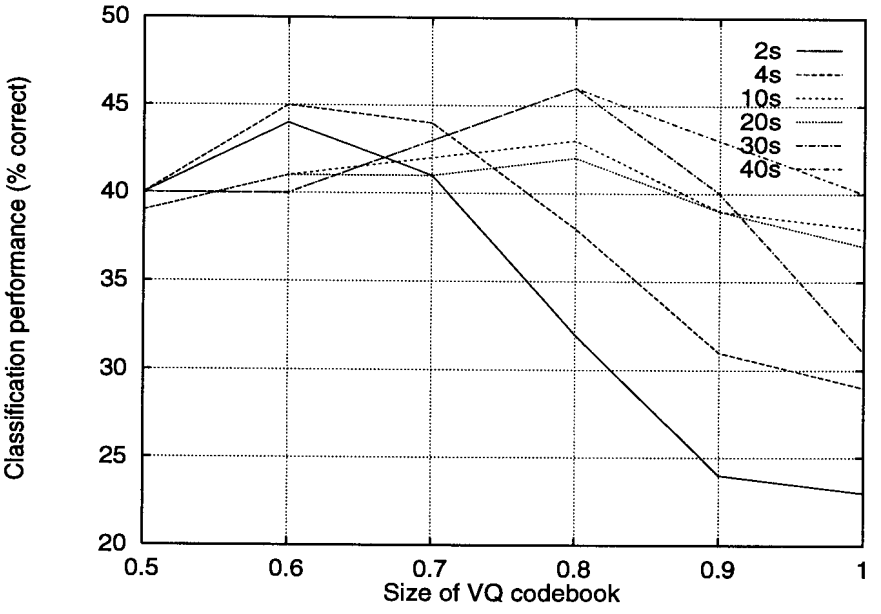

Figure B.51: English - German language classification performance on development set as a function of minimum distinctiveness measure value and utterance duration.

B.7 VQLM system, minimum distinctiveness measure value, development set

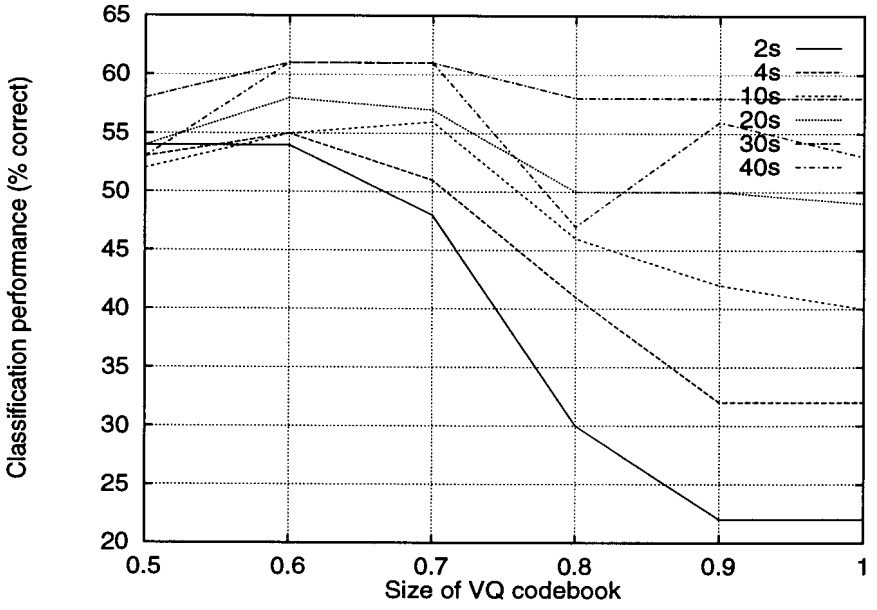

Figure B.52: English - Japanese language classification performance on development set as a function of minimum distinctiveness measure value and utterance duration.

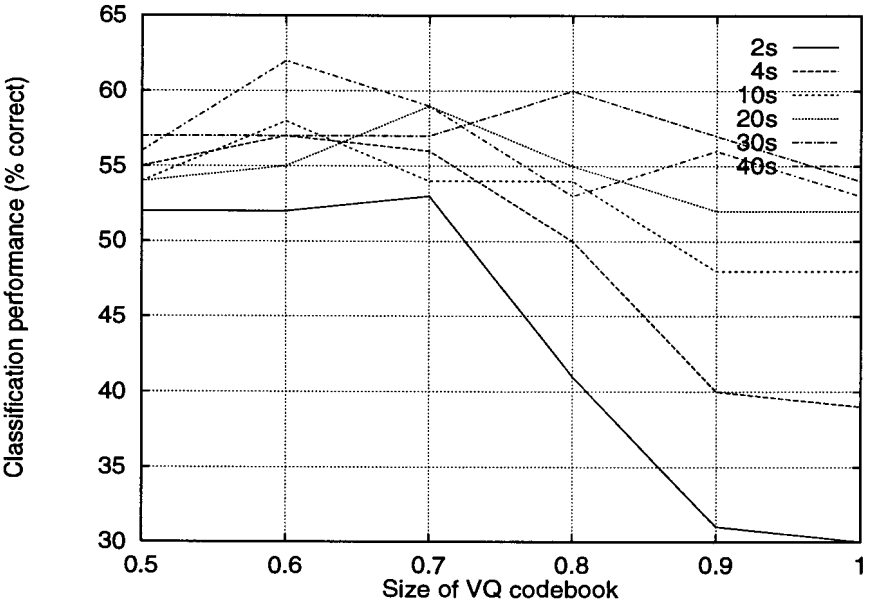

Figure B.53: English - Mandarin language classification performance on development set as a function of minimum distinctiveness measure value and utterance duration.

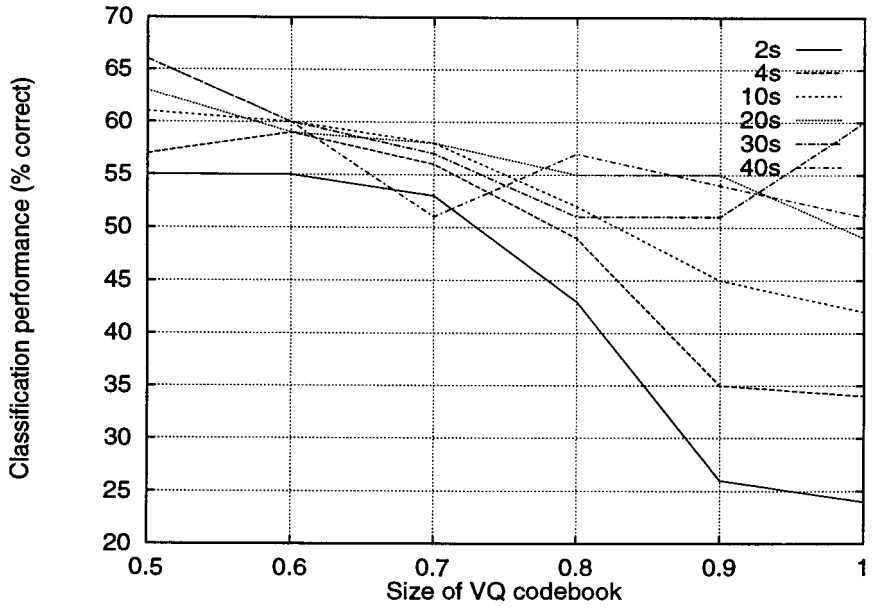

Figure B.54: English - Spanish language classification performance on development set as a function of minimum distinctiveness measure value and utterance duration.

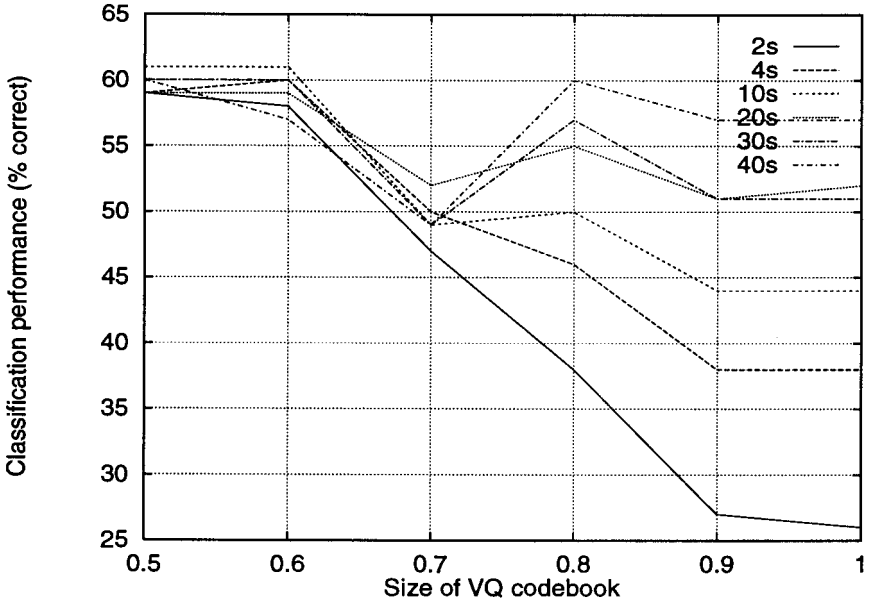

Figure B.55: German - Japanese language classification performance on development set as a function of minimum distinctiveness measure value and utterance duration.

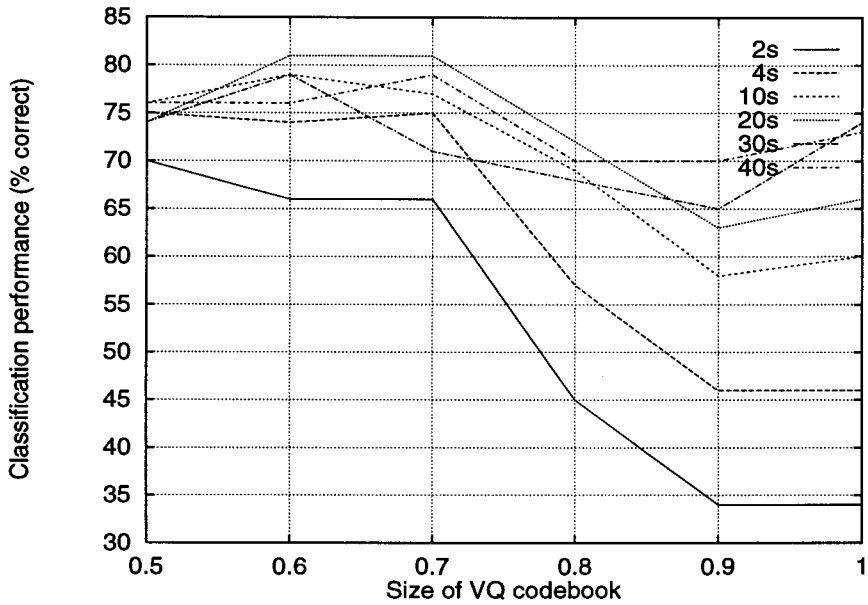

Figure B.56: German - Mandarin language classification performance on development set as a function of minimum distinctiveness measure value and utterance duration.

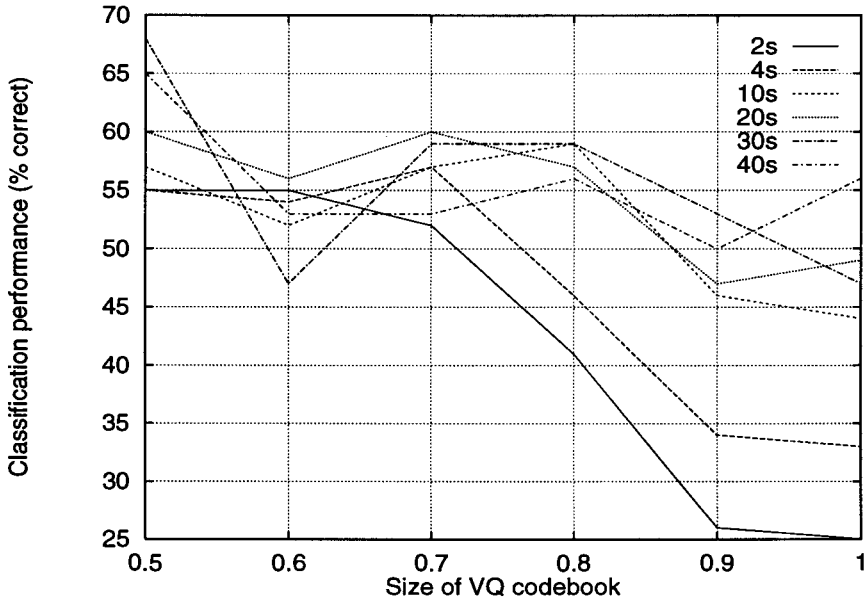

Figure B.57: German - Spanish language classification performance on development set as a function of minimum distinctiveness measure value and utterance duration.

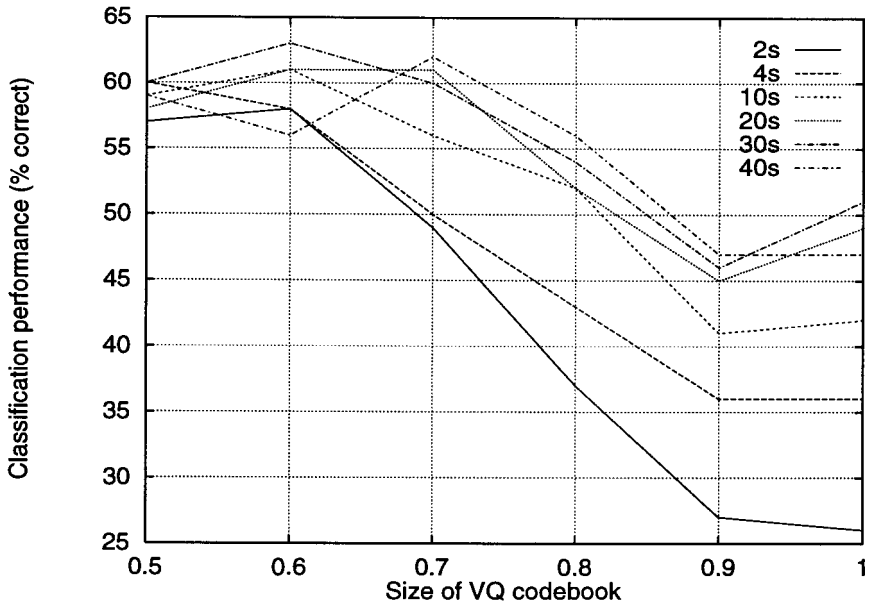

Figure B.58: Japanese - Mandarin language classification performance on development set as a function of minimum distinctiveness measure value and utterance duration.

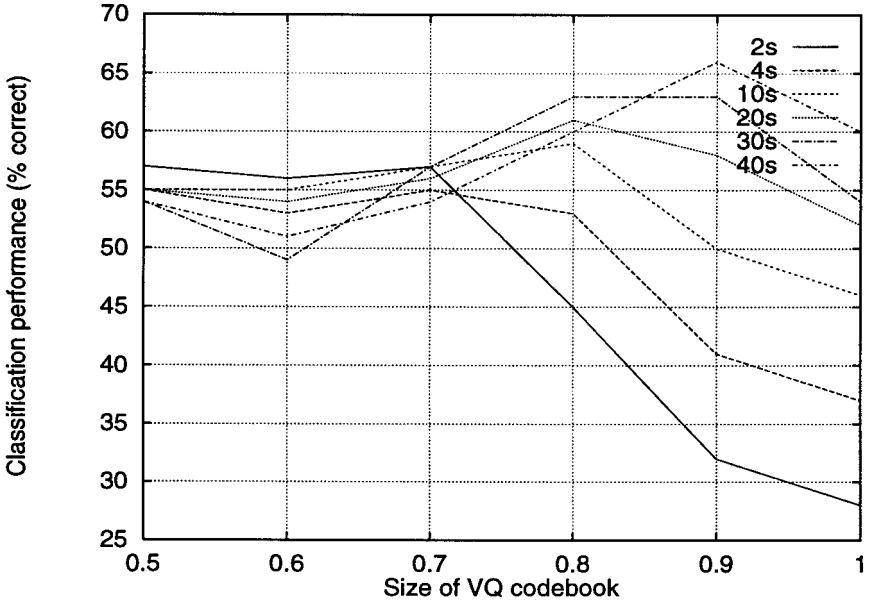

Figure B.59: Japanese - Spanish language classification performance on development set as a function of minimum distinctiveness measure value and utterance duration.

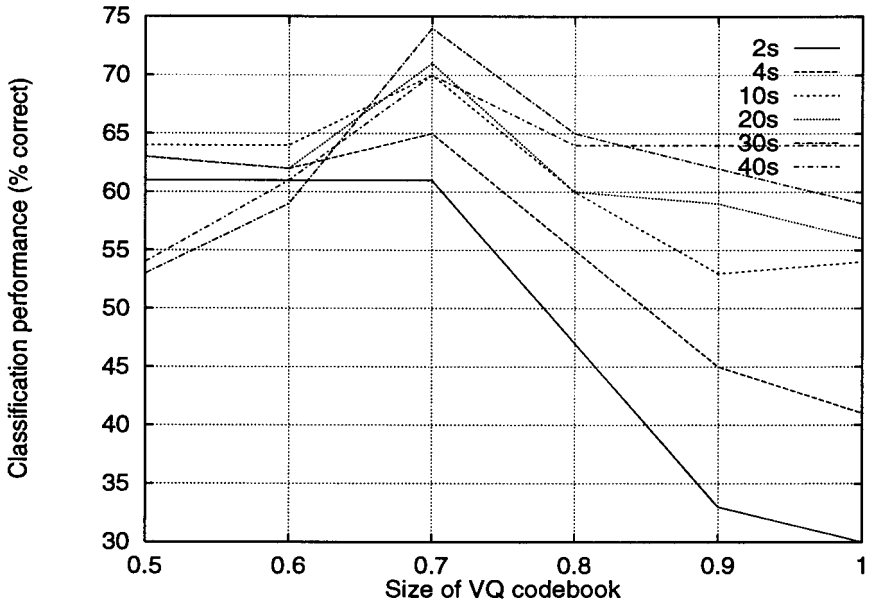

Figure B.60: Mandarin - Spanish language classification performance on development set as a function of minimum distinctiveness measure value and utterance duration.

Appendix C

SPLAT

C.1 Introduction

```
-------------------------------------------------------------------------------
1998-07-31      INTRO           SPLAT 1.2                       H.P. Combrinck
-------------------------------------------------------------------------------

1.  INTRODUCTION
    ------------

Hi, welcome to SPLAT.  SPLAT stands for SPoken Language Analysis
Toolkit.   It is a collection of programs for analysing speech signals
and performing related functions.   These programs are mostly little
more than shells that use a library of speech objects and functions.
Please send comments, suggestions, questions and bug reports to
rikus@suntiger.ee.up.ac.za.

Source Release:  I have now released the source as well.   I did not do
that previously for two reasons:  (1) I did not want multiple
uncorrelated copies drifting around and (2) I did not consider the
code to be user-friendly.   Neither of these have changed, but if other
people can get more out of SPLAT by hacking the code, then so be it.
There are two major weaknesses in the code:  (1) It is not implemented
efficiently and (2) the object-oriented architecture of the whole
thing is neither consistent, nor particularly intuitive.   The first
problem can be addressed by random hacking; the second requires a
total re-write.   If anybody wants to do this, please let me know.   May
the source be with you.

Release 1.2:  Few minor modifications and bug fixes.

Release 1.1:  Added the following new components:  cutwav, lbl2lola,
lola2lbl, trnslola, confuse, entropy.   Most components that used OGI
.lola files now also support PRAAT .Label files.   Documentation and
examples have reached a local minimum and now probably constitutes the
20% of the full documentation set, that provides 80% of the
information.

Release 1.0:  Improves on Release 0.5 (a pre-release version) in that
there is a certain amount of uniformity among the various tools from
the user's point of view.   The software is, however, in a continuing
state of development and may therefore not live up to any expectations
that you might have.   I apologise for the inconvenience and hope that
it will be useful in some way.


2.  SOFTWARE QUALITY
    ----------------

When you write software there are generally three concerns that spring
to mind:  integrity, efficiency and ease of implementation.   When you
use another person's software, ease of implementation is not an issue
and you are (should be?)  interested in the first two points.   Writing
the software was a learning experience, so efficiency came second many
times to ease, elegance and consistency of implementation.   This does
not mean that it was not a concern at all, just that some programs can
run faster and use less memory with a bit of trouble.   Integrity
```

always had highest priority. Where possible, results were tested against examples and common sense. Sometimes (as with the mel-scaled cepstrum features) this was not really possible and I had to be content with something that 'looks right'. Of course, having said this, it is in the nature of software that I cannot guarantee it to be bug free. Since the software is in a continuous state of development (where have I heard this before?), there will probably be some bugs. These, however, should be small and obvious. If you get the software to generate a segmentation fault for instance, please let me know immediately.

3. COMMAND SYNTAX AND USAGE

When a SPLAT command is invoked without any command line switches it produces a usage message:

```
>melcep
Melcep Version 4.1 last compiled on Feb 20 1997.
A component of SPLAT Release 1.0
Copyright (c) 1997 Rikus Combrinck.  All rights reserved.

usage:  melcep [<options>] <source>

-a <circular autocorrelation enable> (switch)
-o <destination> (*.mel)
-w <window size> (10.0)
-s <step size> (2.0)
-p <preemphasis constant> (0.98)
-f <number of filters> (11)
-c <number of coefficients> (9)
-h help
```

All commands are self-documenting; the help switch (-h) supplies a brief help message:

```
>melcep -h
Melcep Version 4.1 last compiled on Feb 20 1997.
A component of SPLAT Release 1.0
Copyright (c) 1997 Rikus Combrinck.  All rights reserved.

usage:  melcep [<options>] <source>

-a <circular autocorrelation enable> (switch)
-o <destination> (*.mel)
-w <window size> (10.0)
-s <step size> (2.0)
-p <preemphasis constant> (0.98)
-f <number of filters> (11)
-c <number of coefficients> (9)

This  command extracts mel-scaled cepstrum features from a speech
signal file <source>.  The signal is preemphasised using a preem-
phasis  constant  of <preemphasis constant>.  A hamming window of
size <window size> milliseconds is moved over the signal  in  in-
crements  of <step size> milliseconds.  For each frame <number of
coefficients> mel-scaled cepstrum coefficients are calculated us-
ing <number of filters> triangular filters.  The first and second
derivatives of these coefficients are calculated as well and  ap-
pended  to the feature vector.  The output is written in OGI tdat
format to <destination>.  Circular autocorrelation filtering is a
bad idea.
```

Options are one of four kinds. There are ordered options that should be supplied in the order indicated and without any switches. In the example there is only one, <source>, the source file. Then there are mandatory switch-value pairs that have to be supplied (none in the example). Thirdly there are switches that take no argument and only switches on or off some feature ('-a' in this case). The last and most common form is some parameter specified by a switch-value pair that has a default value indicated in parentheses.

A useful feature is the file masks. For this example the output file <destination> is specified by '*.mel'. The asterisk will be replaced by the base of the source file. So if the source file was 's1.wav', then the output file will be 's1.mel'. The output file can be a single asterisk '*' which is equivalent to '*.*' and will be replaced by the source file name. This is usually not a good thing to do since the output will overwrite the source file. In general the mask may be

```
any string containing a maximum of two asterisks which will be
replaced by the base and extension of the source file name.   In
addition, any leading directory names are stripped from the source
file name before substitution.   If a file mask contains asterisks, it
should be protected from the shell with double or single quotes.

All this stuff is useful if you are calling SPLAT commands from within
a script and would like to access source files from one directory and
write the output to another directory, with the output name associated
in some way with the input name.   So, for instance:

>melcep -o "/u/rikus/data/mel/*.mel" /u/rikus/data/wav/f1.wav

would be both likely and useful.


4.  SPLAT DATA FILES
   ---------------

SPLAT commands operate on and produce various data files.   These data
files are usually one of the following types:

 .wav   - NIST .wav file format
 .nsp   - another speech file format
 .raw   - 16-bit binary (low byte, hight byte), read as signed short int
 .mel   - mel-scaled cepstral coeffs in OGI .tdat format
 .tfa   - spectrogram in OGI .tdat format
 .tca   - time correlation analysis in OGI .tdat format
 .seq   - sequence of unsigned ints in ascii format
 .ens   - ensemble; multiple sequences
 .vtr   - vector of things (usually numbers in ascii format)
 .mtx   - matrix of things (usually numbers in ascii format)
 .cbk   - code book (.mtx format)
 .hmm   - hidden markov model - contains pi vector, a and b matrices
 .phn   - label file in TIMIT .phn format
 .lola  - label file in OGI .lola format
 .Label - label file in PRAAT .Label format

NOTE: 'Sound file' in this document means either a .wav or .nsp file.

CAUTION: The .nsp file format is currently only supported on Intel
architectures.   Reading it on another architecture (like SUN), will
result in errors, garbage, undefined behaviour or some such bad thing.
This will hopefully be fixed soon.
```

C.2 Conversion utilities

Component	Description
raw2vtr	Convert raw format to ascii vector format.
raw2wav	Convert raw format to NIST wav format.
vtr2wav	Convert ascii vector format to NIST wav format.
vtr2raw	Convert ascii vector format to raw format.
wav2raw	Convert NIST wav format to raw format.
wav2vtr	Convert NIST wav format to ascii vector format.
nsp2wav	Convert nsp format to NIST wav format.
tdat2mtx	Convert OGI tdat format to ascii matrix format.
mtx2tdat	Convert ascii matrix format to OGI tdat format.
phn2lola	Convert TIMIT phn format to OGI lola format.
lola2lbl	Convert OGI lola format to PRAAT Label format.
lbl2lola	Convert PRAAT Label format to OGI lola format.

```
Raw2vtr Version 1.1 last compiled on Sep 30 1999.
A component of SPLAT Release 1.2
Copyright (c) 1999 H.P. Combrinck.   All rights reserved.

usage:  raw2vtr [<options>] <source>

-o <destination> (*.vtr)

This  command  converts a file <source> from SPLAT .raw format to
SPLAT .vtr format <destination>.

-------------------------------------------------------------------
Raw2wav Version 2.1 last compiled on Sep 30 1999.
A component of SPLAT Release 1.2
Copyright (c) 1999 H.P. Combrinck.   All rights reserved.

usage:  raw2wav [<options>] <source>

-o <destination> (*.wav)
-f <sample rate> (8000)

This command converts a file <source> from SPLAT .raw  format  to
TIMIT  .wav  format  <destination>.    <sample rate> is the sample
rate of the signal.   It defaults to 8000Hz.

-------------------------------------------------------------------
Vtr2wav Version 1.1 last compiled on Sep 30 1999.
A component of SPLAT Release 1.2
Copyright (c) 1999 H.P. Combrinck.   All rights reserved.

usage:  vtr2wav [<options>] <source>

-o <destination> (*.wav)
-f <sample rate> (8000)
-a <max level> (detect)

This command converts a file <source> from SPLAT .vtr  format  to
TIMIT  .wav  format  <destination>.    <sample rate> is the sample
rate of the signal.   It defaults to 8000Hz.   <max level>  is  the
maximum amplitude of the signal and is auto detected if not spec-
ified.

-------------------------------------------------------------------
Vtr2raw Version 1.1 last compiled on Sep 30 1999.
A component of SPLAT Release 1.2
Copyright (c) 1999 H.P. Combrinck.   All rights reserved.

usage:  vtr2raw [<options>] <source>

-o <destination> (*.raw)

This  command  converts a file <source> from SPLAT .vtr format to
SPLAT .raw format <destination>.

-------------------------------------------------------------------
Wav2raw Version 2.1 last compiled on Sep 30 1999.
A component of SPLAT Release 1.2
Copyright (c) 1999 H.P. Combrinck.   All rights reserved.

usage:  wav2raw [<options>] <source>

-o <destination> (*.raw)

This command converts a file <source> from TIMIT .wav  format  to
SPLAT .raw format <destination>.

-------------------------------------------------------------------
Wav2vtr Version 1.1 last compiled on Sep 30 1999.
A component of SPLAT Release 1.2
Copyright (c) 1999 H.P. Combrinck.   All rights reserved.

usage:  wav2vtr [<options>] <source>

-o <destination> (*.vtr)

This  command  converts a file <source> from TIMIT .wav format to
SPLAT .vtr format <destination>.

-------------------------------------------------------------------
Nsp2wav Version 1.1 last compiled on Sep 30 1999.
A component of SPLAT Release 1.2
Copyright (c) 1999 H.P. Combrinck.   All rights reserved.

usage:  nsp2wav [<options>] <source>

-o <destination> (*.wav)

This  command  converts a sound file <source> from .nsp format to
```

```
NIST .wav format <destination>.

-----------------------------------------------------------------
Tdat2mtx Version 1.1 last compiled on Sep 30 1999.
A component of SPLAT Release 1.2
Copyright (c) 1999 H.P. Combrinck.   All rights reserved.

usage:  tdat2mtx [<options>] <source>

-o <destination> (*.mtx)

This command converts a file <source> from OGI  .tdat  format  to
SPLAT .mtx format <destination>.

-----------------------------------------------------------------
Mtx2tdat Version 1.1 last compiled on Sep 30 1999.
A component of SPLAT Release 1.2
Copyright (c) 1999 H.P. Combrinck.   All rights reserved.

usage:  mtx2tdat [<options>] <source>

-o <destination> (*.tdat)
-f <sample rate> (8000)
-w <window size> (10.0)
-s <step size> (2.0)
-p <start time> (0.0)

This command converts a file <source> from SPLAT .mtx  format  to
OGI  .tdat  format  <destination>.    It  takes parameters <sample
rate>, <window size>, <step size> and <start time>.   These param-
eters  must be known and correct.   The defaults are only supplied
as much used values.

-----------------------------------------------------------------
Phn2lola Version 1.1 last compiled on Sep 30 1999.
A component of SPLAT Release 1.2
Copyright (c) 1999 H.P. Combrinck.   All rights reserved.

usage:  phn2lola [<options>] <source>

-o <destination> (*.lola)

This command converts a label file <source> from TIMIT .phn  for-
mat to OGI .lola format <destination>.

-----------------------------------------------------------------
Lola2lbl Version 1.0 last compiled on Sep 30 1999.
A component of SPLAT Release 1.2
Copyright (c) 1999 H.P. Combrinck.   All rights reserved.

usage:  lola2lbl [<options>] <source>

-o <destination> (*.Label)

This  command  converts  a file <source> from OGI .lola format to
PRAAT .Label format <destination>.

-----------------------------------------------------------------
Lbl2lola Version 1.0 last compiled on Sep 30 1999.
A component of SPLAT Release 1.2
Copyright (c) 1999 H.P. Combrinck.   All rights reserved.

usage:  lbl2lola [<options>] <source>

-o <destination> (*.lola)

This command converts a file <source> from PRAAT .Label format to
OGI .lola format <destination>.

-----------------------------------------------------------------
```

C.3 Speech file operations

Component	Description
wavgain	Adjust gain of NIST wav file.
wavrate	Adjust sampling rate of NIST wav file.
noisify	Add Gaussian noise to speech file.
silabel	Perform word end-point detection.
remsil	Remove silence from speech file.
phnseg	Perform phonetic segmentation of speech file.
cutwav	Extract labelled speech segments from speech file.
chopwav	Split speech file into equal length segments.

```
-----------------------------------------------------------------
Wavgain Version 1.1 last compiled on Sep 30 1999.
A component of SPLAT Release 1.2
Copyright (c) 1999 H.P. Combrinck.   All rights reserved.

usage:  wavgain [<options>] <source>

-o <destination> (*)
-a <amplitude level> (32760.0)

This command adjusts the maximum amplitude level <amplitude  lev-
el>  of  a .wav file <source>.   The result is stored in <destina-
tion>.   Note that  the  default  <destination>  is  the  same  as
<source> so that <source> will be overwritten.

-----------------------------------------------------------------
Wavrate Version 1.1 last compiled on Sep 30 1999.
A component of SPLAT Release 1.2
Copyright (c) 1999 H.P. Combrinck.   All rights reserved.

usage:  wavrate [<options>] <source> <sampling rate>

-o <destination> (*)

This  command adjusts the sampling rate <sampling rate> of a .wav
file <source>.   The result is stored in <destination>.   Note that
the  default  <destination>  is  the  same  as  <source>  so that
<source> will be overwritten.

-----------------------------------------------------------------
Noisify Version 1.1 last compiled on Sep 30 1999.
A component of SPLAT Release 1.2
Copyright (c) 1999 H.P. Combrinck.   All rights reserved.

usage:  noisify [<options>] <source>

-o <destination> (*.n.*)
-f <variance fraction> (0.1)

This command adds gaussian noise to a  .wav  file  <source>  with
variance  that  is  <variance fraction> of the total signal vari-
ance.   This is not  a  very  sophisticated  approach,  since  the
amount  of  silence have an effect on total signal variance.   The
resulting signal is stored in <destination>.

-----------------------------------------------------------------
Silabel Version 1.5 last compiled on Sep 30 1999.
A component of SPLAT Release 1.2
Copyright (c) 1999 H.P. Combrinck.   All rights reserved.

usage:  silabel [<options>] <source>

-o <destination> (*.sil.lola)
-w <window size> (10.0)
-s <step size> (2.0)
-p <preemphasis constant> (0.98)
-n <minimum no-voice duration> (300.0)
-v <minimum voice duration> (30.0)
```

```
-t <threshold> (1.0)

This command detects silence in a speech signal <source> and la-
bels it accordingly. It can be used to extract discrete words
from a speech stream. The signal is preemphasised using a preem-
phasis constant of <preemphasis constant>. A hamming window of
size <window size> milliseconds is moved over the signal in in-
crements of <step size> milliseconds. The energy for each frame
is calculated and thresholded to make a silence/voice decision.
In order for a segment to be labelled as silence ("" in the des-
tination file), it must be at least <minimum no-voice duration>
milliseconds long. Segments labelled as speech ("w" in the des-
tination file), should equal or exceed <minimum voice duration>
milliseconds. The decision threshold is auto-detected and should
work well in most cases. It can be fine-tuned with the <thresh-
old> factor. The output is written in OGI lola format to
<destination>.

------------------------------------------------------------------
Remsil Version 1.0 last compiled on Sep 30 1999.
A component of SPLAT Release 1.2
Copyright (c) 1999 H.P. Combrinck. All rights reserved.

usage: remsil [<options>] <source>

-c <clean> (switch)
-o <destination> (*.v.*)
-l <lola source> (*.lola)
-q <lola destination> (*.v.lola)
-w <window size> (10.0)
-s <step size> (2.0)
-p <preemphasis constant> (0.98)
-n <minimum no-voice duration> (300.0)
-v <minimum voice duration> (30.0)
-t <threshold> (1.0)

The following stuff is incomplete and partly wrong.

This command detects silence in a speech signal <source> and la-
bels it accordingly. It can be used to extract discrete words
from a speech stream. The signal is preemphasised using a preem-
phasis constant of <preemphasis constant>. A hamming window of
size <window size> milliseconds is moved over the signal in in-
crements of <step size> milliseconds. The energy for each frame
is calculated and thresholded to make a silence/voice decision.
In order for a segment to be labelled as silence ("" in the des-
tination file), it must be at least <minimum no-voice duration>
milliseconds long. Segments labelled as speech ("w" in the des-
tination file), should equal or exceed <minimum voice duration>
milliseconds. The decision threshold is auto-detected and should
work well in most cases. It can be fine-tuned with the <thresh-
old> factor. The output is written in OGI lola format to
<destination>.

------------------------------------------------------------------
Phnseg Version 1.3 last compiled on Sep 30 1999.
A component of SPLAT Release 1.2
Copyright (c) 1999 H.P. Combrinck. All rights reserved.

usage: phnseg [<options>] <source>

-o <destination> (*.lola)
-d <distortion file> (none)
-t <threshold level> (0.5)
-r <region size> (10)

This command segments a tdat (feature) file <source> into
phonemes. The result is stored as a label file <destination>.
The 'distance' between two successive frames is calculated using
a distortion measure. The distance (or distortion) is smoothed
using a quadratic filter with a region of support of <region
size> samples. Phoneme boundaries are assumed to exist where the
distortion exceeds <threshold level> as a fraction of the mean
distortion. NOTE: Phoneme segmentation is a very difficult prob-
lem and accordingly this software does not work very well.

------------------------------------------------------------------
Cutwav Version 1.3 last compiled on Sep 30 1999.
A component of SPLAT Release 1.2
Copyright (c) 1999 H.P. Combrinck. All rights reserved.

usage: cutwav [<options>] <source>

-n <disable numbering> (switch)
-c <disable label coding> (switch)
-l <label file> (*.lola)
-o <sound output mask> (*.*.wav)
```

```
-p <label output mask> (*.*.lola)

This command takes a sound file <source> and its corresponding
label file <label file> as input and then generates a sound file
and a label file for each label. The output is written to files
named using the masks <sound output mask> and <label output
mask>. These masks differ from the SPLAT mask convention and is
defined as follows: the first (or only) asterisk is replaced by a
name based on the label as found in the lola file; the second (if
present) is replaced by the base of <source>. A number repre-
senting the n'th instance of the label is appended to the label
name (first asterisk) in the output filename. <disable number-
ing> disables this feature and instances of the same label are
then simply overwritten. Since output files are named after la-
bels, and labels may contain filename-unfriendly characters, the
labels are encoded by default. <disable label coding> disables
this feature.

------------------------------------------------------------------
Chopwav Version 1.0 last compiled on Sep 30 1999.
A component of SPLAT Release 1.2
Copyright (c) 1999 H.P. Combrinck. All rights reserved.

usage: chopwav [<options>] <source>

-l <label file> (*.lola)
-o <sound output mask> (*.wav)
-p <label output mask> (*.lola)
-s <chunk size> (10)

Beta version. <chunk size> in seconds.

------------------------------------------------------------------
```

C.4 Feature extraction

Component	Description
tfa	Perform time-frequency analysis of speech file (spectogram).
melcep	Extract mel-scaled cepstrum features from speech file.
tca	Perform time correlation analysis on speech file.

```
------------------------------------------------------------------
Tfa Version 1.1 last compiled on Sep 30 1999.
A component of SPLAT Release 1.2
Copyright (c) 1999 H.P. Combrinck. All rights reserved.

usage: tfa [<options>] <source>

-a <circular autocorrelation enable> (switch)
-o <destination> (*.tfa)
-w <window size> (10.0)
-s <step size> (2.0)
-p <preemphasis constant> (0.98)

This command performs time-frequency analysis (i.e. a spectro-
gram) on a speech signal file <source>. The signal is preempha-
sised using a preemphasis constant of <preemphasis constant>. A
hamming window of size <window size> milliseconds is moved over
the signal in increments of <step size> milliseconds. For
each frame a fft is calculated. The output is written in OGI
tdat format to <destination>. Circular autocorrelation filtering
is a bad idea.

------------------------------------------------------------------
Melcep Version 4.2 last compiled on Sep 30 1999.
A component of SPLAT Release 1.2
Copyright (c) 1999 H.P. Combrinck. All rights reserved.

usage: melcep [<options>] <source>

-a <circular autocorrelation enable> (switch)
-n <no energy> (switch)
```

```
-o <destination> (*.mel)
-w <window size> (10.0)
-s <step size> (2.0)
-p <preemphasis constant> (0.98)
-f <number of filters> (11)
-c <number of coefficients> (9)

This  command extracts mel-scaled cepstrum features from a speech
signal file <source>.   The signal is preemphasised using a preem-
phasis  constant  of <preemphasis constant>.   A hamming window of
size <window size> milliseconds is moved over the signal  in  in-
crements  of <step size> milliseconds.   For each frame <number of
coefficients> mel-scaled cepstrum coefficients are calculated us-
ing <number of filters> triangular filters.   The first and second
derivatives of these coefficients are calculated as well and  ap-
pended  to the feature vector.   The output is written in OGI tdat
format to <destination>.   Circular autocorrelation filtering is a
bad idea.

------------------------------------------------------------------
Tca Version 2.1 last compiled on Sep 30 1999.
A component of SPLAT Release 1.2
Copyright (c) 1999 H.P. Combrinck.   All rights reserved.

usage:  tca [<options>] <source>

-a <circular autocorrelation enable> (switch)
-o <destination> (*.tca)
-w <window size> (10.0)
-s <step size> (2.0)
-p <preemphasis constant> (0.98)
-n <number of coefficients> (9)
-c <centre frequency> (200)

This command performs time correlation analysis on a speech  sig-
nal file <source>.   The signal is preemphasised using a preempha-
sis constant of <preemphasis constant>.   A hamming window of size
<window  size>  milliseconds is moved over the signal  in  incre-
ments  of <step size> milliseconds.   For each  frame  <number  of
coefficients>  coefficients  are  calculated, using a centre fre-
quency of <centre frequency>.   The output is written in OGI  tdat
format to <destination>.   Circular autocorrelation filtering is a
bad idea.

------------------------------------------------------------------
```

C.5 Vector quantisation

Component	Description
sconn	Perform vector quantisation and create codebook.
sconntst	Encode data file using VQ codebook.

```
------------------------------------------------------------------
Sconn Version 2.1 last compiled on Sep 30 1999.
A component of SPLAT Release 1.2
Copyright (c) 1999 H.P. Combrinck.   All rights reserved.

usage:  sconn [<options>] <source>

-n <maximum node count> (mandatory)
-u <sense range high> (mandatory)
-v <sense range low> (mandatory)
-m <sense range minimum> (mandatory)
-o <code book> (*.cbk)
-e <existing code book> (none)
-p <maximum pass count> (1000)
-s <random seed> (123)
-r <report every> (100)
-a <alpha> (0.085)
-f <r factor> (0.85)
-c <decay constant> (0.0001)
```

```
This  command creates a vector quantisation code book <code book>
from the mtx file  <source>.    The  rows  in  <source>  represent
training vectors.   The training process is stopped when the maxi-
mum number of output vectors exceeds <maximum  node  count>,  the
number of passes through the whole data set exceeds <maximum pass
count> or the cluster size drops  below  <sense  range  minimum>.
Cluster  size  decays  exponentially  from  <sense range high> to
<sense range low> at  a  rate  determined  by  <decay  constant>.
<random seed> seeds the random number generator and the status of
the algorithm is reported every <report every>  iterations.    See
documentation  for  definitions  of <alpha> and <r factor>.   (The
defaults should work well.)   <existing code book> is an  existing
code  book that can be updated using the training file.   WARNING:
Updating an existing codebook has  not  been  thoroughly  tested.
Using this feature is a bad idea at the moment.

-------------------------------------------------------------------
Sconntst Version 1.2 last compiled on Sep 30 1999.
A component of SPLAT Release 1.2
Copyright (c) 1999 H.P. Combrinck.   All rights reserved.

usage:  sconntst [<options>] <source> <code book>

-o <sequence> (*.seq)

This command encodes (vector quantises) a mtx file <source> using
the code book <code book>.   The output is a set of encoded vector
quanta written to <sequence>.

-------------------------------------------------------------------
```

C.6 Hidden Markov modelling

Component	Description
hmmtrain	Estimate HMM parameters from observation sequence.
hmmtest	Find probability that observation sequence was generated by HMM.
hmmgen	Generate an observation sequence given a certain HMM.
hmmdist	Find "distance" between two HMMs.
buildens	Build ensemble file from labelled speech file.
label	Perform phone classification on pre-segmented speech file.
lola2seq	Create token sequence file from OGI lola file.
confuse	Create confusion matrix of phone classification.
entropy	Calculate entropy of token sequence.

```
-------------------------------------------------------------------
Hmmtrain Version 1.4 last compiled on Sep 30 1999.
A component of SPLAT Release 1.2
Copyright (c) 1999 H.P. Combrinck.   All rights reserved.

usage:  hmmtrain [<options>] <observation>

-n <number of states> (mandatory)
-m <number of symbols> (mandatory)
-l <left-to-right> (switch)
-o <estimated model> (*.hmm)
-u <existing model> (none)
-e <minimum error delta> (0.000001)
-a <minimum iteration count> (10)
-z <maximum iteration count> (10000)
-r <number of restarts> (5)
-s <seed> (123)

This command estimates a hidden markov model from an  observation
file  <observation> and writes it to the hmm file <estimated mod-
```

```
el>.  The model has <number of states> states and <number of sym-
bols>  symbols.  It is initialised either with an existing model
<existing model> or with a random model generated using the  seed
<seed>.  In  this  case it can be restarted <number of restarts>
times with different random models.  The optimal model will auto-
matically  selected.  During training a minimum number of itera-
tions <minimum iteration count> is  forced  and  the  process  is
stopped  either  at  <maximum iteration count> iterations or when
the change in error is smaller than <minimum error delta>.

-------------------------------------------------------------------
Hmmtest Version 1.2 last compiled on Sep 30 1999.
A component of SPLAT Release 1.2
Copyright (c) 1999 H.P. Combrinck.  All rights reserved.

usage:  hmmtest [<options>] <observation> <hmm>


This  command  finds the probability that the sequence(s) in <ob-
servation> was generated by the hidden markov model in <hmm>.

-------------------------------------------------------------------
Hmmgen Version 1.1 last compiled on Sep 30 1999.
A component of SPLAT Release 1.2
Copyright (c) 1999 H.P. Combrinck.  All rights reserved.

usage:  hmmgen [<options>] <hmm>

-o <sequence> (*.seq)
-n <observation count> (10)

This command generates an observation sequence  <sequence>  using
the  hidden  markov model <hmm>.  The sequence contains <observa-
tion count> observations.  The random number generator is  seeded
from the system clock.

-------------------------------------------------------------------
Hmmdist Version 1.0 last compiled on Sep 30 1999.
A component of SPLAT Release 1.2
Copyright (c) 1999 H.P. Combrinck.  All rights reserved.

usage:  hmmdist [<options>] <hmm1> <hmm2>

-t <# of observations> (50)

This command finds the distance between two HMM's.

-------------------------------------------------------------------
Buildens Version 1.3 last compiled on Sep 30 1999.
A component of SPLAT Release 1.2
Copyright (c) 1999 H.P. Combrinck.  All rights reserved.

usage:  buildens [<options>] <sequence>

-c <disable label coding> (switch)
-f <label using filename> (switch)
-l <label file> (*.lola)
-o <destination> (*.ens)
-s <step size> (2.0)

This  command  creates  a set of ensemble files from the sequence
file <sequence> and the matching label file  <label  file>.  The
names of the ensemble files are determined by the file mask <des-
tination> and the labels in the label  file.  An  ensemble  file
contains all the sequences present in the sequence file that rep-
resent a certain symbol as determined by the label  file.  <step
size> is the frame increment (in milliseconds) that was used dur-
ing feature extraction.  Since ensemble files are named after la-
bels,  and labels may contain filename-unfriendly characters, the
labels are encoded by default.  <disable label  coding>  disables
this  feature.  If  the  sequence file <sequence> contain only a
single label (e.g.  a word), then the ensemble file can  be  named
after  the sequence file and no label file is needed.  The <label
using filename> switch does this.  In this case the  output  file
naming  convention  is non-standard.  Only the first asterisk has
meaning in the <destination> file mask - it is replaced with  the
first part (up to the first '.')  of <sequence>.

-------------------------------------------------------------------
Label Version 1.3 last compiled on Sep 30 1999.
A component of SPLAT Release 1.2
Copyright (c) 1999 H.P. Combrinck.  All rights reserved.

usage:  label [<options>] <observation> <hmm list>

-n <no squeeze> (switch)
-l <seg file> (*.seg.lola)
-s <step size> (2.0)
```

```
-k <hmm mask> (*)
-o <destination> (*.lola)

This command automatically labels an utterance represented by the
sequence of codebook symbols in <observation>. The utterance
must be pre-segmented in <seg file>. (Only the time information
in this file is used - the labels are ignored.) The sequence of
symbols are assumed to be generated from frames incremented by
<step size> during the feature extraction process. <hmm list>
should contain a white space delimited list of hmm file names
that represent the set of possible labels. Each segment in the
label will be assigned the base name of the file in the list rep-
resenting the closest matching hmm. The names in <hmm list>
should not contains paths. The search path is determined by
<hmm mask>. Labels are squeezed by default (i.e. adjacent repe-
titions merged). <no squeeze> disables this feature. Output is
written to <destination>.

-------------------------------------------------------------------
Lola2seq Version 1.2 last compiled on Sep 30 1999.
A component of SPLAT Release 1.2
Copyright (c) 1999 H.P. Combrinck. All rights reserved.

usage: lola2seq [<options>] <source>

-o <sequence> (*.seq)
-l <hmm list> (hmm_list)

This command creates a .seq file <sequence> from a label file
<source> and an accompanying list of hidden markov model files
<hmm list>, that were used to generate the label file.

-------------------------------------------------------------------
Confuse Version 1.3 last compiled on Sep 30 1999.
A component of SPLAT Release 1.2
Copyright (c) 1999 H.P. Combrinck. All rights reserved.

usage: confuse [<options>] <file list> <hmm list>

-o <destination> (none)
-t <label test> (*.lola)
-d <label target> (*.target.lola)

This command compares .lola files generated by some classifier
(such as 'label') to correctly labeled files and creates a confu-
sion matrix to evaluate the classifier performance. It takes as
input a whitespace delimited list of filenames <file list> which
is used together with the mask <label test> to determine the
names of the .lola files to be tested. These files are compared
to (the desired) .lola files designated by the mask <label tar-
get>. The confusion matrix is reported to standard output and
can optionally also be written to a file <destination>. Valid
labels are deduced from <hmm list>.

-------------------------------------------------------------------
Entropy Version 1.1 last compiled on Sep 30 1999.
A component of SPLAT Release 1.2
Copyright (c) 1999 H.P. Combrinck. All rights reserved.

usage: entropy [<options>] <source>


This command calculates the entropy (average information in bits
per symbol) of the alphabet used to represent the .seq file
<source>.

-------------------------------------------------------------------
```

C.7 Language modelling

Component	Description
grammar	Find N-gram histograms from token sequences.
hst2grm	Find N-gram language models from histograms.
score	Find language scores for token sequence.

```
-----------------------------------------------------------------
Grammar Version 1.7 last compiled on Sep 30 1999.
Copyright (c) 1999 H.P. Combrinck.   All rights reserved.

usage:  grammar <frame size> <max_n_bins> <max_n_chars> <min_distinct>
 <basename 1> [ ...  <basename 12>]

-----------------------------------------------------------------
Hst2grm Version 1.1 last compiled on Sep 30 1999.
Copyright (c) 1999 H.P. Combrinck.   All rights reserved.

usage:  hst2gram <max_n_cbk> <basename 1> [ ...  <basename 12>]

-----------------------------------------------------------------
Score Version 1.5 last compiled on Sep 30 1999.
Copyright (c) 1999 H.P. Combrinck.   All rights reserved.

usage:  score <report every> <testfile> <grammar 1> [ ...  <grammar 12>]

-----------------------------------------------------------------
```

Bibliography

[1] L. Rabiner and B.-H. Juang, *Fundamentals of Speech Recognition.* Prentice Hall Signal Processing Series, Prentice-Hall, Inc., 1993.

[2] K. Atkinson, "Language identification from nonsegmental cues," *Journal of the Acoustical Society of America*, vol. 44, p. 378(A), 1968.

[3] M. Sugiyama, "Automatic language recognition using acoustic features," Tech. Rep. TR-I-0167, ATR Interpreting Telephony Research Laboratories, 1991.

[4] Y. K. Muthusamy, *A Segmental Approach to Automatic Language Identification.* PhD thesis, Oregon Graduate Institute of Science and Technology, Oct. 1993.

[5] Y. K. Muthusamy, N. Jain, and R. A. Cole, "Perceptual benchmarks for automatic language identification," in *Proceedings IEEE International Conference on Acoustics, Speech and Signal Processing 94*, vol. 1, (Adelaide, Australia), pp. 333–336, Apr. 1994.

[6] Y. K. Muthusamy, R. A. Cole, and B. T. Oshika, "The OGI multi-language telephone speech corpus," in *Proceedings International Conference on Spoken Language Processing 92*, vol. 2, (Banff, Alberta, Canada), pp. 895–898, Oct. 1992.

[7] M. A. Zissman, "Comparison of four approaches to automatic language identification of telephone speech," *IEEE Transactions on Speech and Audio Processing*, vol. 4, pp. 31–44, Jan. 1996.

[8] M. Sugiyama, "Automatic language recognition using acoustic features," in *Proceedings IEEE International Conference on Acoustics, Speech and Signal Processing 91*, vol. 2, (Toronto, Canada), pp. 813–816, May 1991.

[9] Y. K. Muthusamy, E. Barnard, and R. A. Cole, "Reviewing automatic language identification," *IEEE Signal Processing Magazine*, vol. 11, pp. 33–41, Oct. 1994.

[10] K. M. Berkling, *Automatic Language Identification with Sequences of Language-Independent Phoneme Clusters.* PhD thesis, Oregon Graduate Institute of Science and Technology, Oct. 1996.

[11] V. Stockmal, D. Muljani, and Z. Bond, "Perceptual features of unknown languages as revealed by multi-dimensional scaling," in *Proceedings Fourth International Conference on Spoken Language Processing*, vol. 3, (Philadelphia, PA, USA), pp. 1748–1751, Oct. 1996.

[12] Z. S. Bond, D. Fucci, V. Stockmal, and D. McColl, "Multi-dimensional scaling of listener responses to complex auditory stimuli," in *Proceedings 5th International Conference on Spoken Language Processing*, vol. 2, (Sydney, Australia), pp. 93–96, Dec. 1998.

[13] V. Stockmal, D. R. Moates, and Z. S. Bond, "Same talker, different language," in *Proceedings 5th International Conference on Spoken Language Processing*, vol. 2, (Sydney, Australia), pp. 97–100, Dec. 1998.

[14] R. G. Leonard and G. R. Doddington, "Automatic language identification," Tech. Rep. RADC-TR-74-200 / TI-347650, Air Force Rome Air Development Centre / Texas Instruments, Inc., Dallas, TX, Aug. 1974.

[15] R. G. Leonard and G. R. Doddington, "Automatic classification of languages," Tech. Rep. RADC-TR-75-264, Air Force Rome Air Development Centre / Texas Instruments, Inc., Dallas, TX, Oct. 1975.

[16] R. G. Leonard and G. R. Doddington, "Automatic language discrimination," Tech. Rep. RADC-TR-78-5, Air Force Rome Air Development Centre / Texas Instruments, Inc., Dallas, TX, Jan. 1978.

[17] R. G. Leonard, "Language recognition test and evaluation," Tech. Rep. RADC-TR-80-83, Air Force Rome Air Development Centre / Texas Instruments, Inc., Dallas, TX, Mar. 1980.

[18] D. Cimarusti and R. B. Ives, "Development of an automatic identification system of spoken languages: Phase I," in *Proceedings IEEE International Conference on Acoustics, Speech and Signal Processing 82*, (Paris, France), pp. 1661–1663, May 1982.

[19] J. T. Foil, "Language identification using noisy speech," in *Proceedings IEEE International Conference on Acoustics, Speech and Signal Processing 86*, vol. 2, (Tokyo, Japan), pp. 861–864, Apr. 1986.

[20] F. J. Goodman, A. F. Martin, and R. E. Wohlford, "Improved automatic language identification in noisy speech," in *Proceedings IEEE International Conference on Acoustics, Speech and Signal Processing 89*, vol. 1, (Glasgow, Scotland), pp. 528–531, May 1989.

[21] M. Savic, E. Acosta, and S. K. Gupta, "An automatic language identification system," in *Proceedings IEEE International Conference on Acoustics, Speech and Signal Processing 91*, vol. 2, (Toronto, Canada), pp. 817–820, May 1991.

[22] S. Nakagawa, Y. Ueda, and T. Seino, "Speaker-independent, text-independent language identification by HMM," in *Proceedings International Conference on Spoken Language Processing 92*, vol. 2, (Banff, Alberta, Canada), pp. 1011–1014, Oct. 1992.

[23] S. C. Kwasnya, B. L. Kalman, W. Wu, and A. M. Engebretson, "Identifying language from speech: An example of high-level, statistically-based feature extraction," in *Proceedings 14th Annual Conference of the Cognitive Science Society*, pp. 909–913, 1992.

[24] S. C. Kwasny, B. L. Kalman, A. M. Engebretson, and W. Wu, "Real-time identifying of language from raw speech waveforms," in *IWANNT 93*, 1993.

[25] W. Wu, S. C. Kwasny, B. L. Kalman, and A. M. Engebretson, "Identifying language from raw speech: An application of recurrent neural networks," in *Proceedings of the 5th Midwest Artificial Intelligence and Cognitive Science Society Conference*, pp. 53–57, 1992.

[26] M. A. Zissman, "Automatic language identification using Gaussian mixtures and hidden Markov models," in *Proceedings IEEE International Conference on Acoustics, Speech and Signal Processing 93*, vol. 2, (Minneapolis, MN, USA), pp. 399–402, Apr. 1993.

[27] L. Riek, W. Mistreta, and D. Morgan, "Experiments in language identification," Tech. Rep. SPCOT-91-002, Lockheed Sanders, Inc., Nashua, NH, Dec. 1991.

[28] J. A. du Preez and D. M. Weber, "Automatic language recognition using high-order HMMs," in *Proceedings 5th International Conference on Spoken Language Processing*, vol. 2, (Sydney, Australia), pp. 117–120, Dec. 1998.

[29] A. S. House and E. P. Neuberg, "Toward automatic identification of the language of an utterance. I. Preliminary methodological considerations," *Journal of the Acoustical Society of America*, vol. 62, pp. 708–713, Sept. 1977.

[30] K. P. Li and T. J. Edwards, "Statistical models for automatic language identification," in *Proceedings IEEE International Conference on Acoustics, Speech and Signal Processing 80*, vol. 3, (Denver, CO, USA), pp. 884–887, Apr. 1980.

[31] R. J. D'Amore and C. P. Mah, "One-time complete indexing of text: Theory and practice," in *Proceedings Eighth International ACM Conference Res. Dev. Information Retrieval*, pp. 155–164, 1985.

[32] J. C. Schmitt, "Trigram-based method of language identification." US Patent 5 062 143, Oct. 1991.

[33] T. A. Albina, "A system for clustering spoken documents," in *Proceedings 2nd European Conference on Speech Communication and Technology*, vol. 2, (Berlin, Germany), pp. 1371–1374, Sept. 1993.

[34] R. C. F. Tucker, M. J. Carey, and E. S. Parris, "Automatic language identification using sub-word models," in *Proceedings IEEE International Conference on Acoustics, Speech and Signal Processing 94*, vol. 1, (Adelaide, Australia), pp. 301–304, Apr. 1994.

[35] L. Lamel and J. Gauvain, "Language identification using phone-based acoustic likelihoods," in *Proceedings IEEE International Conference on Acoustics, Speech and Signal Processing 94*, vol. 1, (Adelaide, Australia), pp. 293–296, Apr. 1994.

[36] Y. K. Muthusamy, K. Berkling, T. Arai, R. A. Cole, and E. Barnard, "A comparison of approaches to automatic language identification using telephone speech," in *Proceedings 2nd European Conference on Speech Communication and Technology*, vol. 2, (Berlin, Germany), pp. 1307–1310, Sept. 1993.

[37] 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 *Proceedings IEEE International Conference on Acoustics, Speech and Signal Processing 94*, vol. 1, (Adelaide, Australia), pp. 121–124, Apr. 1994.

[38] P. Dalsgaard, O. Andersen, H. Hesselager, and B. Petek, "Language-identification using language-dependent phonemes and language-independent speech units," in *Proceedings Fourth International Conference on Spoken Language Processing*, vol. 3, (Philadelphia, PA, USA), pp. 1808–1811, Oct. 1996.

[39] O. Andersen and P. Dalsgaard, "Language-identification based on cross-language acoustic models and optimised information combination," in *Proceedings 5th European Conference on Speech Communication and Technology*, vol. 1, (Rhodes, Greece), pp. 67–70, Sept. 1997.

[40] P. Dalsgaard, O. Andersen, and W. Barry, "Cross-language merged speech units and their descriptive phonetic correlates," in *Proceedings 5th International Conference on Spoken Language Processing*, vol. 6, (Sydney, Australia), pp. 2623–2626, Dec. 1998.

[41] K. M. Berkling, T. Arai, , and E. Barnard, "Analysis of phoneme-based features for language identification," in *Proceedings IEEE International Conference on Acoustics, Speech and Signal Processing 94*, vol. 1, (Adelaide, Australia), pp. 289–292, Apr. 1994.

[42] K. M. Berkling and E. Barnard, "Language identification with multilingual phoneneme clusters," in *Proceedings International Conference on Spoken Language Processing 94*, vol. 4, (Yokohama, Japan), pp. 1891–1894, Sept. 1994.

[43] K. M. Berkling and E. Barnard, "Theoretical error prediction for a language identification system using optimal phoneme clustering," in *Proceedings of the 4th European Conference on Speech Communication and Technology*, vol. 1, (Madrid, Spain), pp. 351–354, Sept. 1995.

[44] T. J. Hazen and V. W. Zue, "Automatic language identification using a segment-based approach," in *Proceedings 2nd European Conference on Speech Communication and Technology*, vol. 2, (Berlin, Germany), pp. 1303–1306, Sept. 1993.

[45] M. A. Zissman and E. Singer, "Automatic language identification of telephone speech messages using phoneme recognition and n-gram modeling," in *Proceedings IEEE International Conference on Acoustics, Speech and Signal Processing 94*, vol. 1, (Adelaide, Australia), pp. 305–308, Apr. 1994.

[46] Y. Yan and E. Barnard, "An approach to language identification with enhanced language model," in *Proceedings of the 4th European Conference on Speech Communication and Technology*, (Madrid, Spain), pp. 1351–1354, Sept. 1995.

[47] Y. Yan and E. Barnard, "An approach to automatic language identification based on language-dependant phone recognition," in *Proceedings IEEE International Conference on Acoustics, Speech and Signal Processing 95*, vol. 5, (Detroit, MI, USA), pp. 3511–3514, May 1995.

[48] T. J. Hazen and V. W. Zue, "Recent improvements in an approach to segment-based automatic language identification," in *Proceedings International Conference on Spoken Language Processing 94*, vol. 4, (Yokohama, Japan), pp. 1883–1886, Sept. 1994.

[49] C. Corredor-Ardoy, J. L. Gauvain, M. Adda-Decker, and L. Lamel, "Language identification with language-independent acoustic models," in *Proceedings 5th European Conference on Speech Communication and Technology*, vol. 1, (Rhodes, Greece), pp. 55–58, Sept. 1997.

[50] K. M. Berkling and E. Barnard, "Language identification with inaccurate string matching," in *Proceedings Fourth International Conference on Spoken Language Processing*, vol. 3, (Philadelphia, PA, USA), pp. 1796–1799, Oct. 1996.

[51] E. S. Parris, H. Lloyd-Thomas, M. J. Carey, and J. H. Wright, "Bayesian methods for language verification," in *Proceedings 5th European Conference on Speech Communication and Technology*, vol. 1, (Rhodes, Greece), pp. 59–63, Sept. 1997.

[52] H. Lloyd-Thomas, E. S. Parris, and J. H. Wright, "Recurrent substrings and data fusion for language recognition," in *Proceedings 5th International Conference on Spoken Language Processing*, vol. 2, (Sydney, Australia), pp. 169–172, Dec. 1998.

[53] J. Navratil and W. Zuhlke, "Phonetic-context mapping in language identification," in *Proceedings 5th European Conference on Speech Communication and Technology*, vol. 1, (Rhodes, Greece), pp. 71–74, Sept. 1997.

[54] S. Kadambe and J. L. Hieronymus, "Language identification with phonological and lexical models," in *Proceedings IEEE International Conference on Acoustics, Speech and Signal Processing 95*, vol. 5, (Detroit, MI, USA), pp. 3507–3510, May 1995.

[55] S. Itahashi and L. Du, "Language identification based on speech fundamental frequency," in *Proceedings IEEE International Conference on Acoustics, Speech and Signal Processing 95*, vol. 2, (Detroit, MI, USA), pp. 1359–1362, May 1995.

[56] S. E. Hutchins and A. E. Thyme-Gobbel, "Experiments using prosody for language identification," in *Proceedings Speech Research Symposium XIV*, (Baltimore, Maryland), June 1994.

[57] A. E. Thyme-Gobbel and S. E. Hutchins, "On using prosodic cues in automatic language identification," in *Proceedings Fourth International Conference on Spoken Language Processing*, vol. 3, (Philadelphia, PA, USA), pp. 1768–1771, Oct. 1996.

[58] R. B. Ives, "A minimal rule AI expert system for real-time classification of natural spoken languages," in *Proceedings 2nd Annual Artificial Intelligence and Advanced Computer Technology Conference*, (Long Beech, CA), pp. 337–340, May 1986.

[59] K. P. Li, "Automatic language identification using syllabic spectral features," in *Proceedings IEEE International Conference on Acoustics, Speech and Signal Processing 94*, vol. 1, (Adelaide, Australia), pp. 297–300, Apr. 1994.

[60] J. L. Hieronymous and S. Kadambe, "Spoken language identification using large vocabulary speech recognition," in *Proceedings Fourth International Conference on Spoken Language Processing*, vol. 3, (Philadelphia, PA, USA), pp. 1780–1783, Oct. 1996.

[61] T. Schultz, I. Rogina, and A. Waibel, "LVCSR-based language identification," in *Proceedings IEEE International Conference on Acoustics, Speech and Signal Processing 96*, (Atlanta, GA, USA), pp. 781–784, May 1996.

[62] S. Mendoza, L. Gillick, Y. Ito, S. Lowe, and M. Newman, "Automatic language identification using large vocabulary continuous speech recognition," in *Proceedings IEEE International Conference on Acoustics, Speech and Signal Processing 96*, (Atlanta, GA, USA), pp. 785–788, May 1996.

[63] H. Kwan and K. Hirose, "Unknown language rejection in language identification system," in *Proceedings Fourth International Conference on Spoken Language Processing*, vol. 3, (Philadelphia, PA, USA), pp. 1776–1779, Oct. 1996.

[64] H. Kwan and K. Hirose, "Use of recurrent network for unknown language rejection in language identification system," in *Proceedings 5th European Conference on Speech Communication and Technology*, vol. 1, (Rhodes, Greece), pp. 63–66, Sept. 1997.

[65] D. Matrouf, M. Adda-Decker, L. F. Lamel, and J. L. Gauvain, "Language identification incorporating lexical information," in *Proceedings 5th International Conference on Spoken Language Processing*, vol. 2, (Sydney, Australia), pp. 181–184, Dec. 1998.

[66] E. Barnard and Y. Yan, "Towards new language adaption for language identification," *Speech Communication*, vol. 21, pp. 245–254, 1997.

[67] D. A. Reynolds, E. Singer, B. A. Carlson, G. C. O'Leary, J. J. McLaughlin, and M. A. Zissman, "Blind clustering of speech utterances based on speaker and language characteristics," in *Proceedings 5th International Conference on Spoken Language Processing*, vol. 7, (Sydney, Australia), pp. 3193–3196, Dec. 1998.

[68] Y. Yan and E. Barnard, "Experiments with conversational telephone speech for language identification," in *Proceedings IEEE International Conference on Acoustics, Speech and Signal Processing 96*, (Atlanta, GA, USA), pp. 789–792, May 1996.

[69] K. Berkling, M. Zissman, J. Vonwiller, and C. Cleirigh, "Improving accent identification through knowledge of English syllable structure," in *Proceedings 5th International Conference on Spoken Language Processing*, vol. 2, (Sydney, Australia), pp. 89–92, Dec. 1998.

[70] L. R. Yanguas, G. C. O'Leary, and M. A. Zissman, "Incorporating linguistic knowledge into automatic dialect identification of Spanish," in *Proceedings 5th International Conference on Spoken Language Processing*, vol. 2, (Sydney, Australia), pp. 237–240, Dec. 1998.

[71] K. Kumpf and R. W. King, "Foreign speaker accent classification using phoneme-dependent accent discrimination models and comparisons with human perception benchmarks," in *Proceedings 5th European Conference on Speech Communication and Technology*, vol. 5, (Rhodes, Greece), pp. 2323–2326, Sept. 1997.

[72] R. H. Robins, *General Linguistics: An Introductory Survey*. Longmans' Linguistics Library, Longmans, Green and Co. Ltd., 1968.

[73] J. Lyons, *Introduction to Theoretical Linguistics*. Cambridge University Press, 1968.

[74] R. Burling, *Patterns of Language: Structure, Variation, Change.* Academic Press, Inc., 1992.

[75] A. Akmajian, R. A. Demers, and R. M. Harnish, *Linguistics: An Introduction to Language and Communication.* The MIT Press, second ed., 1984.

[76] S. S. Stevens, J. Volkmann, and E. B. Newman, "A scale for the measurement of the psychological magnitude pitch," *Journal of the Acoustical Society of America*, vol. 8, pp. 185–190, Jan. 1937.

[77] S. S. Stevens and J. Volkmann, "The relation of pitch to frequency: A revised scale," *American Journal of Psychology*, vol. 53, pp. 329–353, July 1940.

[78] H. Fletcher and W. A. Munson, "Loudness, its definition, measurement and calculation," *Journal of the Acoustical Society of America*, vol. 5, pp. 82–108, Oct. 1933.

[79] H. Fletcher and W. A. Munson, "Relation between loudness and masking," *Journal of the Acoustical Society of America*, vol. 9, pp. 1–10, July 1937.

[80] H. Fletcher, "Auditory patterns," *Reviews of Modern Physics*, vol. 12, pp. 47–65, Jan. 1940.

[81] E. G. Shower and R. Biddulph, "Differential pitch sensitivty of the ear," *Journal of the Acoustical Society*, vol. 3, pp. 275–287, Oct. 1931.

[82] S. S. Stevens, "Calculation of the loudness of complex noise," *Journal of the Acoustical Society of America*, vol. 28, pp. 807–832, Sept. 1956.

[83] E. Zwicker, G. Flottorp, and S. S. Stevens, "Critical bandwidth in loudness summation," *Journal of the Acoustical Society of America*, vol. 29, pp. 548–557, May 1957.

[84] T. H. Schafer, R. S. Gales, C. A. Shewmaker, and P. O. Thompson, "The frequency selectivity of the ear as determined by masking experiments," *Journal of the Acoustical Society of America*, vol. 22, pp. 490–496, July 1950.

[85] J. E. Hawkins, Jr. and S. S. Stevens, "The masking of pure tones and of speech by white noise," *Journal of the Acoustical Society of America*, vol. 22, pp. 6–13, Jan. 1950.

[86] E. Zwicker, "Subdivision of the audible frequency range into critical bands (Frequenzgruppen)," *Journal of the Acoustical Society of America*, vol. 33, p. 248, Feb. 1961.

[87] E. Zwicker and E. Terhardt, "Analytical expressions for critical-band rate and critical bandwidth as a function of frequency," *Journal of the Acoustical Society of America*, vol. 68, pp. 1523–1525, Nov. 1980.

[88] N. R. French and J. C. Steinberg, "Factors governing the intelligibility of speech sounds," *Journal of the Acoustical Society of America*, vol. 19, pp. 90–119, Jan. 1947.

[89] J. S. Bridle and M. D. Brown, "An experimental automatic word recognition system," Tech. Rep. JSRU Report No. 1003, Joint Speech Research Unit, Ruislip, England, 1974.

[90] P. Mermelstein, "Distance measures for speech recognition - psychological and instrumental," in *Pattern Recognition and Artificial Intelligence* (C. H. Chen, ed.), pp. 374–388, Academic Press, Inc., 1976.

[91] S. B. Davis and P. Mermelstein, "Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 28, pp. 357–366, Aug. 1980.

[92] C. R. Jankowski Jr., H.-D. H. Vo, and R. P. Lippmann, "A comparison of signal processing front ends for automatic word recognition," *IEEE Transactions on Speech and Audio Processing*, vol. 3, pp. 286–293, July 1995.

[93] S. Sandhu, "A comparative study of mel cepstra and EIH for phone classification under adverse conditions," Master's thesis, Massachusetts Institute of Technology, Feb. 1995.

[94] L. Rabiner and B.-H. Juang, *Fundamentals of Speech Recognition*, section 4.5.6, pp. 183–190. In *Prentice Hall Signal Processing Series* [1], 1993.

[95] N. Merhav and C.-H. Lee, "On the asymptotic statistical behaviour of empirical cepstral coefficients," *IEEE Transactions on Signal Processing*, vol. 41, pp. 1990–1993, May 1993.

[96] L. C. W. Pols, *Spectral Analysis and Identification of Dutch Vowels in Monosyllabic Words.* PhD thesis, Free University, Amsterdam, 1966.

[97] J. F. Blinn, "What's the deal with the DCT?," *IEEE Computer Graphics and Applications*, vol. 13, pp. 78–83, July 1993.

[98] L. Rabiner and B.-H. Juang, *Fundamentals of Speech Recognition*, section 4.6, pp. 194–200. In *Prentice Hall Signal Processing Series* [1], 1993.

[99] D.-I. Choi and S.-H. Park, "Self-creating and organizing neural networks," *IEEE Transactions on Neural Networks*, vol. 5, pp. 561–575, July 1994.

[100] T. Kohonen, *Self Organizing Feature Map*, chapter 5, pp. 119–157. New York: Springer-Verlag, second ed., 1988.

[101] Y. Linde, A. Buzo, and R. M. Gray, "An algorithm for vector quantizer design," *IEEE Transactions on Communications*, vol. 28, pp. 84–95, Jan. 1980.

[102] L. R. Rabiner, "An introduction to hidden Markov models," *IEEE Acoustics and Signal Processing Magazine*, pp. 4–16, Jan. 1986.

[103] L. R. Rabiner, "A tutorial on hidden Markov models and selected applications in speech recognition," *Proceedings of the IEEE*, vol. 77, pp. 257–286, Feb. 1989.

[104] H. F. Silverman and D. P. Morgan, "The application of dynamic programming to connected speech recognition," *IEEE Acoustics and Signal Processing Magazine*, pp. 6–25, July 1990.

[105] J. Picone, "Continuous speech recognition using hidden Markov models," *IEEE Acoustics and Signal Processing Magazine*, pp. 26–41, July 1990.

[106] J. A. du Preez and D. M. Weber, "Efficient high-order hidden Markov modelling," in *Proceedings 5th International Conference on Spoken Language Processing*, vol. 7, (Sydney, Australia), pp. 2911–2914, Dec. 1998.

[107] L. Rabiner and B.-H. Juang, *Fundamentals of Speech Recognition*, chapter 6, pp. 321–389. In *Prentice Hall Signal Processing Series* [1], 1993.

[108] L. Rabiner and B.-H. Juang, *Fundamentals of Speech Recognition*, section 6.2–6.4, pp. 322–348. In *Prentice Hall Signal Processing Series* [1], 1993.

[109] "NIST ALI evaluation," 1996. URL: ftp://jaguar.ncsl.nist.gov/lid96.

[110] B. P. Strope, "A model of dynamic auditory perception and its application to robust speech recognition," Master's thesis, University of California, 1995.

[111] J. Rissanen, "Stochastic complexity in statistical inquiry," *Singapore: World Scientific.*

[112] C. E. Shannon, "A mathematical theory of communication," *Bell System Technical Journal*, vol. 27, pp. 379–423, 623–656, 1948.

[113] D. J. Field, *Visual Coding, Redundancy, and 'Feature Detection'*, pp. 1012–1016. The MIT Press, 1995. Part III: Articles.

[114] B.-H. Juang, W. Chou, and C.-H. Lee, "Minimum classification error rate methods for speech recognition," *IEEE Transactions on Speech and Audio Processing*, vol. 5, May 1997.