PROSPECTS FOR APPLYING SPEAKER VERIFICATION TO UNATTENDED SECURE BANKING

Malcolm Ian Hannah

A THESIS SUBMITTED IN PARTIAL FULFILMENT OF THE REQUIREMENTS OF THE UNIVERSITY OF ABERTAY DUNDEE FOR THE DEGREE OF DOCTOR OF PHILOSOPHY

This research project was supported by the Technology Development Group at NCR

November 1996

I certify that this thesis is the true and accurate version of the thesis approved by the examiners.

Signed ....[redacted].... Date .24. July 1997

ies)

Abstract

Using voice characteristics to verify identity is an emerging science, reporting ever-lower error rates. This project investigates the application of this technology to unattended secure banking, such as automated teller machines. The aim was not to produce a final system but to create a verifier with parameters which could be varied to investigate the effects. Following Furui (1981) and Bernasconi (1990), the text-dependent, dynamic time-warping (DTW) platform was chosen.

To help evaluate changes to the verifier configuration, methods of assessment based on the separation between genuine and impostor DTW (dissimilarity) score distributions are proposed. These offer alternatives to the often-quoted but sometimes uninformative equal-error rate. A technique of generating speaker-specific but globally adjustable thresholds is presented for occasions when error rates are of interest. Also, two databases, one with over 200 speakers, have been collected.

Enrolling customers record several versions of the same word or phrase and these are used to make a characteristic template. Four different approaches are examined: two keep all of these initial tokens separate but the performance gain over the other two (combination) methods of Bernasconi and Furui – whose approach suffers from arbitrary treatment of initial data – does not merit the extra computation when verifying.

As expected, long utterances are found to work better than short ones, up to a point, probably about 2 seconds. However, combining DTW scores for sequences of individual short words may be as effective.

Traditional techniques for weighting the distance measure treat each vector dimension in isolation but by considering their combined effect on the separation between score distributions, when generated either by genetic algorithm or purely randomly, great gains may be made.

In practice, a central database of speech would be necessary to calibrate such a system for each enrolling user. Also, to provide high security, the text to be spoken should consist of a sequence of randomly-chosen short words, such as the digits. Personalised modification of the input speech through the use of weighting functions is found to stabilise the verifier performance and reduce error rates to a level likely to be acceptable to financial institutions.

ACKNOWLEDGEMENTS

Thanks to:

- the supervisory team of Andy Sapeluk and Bob Damper,
- lab-mates Andy Hughes and Robert 'Chuck Geldof' Finan,
- Fiona Steers for help with phonetic transcriptions,
- Brenda Keir, Jacqui Pirie, Karen Meek, Tracy More, Andy Facey, Andy Macdonald, Neil Taylor, Craig Garvie, Fergus MacLean, Darren Steers and Alistair Douglas for donating their voices to science,
- all the technical staff who prepared the speech recordings,
- Sam Flynn and Ian M. Roger of NCR,
- my parents,
- mmmmmmMary-Kate.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

Speaker verification is the process of verifying a person's identity using only their voice characteristics. Since the 1930's, the general topic of speaker recognition has been investigated in various forms and with varying degrees of success. Initially motivated by forensic science, attempts were made to produce a visual representation of a voice (a 'voiceprint') which could then be compared with another by an 'expert'. The implied comparison with fingerprints is not valid because fingerprint patterns do not vary with age and no cases of duplication have been reported. In contrast, current methods of characterising voices have never been proven to be unique and every system subjected to long-term tests has been susceptible to error due to a significant variation in each individual's voice. This project investigates automatic (computer-based) speaker verification (ASV) as a method of securing unattended banking transactions, probably based on automated teller machines (ATMs).

Personal identification numbers (PINs) are currently used as the security measure in ATMs. Often, this causes problems, both for customers and for the banks because PINs are easily forgotten, especially if a customer has several to remember. The banks are (privately) very concerned by apparently unauthorised 'phantom' withdrawals and seek to complement or replace the PIN with another form of identification. Biometric measures truly represent identity and cannot be transferred from person to person so tests of hand geometry, patterns of palm veins, retinal scans and signature verification have all been proposed but have been generally rejected as impractical or unlikely to be accepted by customers.

Over a third of the world's ATMs are produced in the Dundee manufacturing facility of NCR and, aware of the potential market for machines with extra security, they provided funding for this project. The aim was not necessarily to produce a speaker verification system to be incorporated into their machines but to find out about the mechanism by which such systems work and the feasibility of using such technology in the near future. However, many of the obstacles to a high-street implementation of ASV are not covered in this report. Noisy traffic and passers-by create a cluttered and unpredictable background sound level and an investigation into effective pre-processing in itself would merit a Ph.D. Also, the effects of ageing and the extent to which a speaker's psycho-physical state (illness, stress or intoxication) is reflected by voice condition are beyond the scope of this project.

Structure of the thesis

Each chapter describes an individual element of the development or testing. Chapter 2 presents a review of current ASV technology to provide a backdrop against which this work may be viewed, and introduces the elements of the system used for testing in later chapters. Chapter 3 describes the speech database used in these tests and the verification system itself is described in detail in Chapter 4.

Several methods exist for assessing the performance of a speaker verification system and these are defined in Chapter 5 along with new methods developed for this project. These assessment methods allow detailed treatment in Chapters 6, 7 and 8 of experiments relating to utterance length and content, reference template construction and discrimination optimisation respectively. Chapter 9 brings together the most positive features of these experimental systems to describe an optimised but practical verifier before Chapter 10 concludes.

Chapter 2

Review of speaker verification technology

Since the early 1960s published reports have become increasingly common about the broad area of speaker recognition. Speaker *identification* (SI) determines which of a closed set of candidates is speaking; speaker *verification* (SV) is a simpler task where a specific identity is claimed and the decision to accept or reject the claim is based on the speaker's voice. Compared to SV, the potential applications of SI are fewer and less commercial but most of the developed systems could be modified easily to perform either task. For this reason, this chapter reviews the development of both technologies referring to published reports and commercial products. This provides a context in which to view the findings of later chapters.

2.1 Literature survey

When reading the literature it is difficult to make meaningful comparisons between systems because there are no standard procedures for testing and reporting results. This is not surprising because there are so many variables in any system, such as quality of the speech signal, number of enrolled speakers, number of trials conducted, quantity of data for training and testing, parameters used to represent the speech and the time taken to make a decision. These are all important factors in determining the usefulness of a system.

2.1.1 Speech features

Most recognition systems work with features extracted from the speech waveform, comparing those of an unknown speaker with feature templates constructed during enrollment of registered users. The dissimilarity of the two is quantified and, in the case of verification, determines whether or not the claimant is accepted or rejected.

The speech features, in almost all of the reported strategies, are obtained by frequency-domain analysis, due to the way in which speech is produced. Air flows through the vocal tract, which begins at the vocal cords, ends at the lips and includes all the articulators (such as the tongue and teeth) in between. The vocal tract acts as a filter with resonances and anti-resonances according to the dimensions of the throat and the positions of the articulators. Since these are physiological factors, usually beyond conscious control, they are thought to be specific to each person. Thus, the frequency spectrum of an utterance should contain indications of the speaker's identity, as well as the text spoken. Intuitively, averaging spectra over several phonetically varied sentences should highlight characteristics of the speaker rather than the text. Early attempts at speaker recognition (Pruzansky 1963, Hollien and Majewski 1977) operated on this principle but generally gave way to analysis of shorter periods (*frames*), to improve system response times.

Several methods of estimating the speech spectrum (which is concentrated mostly in the 50–5000 Hz range) are available. The most direct method is filter-bank analysis where the speech is applied to a parallel array of band-pass filters, with each band covering successive frequency ranges. Considered together, the outputs of the filters form a histogram, showing the relative concentration of speech energy across the spectrum. The centre-frequencies of the filters are sometimes spaced linearly (Doddington 1985) but often use non-linear scales such as mel (Rose, Fitzmaurice, Hofstetter and Reynolds 1991) or Bark to match the sensitivities of the human ear. Generally, no other allusions to human hearing have been made until recently: Colombi *et al* (1993) and Anderson and Patterson (1994) extracted voice features using models of the human auditory system (specifically, the behaviour of the basilar membrane) but found the potential rewards are currently outweighed by the extra computational burden.

Short-time Fourier transforms (including cosine and sine transforms) offer very detailed spectral estimates and efficient software implementations have been developed. However, analysis by linear predictive coding (LPC) is more common, as it ignores much of the redundancy inherent in speech whilst representing efficiently the principal features. LPC models only the vocal tract and not the excitation source (glottal wave). A further transformation leads to cepstral coefficients which have been shown to be superior for recognition (Atal 1974). Cepstral parameters can also be obtained via the Fourier transform (Rudasi and Zahorian 1991). The cepstrum represents a greatly smoothed version of the spectrum but on a logarithmic scale, thus emphasising peaks and troughs. Also, the logarithm allows deconvolution of frequencies by subtraction and this property is often used to remove the invariant channel characteristics to make the system less context-sensitive.

2.1.2 Comparison methods

Verification is usually a process of comparing the speech features from an unknown speaker with those of the claimed identity. However, since the features are often only representative of a short excerpt of speech, comparison over an entire utterance is not straightforward. Many of the methods used in speaker recognition have a history of application in word recognition problems. Li and Wrench (1983) first applied the technique of vector quantisation (VQ) to speaker identification. The features extracted from speech are stored inexactly as the vectors in a small set or *codebook* which are closest to those actually uttered. Soong, Rosenberg, Rabiner and Juang (1985) performed more extensive tests using VQ codebooks for short utterances (the digits) and found that 10 different digits offered better discrimination than 10 repetitions of the same digit. The best digit was *nine*, due to the nasal coarticulation, which is hard to modify consciously. A later study (Soong and Rosenberg 1986) found that information about how the cepstral coefficients changed from frame to frame could be combined with their instantaneous values for improved performance.

For text-dependent verification, the method of template matching is appropriate, where input speech is compared with a model of the correct word spoken by the claimed identity. Unfortunately, there can be considerable variation in the way someone says a word or phrase, noticeable even in two consecutive utterances. Other than changes in inflection or pitch, subtle timing differences (which are not constant throughout an utterance) contribute most to this variation. The technique known as *dynamic time-warping* (DTW) has been developed (Sakoe and Chiba 1978) which allows a localised tolerance in the matching of corresponding portions of utterances. In effect, the time-axis of one utterance undergoes a series of localised compressions and expansions to match the time-axis of the other utterance. These axis alterations are defined by a *warp-path*, along which the local distances (between the feature vectors for corresponding frames) may be accumulated to give a global distance, or difference between the utterances.

Although used mostly for word recognition (Myers, Rabiner and Rosenberg 1981), DTW has also been applied successfully to speaker verification, first by Furui (1981). Velius (1988) and Bernasconi (1990) used DTW systems as a basis for investigating properties of cepstral coefficients.

Hidden Markov models (HMMs) have also been associated with word recognition but increasingly are being applied to speaker recognition, both text-dependent (Zheng and Yuan 1988, Netsch and Doddington 1992) and text-independent (de Veth and Bourlard 1994). The HMM approach can be likened to a (less constrained) DTW comparison of a test utterance (as a sequence of feature vectors) with a compressed reference template for the same word, consisting of only a single vector for each sub-word unit. Grant (1991) gives a good introduction to HMMs and DTW.

The connectionist approach to speaker recognition using artificial neural networks (ANNs) is comparatively new, first proposed by Oglesby and Mason (1988). Their results for text-dependent speaker identification were comparable with a VQ system (Oglesby and Mason 1989). Information about the general population or a confusible subset (Oglesby and Mason 1990) is included in each speaker's model so that areas of overlap can be ignored and speaker-specific information can be highlighted. Bennani and Gallinari (1991) combined several neural networks with each assigned a sub-task of the overall identification, i.e. an unknown speaker is first classified by gender then dialect. This modular approach makes each task easier and reduces the training time of each network.

2.2 Commercial survey

A search of the appropriate databases[1] for patents relating to speaker verification revealed that many of the leading electronics/communications companies in the

[1]Conducted in 1992 through Dialog Information Systems.

world have worked in this area yet few have made any systems commercially-available. This section describes companies who have worked on a commercial product and have reinforced their marketing material with published papers. (A more comprehensive listing of interested companies may be found in Hannah (1992).)

2.2.1 AT&T

AT&T Bell Laboratories have been studying speaker recognition since the early 1970's and have been associated with published papers covering almost every technique. Recently the HMM-based system described by Rosenberg, Lee and Gokcen (1991) was incorporated into an automatic teller machine and the results of initial field trials were reported by Jacobs and Setlur (1994). Each user was prompted to speak a randomly-selected 4-digit utterance and was allowed 3 attempts to match their own pre-recorded reference material. Testing is on-going but the reported results are not extensive enough to determine the feasibility of the system. A major factor to be overcome is the hardware: currently the system uses a dedicated digital signal processor and 1 MB of static memory.

2.2.2 British Telecom

British Telecom supplied the Royal Bank of Scotland with speaker verification technology in 1988 and began testing it for use in their telephone banking system, which included word recognition also. Few technical details of the system are available but field trials conducted by the bank in 1991 concluded that enrolled customers were rejected too frequently for use with the general public. However, the system was considered to be successful for internal bank use by about 100 members of staff.

2.2.3 Ensigma

Ensigma Ltd. are a telecommunications consultancy who patented a speaker verification algorithm in 1990 and developed the system described by Carey, Parris and Bridle (1991) into a product, available as software or on a plug-in card for a personal computer. The system uses two hidden Markov voice models: one for the enrolled customer and one for the general population. If the speech of an unknown user is closest to the world model then the speaker is rejected. In a pilot study using 6 speakers, verification typically took less than 10 seconds; users were prompted to say 5 randomly-selected digits and the results were encouraging.

Ensigma have not developed this product any further and do not have the resources to market it but it is currently being used by the Australian Prison Service for their 'home arrest' scheme.

2.2.4 Lernout & Hauspie

The Belgian company Lernout & Hauspie provide several different applications of voice technology and their speaker verification system is designed to work with strings of digits spoken over the telephone. The implementation uses HMMs and is available as software for a specific digital signal processor. Recently, the results of experimenting with different types of HMMs were reported (de Veth and Bourlard 1994).

2.2.5 Logica Cambridge

The speaker verification system is one of several telephony services offered by Logica and uses the Fourier-Bessel series to estimate the probability of DTW distances being attributable to the templates for a given speaker (Gaganelis and Frangoulis 1991). The system has been tested with 60 speakers, who were prompted to speak 4 words. In 1992, Logica were still experimenting with the system; since then, no further developments have been published.

2.3 Choosing a system to simulate

This project required the use of a verification system and it was decided to implement a DTW-based system following Bernasconi (1990), whose system was itself based on that of Furui (1981). Both of these studies highlighted many different possible configurations which could be investigated and modified. Also, it was important that a system with a realistic chance of eventual application should be studied, especially since the interest of the project sponsors was more than just academic. Both of the systems produced low error rates, which did not rise inexorably over time. A full description of the simulation is described in Chapter 4 and subsequent chapters discuss modifications with a view to improvements.

Chapter 3

Speech Data

Chapters 6, 7 and 8 describe experiments which test the system performance under various conditions. In each case, to make qualitative assessments, it was necessary to use the same input data. The absence of a suitable, available, recognised standard set of data forced the collection of local databases. A small database was formed for initial experiments but later, to test the extension of the initial findings to a wider population, another (larger) database was collected. This chapter describes the attempts to obtain a standard database and gives details of content, conditions and constructional difficulties of the locally-gathered databases.

3.1 Existing speech databases

When this project reached the stage of reporting results at conferences, an approach was made to the UK National Physical Laboratory (NPL) who maintained an archive of miscellaneous public-domain speech data (Goldsmith 1989). None was suitable for this project, having either too few speakers or only isolated words or not enough repetitions of each word by each speaker. At the time (1992), the NPL was trying to start collaboration amongst UK research establishments, commercial and academic, who were collecting their own speech databases independently, at considerable expense and effort, and probably not following a standard procedure. Recording enough utterances for each speaker to a consistent standard is difficult due to varying background noise, varying channel characteristics (if recording over telephone lines), and the unpredictable colouration of speech caused by varied emotional states. The NPL project known as SCRIBE was started with plans to record and document speech data at six or seven sites in the UK. Unfortunately, the envisaged standard, multi-purpose, computer-readable database was never developed, and the Speech Technology section of the NPL has since closed.

A similar fate has befallen a European initiative to define a standard assessment methodology for speaker verification systems. Funding for the project was diverted by ESPRIT to activities with more immediate commercial returns, although after the first year, a number of recommendations for a standard database were drawn up (Bimbot 1994).

In recent years, a number of databases have been made available; Godfrey, Graff and Martin (1994) review five databases in this category. In particular, the TIMIT database has been used in many published reports but was not available to this project in 1992.

3.2 Small database

The small database was collected by recording people who visited the research laboratory and consists of 5 males and 5 females saying the phrase *ABCDEFG* 20 times each. By chance, most of the speakers are from different places but none has a strong enough accent to suggest an overly influential role in the verification. The age range of the speakers is about 19–40.

Speech was captured using an in-house program which controlled a Texas Instruments TMS320C25-based card installed in a PC. The speech is transferred in real time to the PC's memory before being written to disk as a binary file, using an in-house format (described in Appendix A) hereafter referred to as IDF[2]. Although the card has many facilities for signal processing, only the analogue-to-digital converter chip (a Burr-Brown PCM75JG, 16-bit resolution) was used. All the signal-processing was implemented in software to minimise the unit cost of the verifier, which would be a primary consideration when applied to each machine in a network of ATMs.

Each speaker's contribution was recorded in a single session with a short interval half-way through to prevent speaker fatigue. All of the speakers were familiar with the project although only 3 had any knowledge of speech science. The timescale of the session was set by the speaker in that they controlled the

[2]IDF stands for integer data file, apparently.

start of each (fixed-duration) audio capture, although it was checked immediately by the author to ensure that the entire utterance had been recorded. The audio capture package was also used later by the author to remove silences preceding and following the actual speech. This *end-point detection* process may also be automated but at the risk of less accurate delineation of speech (see Chapter 4).

Speech technology systems generally compromise accuracy for processing speed. Both are equally important because immediacy is one of the attractions of a speech interface although there is little point if the speech is not interpreted correctly. This compromise is quantified usually by the specification of operating conditions, such as the required sound quality and the response time of the system.

Assessments of speech quality are based on the extent and detail of spectral features which are included (or *not excluded*) in the given bandwidth. Recordings covering 0–12 kHz would give hi-fi quality speech because virtually all the spectral information is included but even half of this bandwidth would give 'good' quality speech. Telephone-quality speech (covering approximately 300–3400 kHz) is usually intelligible for conversation despite the omission of the information needed to discriminate between certain sounds (such as /f/ and /s/). Human listeners are able to recognise voices on the telephone and many studies, such as Furui (1981) and Soong *et al* (1985) have demonstrated the principles of automatic speaker verification using similar restricted bandwidth speech.

For this project, an upper frequency limit of about 4 kHz was chosen originally, so that speech of a reasonable quality could be used. Sampling at 10 kHz would then satisfy the well-known *Nyquist criterion* in terms of completely describing the signal, but to eliminate the possibility of aliasing error the low-pass filter must effectively block frequencies approaching 5 kHz (Lathi 1989, pages 66–67). To this end, a 2-pole Butterworth filter was included in a purpose-built microphone pre-amplifier unit constructed using op-amps (Horowitz and Hill 1989, pages 272–276). The cut-off frequency of 3030 Hz ensured substantial attenuation at 5 kHz.

This is approaching the minimum bandwidth thought to be necessary for successful voice analysis (O'Shaughnessy 1987, page 256) and perhaps highlights the need for a different filter design. However, there are many standard methods each with advantages and disadvantages and choosing amongst them is a topic which occupies many volumes of electronic engineering texts. For this application, the Butterworth method is attractive because it offers a maximally-flat pass-band, thus minimising distortion of the spectral characteristics of a speaker's voice.

The recordings were made in a quiet, but not sound-proof, laboratory environment. In practice, the verifier could be expected to work in very non-deterministic noisy environments such as busy streets, train stations etc. but successful extraction of the speech from such a background signal merits a research project in itself! Consequently, the verifier in this study assumes 'clean' speech (with a high signal-to-noise ratio) as input. To aid in this, the microphone chosen was unidirectional (Tandy 33-984C, manufactured by Shure) and speakers were encouraged to go as close as was comfortably possible. It sometimes took quite an effort to ensure subjects maintained a steady distance from, and angle to, the microphone but the automatic scaling of the signal (described in Chapter 4) should minimise these effects.

3.3 Large database

The small database lacked speakers and utterances but was considered sufficient to demonstrate the effects of basic changes of the verifier's configuration. However, the results of experiments obtained with it left some degree of doubt about their applicability to a large population – a major consideration for this project. A temporary part-time employee was recruited to collect a much larger database which could also be used for future speech projects at UAD and NCR.

In order to allow the possibility of future recordings by these speakers, names and contact addresses were noted. The speakers were also asked their country of birth, first language, accent and voice condition, which most described as 'normal' although some mentioned having hay fever, asthma or a cold. The Human Factors group at NCR also took the opportunity to canvass the contributors on their feelings about using voice to control an ATM. This took the form of nine multiple-choice questions plus a space for 'other comments'. Finally, the speakers were asked if their occupation involved extensive use of the telephone and also to indicate their approximate age by ticking one of five ranges.

Recording sessions took place in a quiet laboratory in the University and in various rooms at the NCR plant, not guaranteed to be quiet. To encourage people to contribute to the database, the names were entered into a draw for two attractive prizes[3]. Over a period of several months, about 300 speakers were recorded. Most of them were students or staff at the University or NCR employees. Each recording consists of 140 utterances: 20 repetitions of a short phrase, plus 10 repetitions of each of 12 short words. The inclusion of the short phrase was to allow continuation of experiments with the single-phrase small database; the short words were for use in a different verification scheme where the system would prompt for individual words. The words used were the names of the digits i.e. *one, two,* $\cdots$, *nine*, plus *zero, oh* and *nothing*, so that the database could be used at a later date to test a word-recognition system, also for potential inclusion in an ATM. It was decided that the short phrase should be the name of some well-known person, so that the speakers would be familiar with it, without feeling self-conscious about saying it. The name *Sir Winston Churchill* was chosen, since it satisfied this criterion and also has some local historical significance[4]. Although this phrase contains 3 separate words, they were spoken as a continuous phrase in all observed cases.

Experience of collecting the small database suggested some changes in procedure. Instead of capturing directly to the computer, a Digital Audio Tape (DAT) recorder (SONY DTC-750), which samples at studio-quality 48 kHz and has built-in anti-alias filters, was used. This is advantageous because no bandwidth restrictions are imposed on the data at the time of recording; the speech could be used in many different experiments which might require different sampling rates and bandwidths. Also, an attachment was made which positioned a thin ring at a fixed distance (4 inches) from the front of the microphone, to act as a target 'mouthpiece'. These attempts to normalize the speaking distance were a response to suggestions that at close range, a dynamic microphone becomes sensitive to velocity, not pressure, resulting in occasional excessive plosive 'pop'. A corresponding normalization of speech amplitude did not occur but as mentioned before, this is compensated for by automatic scaling of the signal. For these recordings, a new dynamic microphone (AKG D190E) was obtained and a suitable interface to the DAT recorder was built around the HXP006 pre-amplifier module, supplied by ILP Electronics Ltd.

It was decided to record all 140 utterances in a single session, due to the

[3]A pair of active speakers, and a portable CD player.

[4]The people of Dundee were castigated by Winston Churchill when they failed to return him as sitting MP in 1922.

difficulties in scheduling, and finding people who would commit to, multiple sessions. The timescale of the recordings was set by a computer running a program which simply prompted for each word, with a specified delay between prompts, which could be changed at the speaker's request. A 10-second interval incorporating a count-down to the resumption was inserted half-way through the sequence. The whole procedure took 4 minutes on average, and was captured as a single continuous DAT recording. The computer display was chosen to try to put the speakers at ease; the screen was divided into two horizontal areas, the top half blue and the bottom half green, resembling abstractly a countryside view. The words to be spoken were displayed in the centre of the screen, following a predetermined random order, the same sequence for all speakers.

The tape and track references of each speaker's recording were stored on a database, along with the questionnaire results. After several months, 290 speakers had contributed. Each speaker's contribution was played back through a 120–4200 Hz band-pass filter into a TMS320C25-based card in a PC running Hypersignal. By sampling at 10 kHz, the entire 4 minute recording was captured directly to disk as a single file, in proprietary Hypersignal TIM format. A program was written which would read in portions of this file and display the waveform on the screen. A mouse could then be used to select the approximate end-points for each word. Teams of UAD technicians segmented each of the 290 TIM files into individual word-length IDF files, which were automatically named *xxx_y*, where *xxx* is a unique number (padded with leading zeros if necessary) given to each speaker, and *y* is the number of the utterance in the 140-long sequence. This naming convention allowed test programs to select automatically the file corresponding to a particular utterance by a specific speaker.

The disk capacity required to store all 140 files for each speaker is 4–5 MB. A partition of a 3 GB disk is used to accommodate all 290 speakers, with each speaker's files contained in a separate directory. This format allows further use of the database for future projects.

3.4 Critical appraisal of the large database

A major criticism of both databases is that they were recorded on a single-session basis. Several studies suggest that results obtained with such data will indicate an upper bound of performance, which would degrade with time (Bernasconi 1990, Furui 1974). Apart from the ageing process, there are many factors which influence inter-session vocal variations, such as emotional state, level of tiredness and time since eating or drinking. However, at this stage of the project, even an optimistic assessment of performance is useful to determine the value of proceeding further.

For this reason, it would be desirable to use the 'cleanest' possible speech for testing the verifier, despite the adversity of the expected operating conditions. To test robustness, varying levels of background noise could be added, but could not be removed. It was therefore a disappointment to find that many of the recordings contained extraneous material ranging from the sounds of closing doors (UAD recordings) to Tannoy announcements (NCR recordings). There are also a few instances of background conversation. In most cases, the speech signal is much stronger than the background noise and is assumed to be unaffected. It is possible that construction of a desk-top recording booth may have prevented these problems but the price per square metre of acoustic-deadening material put this option beyond the project's budget. Also, the original specification of the database equipment included the possibility of recording at several sites so portability was an issue.

The database should have been checked more frequently during its collection, although this is not always guaranteed to detect problems. At some of the data collection locations, the noise of the fan in the PC running the prompting program has been recorded on the tape. Investigations suggest that the noise is actually vibration 'conducted' by the desk and the microphone stand. Listening to the tapes *in situ* understandably failed to detect this phenomenon which was discovered by visual inspection of the signal. The noise looks like 50 Hz mains hum, which indeed is also present in some recordings, probably due to electrical equipment at NCR. The presence of some such constant frequency could falsely partition the data set, similar to the 'great divide' in the TIMIT database (Godfrey *et al* 1994) so when being captured to disk, a high-pass filter with a cut-off frequency of 120 Hz was applied. This attenuated the mains hum and its first harmonic, and the test results described in later chapters do not exhibit any data partitioning.

Before speaker verification could be applied to ATMs, tests would be necessary to ensure equal success rates for speakers of all accents and ethnic origins, to prevent offending true customers. However, in compiling the database, an emphasis was placed on collecting as many recordings as possible, regardless of accent etc. As a result, the speaker set is geographically focussed on Dundee/Angus although the cosmopolitan nature of both recording sites has ensured some miscellany. The verification task is made harder by this focus and in practice it is likely that potential impostors would have the same accent as their victim. The banks have indicated to NCR that, at present, most cash-card fraud takes place *within* social groups and families. This highlights another drawback of the database: all of the contributors spoke naturally and there were no deliberate attempts at impersonation. This would have to be investigated before a verification system could be implemented for public use.

Automatic prompting by computer has the advantages of being silent (supposedly!) without the risk of losing the place, as might happen if reading from a printed page. However, the speaker's perceived lack of control could be expected to add tension to an already unnatural situation. Again, this scenario would occur in the envisaged application but it would have been preferable to remove as many negative effects as possible at this stage of the research.

The DAT recorder provided a high-quality, manageable data-storage system. Unfortunately, the data gatherer was unaccustomed to its use and appears to have allowed single utterances to be erased occasionally. For some speakers, the 140th utterance is missing and a likely explanation is that when the 'stop' button is pressed after a recording, the tape unwinds slightly, and the subsequent recording overlaps.

The potential exists for computer control of the DAT recorder due to the ability to write machine-readable index codes on the tape. This could have saved hundreds of man-hours by automating the transfer of the 4-minute recordings on tape to the 140 word-length files on disk. Unfortunately, the only facility for remote control of the DAT recorder is the infra-red handset. Construction and programming of a computer peripheral device to emulate these signals was beyond the scope of this project.

When the recordings were being segmented into individual files, the technicians noted those most prominently affected by any of the problems mentioned above. Rather than checking every file to see what could be salvaged, it was decided to limit the test data to that produced by speakers whose 140 utterances were reported to be error-free. This reduced the effective speaker set to 214: 148 males and 66 females. A summary of both databases is given in Table 3.1.

	Database	
	small	large
Number of speakers	10	214
Males/females	5/5	148/66
Bandwidth	0–3 kHz	120–4200 Hz
Sampling rate	10 kHz	10 kHz
Environment	quiet lab	varied
End-point detection	fine	coarse
ABCDEFG	20	0
Sir Winston Churchill	-	20
one	-	10
two	-	10
three	-	10
four	-	10
five	-	10
six	-	10
seven	-	10
eight	-	10
nine	-	10
zero	-	10
oh	-	10
nothing	-	10

Table 3.1: Characteristics and content of the large and small speech databases used in this project.

Chapter 4

Verifier Operation

As mentioned in Chapter 2, the system described by Bernasconi (1990) seemed suitable for an investigation into the effects of modifying certain parameters, common to many verifiers. The design is basically a pattern-matching procedure applied to two feature sets. One set is extracted from the speech of an unknown user and the other is a *reference template* for the registered customer whose identity is being claimed. This template is itself constructed using speech recorded in an enrollment session; several methods for this are described in Chapter 7.

Figure 4.1 illustrates the verification procedure. DTW (dynamic time-warping) produces a single *score* which reflects the level of mismatch between the test and reference feature patterns. The features used are cepstral coefficients derived from LPC (linear predictive coding) of short segments of speech. In a practical system, the score would be passed to some decision-making mechanism along with a pre-determined threshold, and the unknown speaker would be accepted or rejected. However, reliably determining the threshold is very difficult and is beyond the scope of this project, which seeks to demonstrate optimisations for potential – not final – speaker verification systems. Instead, ways of interpreting the score and assessing potential error rates are discussed in Chapter 5.

This chapter describes each stage of the verification process in detail, beginning with the treatment of a captured utterance to make it suitable for feature analysis.

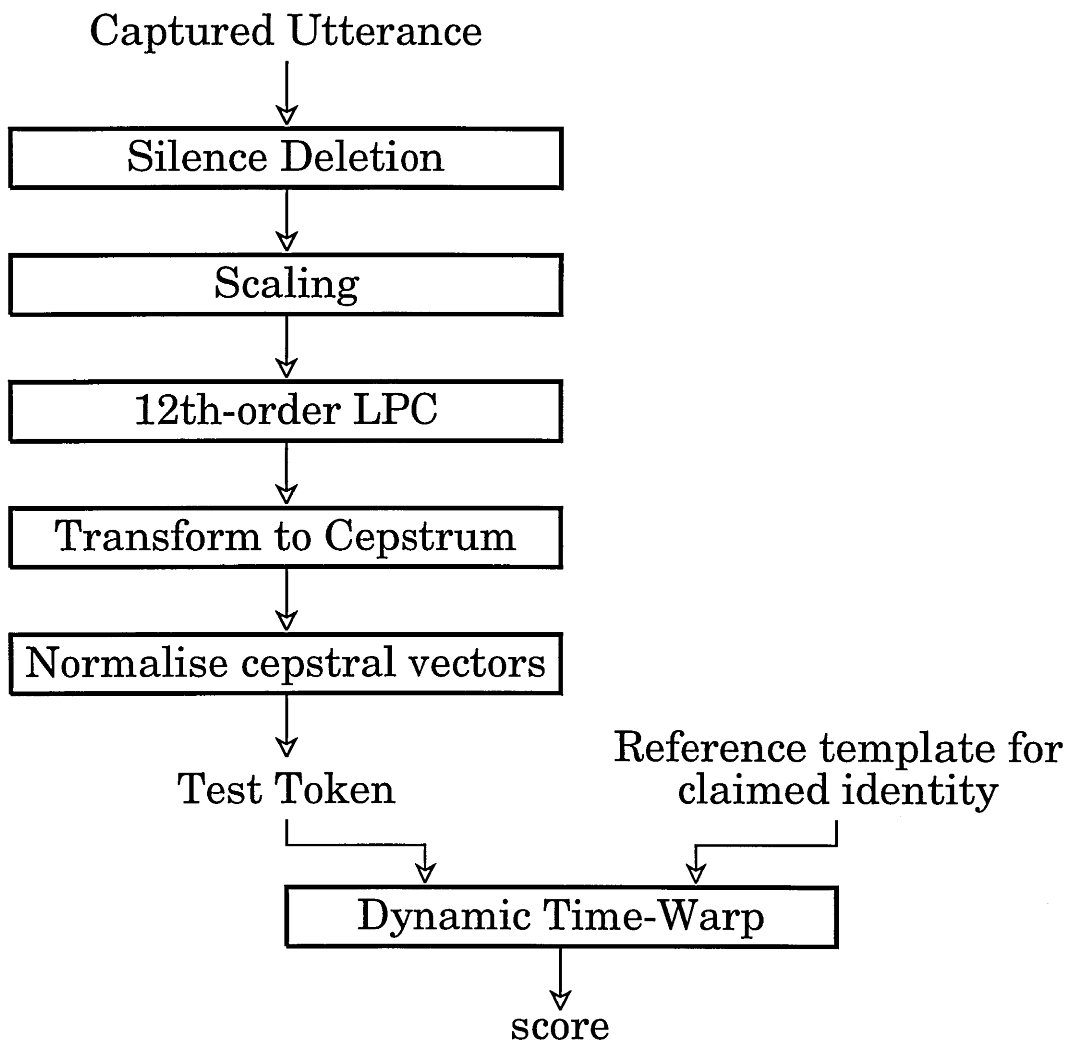

Figure 4.1: The processes of the verifier used in this project.

4.1 Silence deletion

As mentioned in Chapter 3, a captured utterance exists as a computer file in IDF format, which is simply a 16-byte header (with duration information and the sampling rate) followed by the raw data. The data are stored as a sequence of signed integers, equi-spaced samples of the input signal. In most cases, these data contain not only the speech but also signals preceding and following the speech; technically this is background noise although it is often referred to as 'silence'. This is due to the speech-acquisition process which must start slightly before, and continue slightly after, the actual utterance to ensure that the complete text is captured.

To prevent wasteful analysis of non-speech portions of the captured signal, an attempt is made to locate the start and finish of the speech within the data sequence. Accurate *end-point detection* is also important for correct time-alignment of test and reference tokens in the DTW stage, which is central to accurate verification. End-point detection is a special case of *silence deletion*, which removes portions of silent data anywhere in the captured utterance, for example, between words. In both cases, the signal is analysed in short segments (groups of samples) usually about 10–20 ms long. Each segment is classified as speech or non-speech according to certain of its characteristics and (sometimes) those of neighbouring segments. The task of silence deletion is deceptive in its apparent simplicity: over the course of this project, several different approaches have been tried with varying levels of complexity and success.

4.1.1 Multi-threshold technique

The strongest indicator of speech or non-speech is the instantaneous signal magnitude, provided that the recordings exhibit a high signal-to-noise ratio. This was true of the earliest recordings used in this project, which were highly supervised, and a correspondingly simple algorithm was developed based on thresholding the energy profile of the data. Similar, more sophisticated methods are described by Lamel, Rabiner, Rosenberg and Wilpon (1981) and Dermatas, Fakotakis and Kokkinakis (1991).

The energy E of a finite segment of the signal is defined (O'Shaughnessy 1987, page 42) as:

$$E = \sum_{i=1}^{N} {x_i}^2 \tag{4.1}$$

where x_i is the ith of N samples in the segment. The squaring operation prevents the addition of negative values to what is essentially an accumulation of the amplitudes of samples in the segment. A similar measure, the average amplitude A, is preferred because it uses no multiplication and is therefore faster to compute.

$$A = \frac{1}{N} \sum_{i=1}^{N} | x_i | \tag{4.2}$$

With $N = 100$ and sampling at 10 kHz, each segment is 10 ms, short enough for precise location of end-points but long enough to include contextual information.

For example, a segment consisting mostly of low-level samples would probably not be corrupted by a few high-level samples due to a noise spike.

The method is illustrated in Figure 4.2. The average amplitude for each segment is calculated and placed in an array, which gives an approximate profile of the signal energy over the entire utterance. A coarse estimate of the start-point of the speech is obtained by working forward through the array until the *major* threshold is exceeded; this can then be adjusted finely by working backwards until A falls below the *minor* threshold. All of the segments preceding this point can then be discarded. The combination of thresholds helps to prevent breath noise corrupting the delineation of the speech. A similar process is applied to the end of the utterance, but working backwards then forwards.

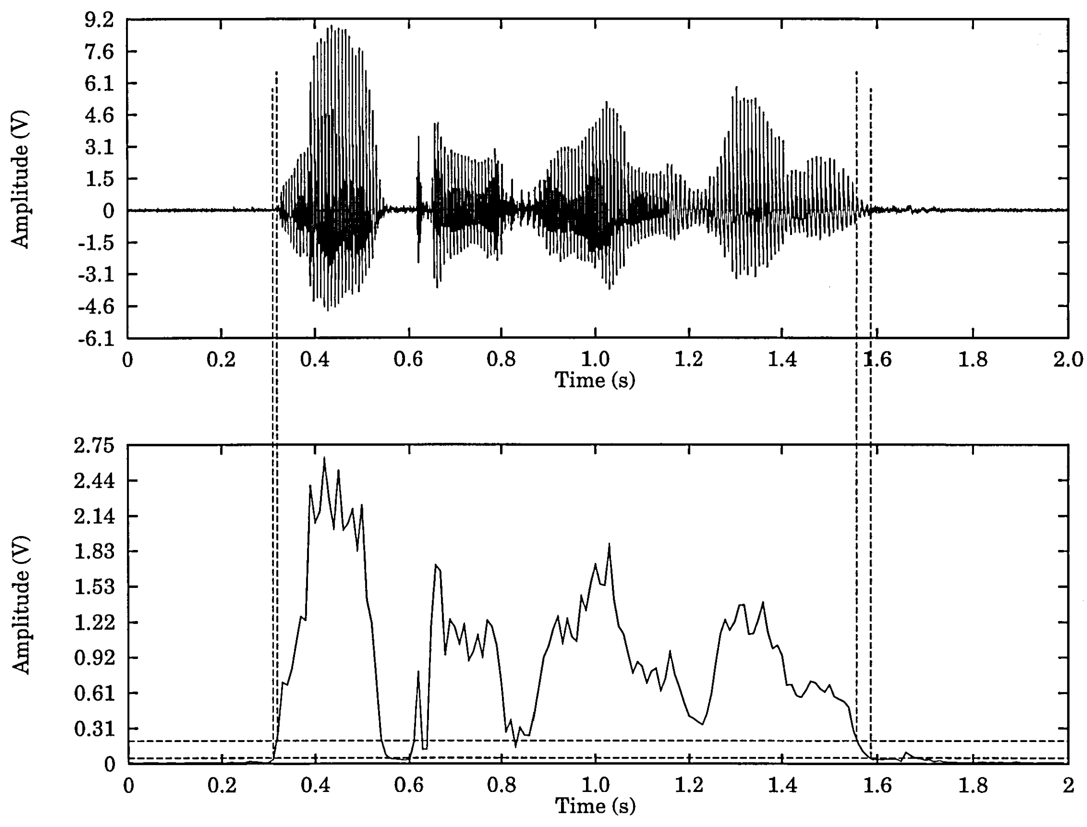

Figure 4.2: The multi-threshold method was used in the initial demonstration versions of the verifier. The text was the speaker's full name, in this case *Malcolm Ian Hannah*. The lower plot shows the contour of average amplitude and broken horizontal lines show the major and minor thresholds. The broken vertical lines project the detected end-points onto the plot of the actual utterance.

The major threshold T_{major} is calculated using one of two different criteria, depending on the nature of the data.

$$T_{\text{major}} = \begin{cases} \frac{1}{3}\bar{A} & \bar{A} > 183 \text{ mV} \\ \frac{1}{6}x_{\text{max}} & \text{otherwise} \end{cases} \tag{4.3}$$

where $\bar{A}$ is the mean value of A, and x_{max} is the maximum sample value observed, over the entire recording. In both cases, the minor threshold $T_{\text{minor}} = \frac{1}{4}T_{\text{major}}$.

Exclusively using one or other of these criteria to determine T_{major} produced inconsistent results. Checking that $\bar{A} > 183$ mV ensures that $\bar{A}$ is calculated mostly from speech rather than background noise, which could arise if a person spoke too quickly, softly or far from the microphone. The thresholds and relationships between them were determined empirically with fairly 'clean' speech. Predictably, such a simple approach does not perform well with speech recorded with less supervision where artifacts such as mouth clicks and 'tuts' have a greater influence and can confuse an energy-based algorithm. Consequently, this approach was used only for the first demonstration versions of the verifier; later, more sophisticated techniques were developed.

4.1.2 Double-parameter approaches

Amplitude alone will not always distinguish speech from silence so several algorithms using additional signal parameters have been proposed as robust end-point detectors. Rabiner and Sambur (1975) and Gan and Donaldson (1988) make use of the relatively high *zero-crossing rate* exhibited by the signal during weak fricatives which may be used to differentiate them from background noise. The zero-crossing rate Z of a segment of N samples is defined as:

$$Z = \sum_{i=1}^{N} z_i$$

$$\text{where } z_i = \begin{cases} 1 & \text{if } x_i x_{i-1} < 0 \\ 0 & \text{otherwise} \end{cases} \tag{4.4}$$

Algorithms which work with both A and Z profiles can usually cope well in most situations, although the modelling of the background sound is important. Rabiner and Sambur (1975) use the initial samples of the file for this purpose but this assumption may not hold when recordings are taken in public places from speakers who have little experience of the system. Also, the algorithm is non-causal (involves searching forwards through the data), which limits the possibilities for real-time implementation, a potentially serious consideration. For most of the experiments in this project, the Gan and Donaldson (1988) approach was taken, which does not suffer from these limitations.

The idea is similar to that of Ney (1981), who identified a number of *states*, and allowable transitions between them which could describe the utterance. Recently, similar approaches using a state machine (Mauuary and Monné 1993) and a hidden Markov model (Acero, Crespo, de la Torre and Torrecilla 1993) have been reported. Also, Haigh and Mason (1993) implemented end-point detection using VQ models for speech and non-speech represented by cepstral coefficients, but without extensive testing.

The method seems appropriate for the ATM application because it was designed for nearly real-time deletion of silence from digitally-encoded voice mail messages. For the experiments in this project, only the initial and final silence/speech transitions were of interest and real-time operation was not appropriate as the speech was being recorded onto DAT. (As mentioned in Chapter 3, the data recorded for the initial tests were manually delineated; only the recordings of the large database required automatic end-point detection.)

The algorithm classifies segments of speech using A and Z defined above[5]. One of two adaptive thresholds for A is used to detect possible speech/silence transitions, depending on the classification of the current segment. The thresholds adapt (to varying signal conditions) because they are related to the average value of A for the 10 most-recent silent segments. The 10 A values are stored in a continuously-updated FIFO (first in, first out) buffer. This gives some immunity to variations in the background noise level and prevents pre-speech breath noise corrupting the true start-point of an utterance.

A segment may also be classified as speech if $Z > Z_{\text{sil}}$, a predetermined upper limit on Z during silent segments. The threshold conditions for both A and Z must apply for a (definable) number of contiguous segments before any or all of them are classed as speech. After some experimentation it was found that allowing Z_{sil} to adapt, using another FIFO buffer, improved accuracy in cases where the speech is quite soft and there is noticeable background noise. Since Z

[5]Actually, the equation for Z given by Gan and Donaldson (1988) is wrong.

tends to fluctuate more rapidly than A, storing more than 5 values in the buffer confused the algorithm.

The volume of utterances used in the large-scale tests prevented the possibility of checking the end-point detection of any more than a sample of the database. Most cases were adequate but it has been noted that even the modified algorithm is still very dependent on the initial values given to the parameters such as the transition thresholds and the default numbers in the FIFO buffers. If these are set up to give good results for particular recording conditions, the same level of performance cannot be guaranteed for other recordings. Unfortunately, the method is not adaptive enough.

Also, although there is no requirement to classify the first segment as silence, incorrectly marking it as speech – perhaps due to some background noise – causes an irretrievable misclassification of the start point. This is because the simplified approach employed here identifies the start-point of the utterance as the first segment to be labelled 'speech' (and the end-point as the last). The algorithm itself is less reliable at the start of the file because the FIFO buffers are still filled with their default values and no adaptation has taken place.

To analyse the variable behaviour of the program, an option was added for printing a file which logged the values of A and Z for each segment. Inspection of this file usually allowed accurate delineation of the speech, even in cases where the automated method had failed. The start-points, which seem to be harder for the algorithm to detect, are indicated by a few segments of higher than normal Z followed by a sharp rise in A. The dependence on initial parameters prevents the program from detecting this pattern in some cases. The ability, in most cases, to find manually the silence/speech transitions from the log file suggested that a pattern-recognition procedure might cope with such diverse signal conditions.

4.1.3 Artificial neural network

Artificial neural networks (ANNs) are often useful where a decision must be made by considering several factors or trends which a single algorithm may be unable to identify. The silence deletion problem seemed to be in this category so an ANN was generated using the NeuralWorks program and was trained with a few (about 5) sample files. In each case, the full captured signal was presented in segments of 100 samples, along with a desired output, either 0 or 1, depending on whether the segment was to be regarded as silence or speech. The ANN was a back-propagation type, with 100 inputs, 15 elements in the hidden layer, and a single output. After training, the network was exported as a C source file and was linked into an executable.

The utterances were to be used in experiments investigating different methods of reference template construction (described in Chapter 7). The experiments used the longest available utterance (*Sir Winston Churchill*) and it had been noted that only the start-point needed to be identified. The problem was simplified even further by allowing the initial /s/ to be removed.

The system seemed to work well but slowly, although this can be attributed mostly to the 99% overlap of analysis segments. Also, the computation for each segment involves multiplying large arrays by floating-point weights. Experiments with reduced segment sizes were still slow compared to the method described in Section 4.1.2. Since there were only 20 files to be processed for each of 214 speakers, the time factor was not critical but for other experiments with the digits, the ANN method was too slow to be useful.

The methods of silence deletion are summarised in Table 4.1.

Text	method	points detected
speaker's name	multi-threshold	start & end
ABCDEFG	manual	start & end
digits	(Gan and Donaldson 1988)	start & end
Sir Winston Churchill	ANN	start only

Table 4.1: The methods of silence deletion applied to the texts of the small and large databases. The *Sir Winston Churchill* utterances had only the silence preceding the speech removed.

4.1.4 Speech analysis

Following silence deletion, the utterance itself is considered to be purely speech from a single speaker: no corrective attempts are made when these ideal conditions do not apply. To prepare the speech for input to the comparison stage, the data are scanned to find the maximum amplitude, which is used to scale all the samples to be between -1 and 1. There are two reasons for this: firstly, it compensates for inter-utterance variations in speech amplitude and secondly, it prevents any numeric overflow that might occur in the subsequent analysis processes. Until this stage, the speech samples are stored as signed integers but scaling converts them to single-precision floating-point values.

Linear predictive coding

As mentioned in Chapter 2, the most favoured method of extracting vocal tract information from the speech signal is based on linear predictive coding (LPC). The topic is covered extensively in the literature (Atal and Hanauer 1971, Makhoul 1975) but briefly, LPC models the vocal tract as an all-pole filter excited by either quasi-periodic pulses from the vocal cords (voiced speech) or turbulent air-flow (unvoiced speech). During certain sounds, such as nasals, it is known that the vocal tract transfer function also contains zeros but generally these can be represented by combinations of poles.

For most speech sounds the shape of the vocal tract is almost constant over short periods of time (10–30 ms) and analyses of the vocal tract tend to be unchanged over this period. It is therefore sufficient to examine the speech signal in similar small *frames*. This is illustrated in Figure 4.3. In this project the frame size is 250 samples, equivalent to 25 ms. Each frame is represented by 12 LPC coefficients generated using the maximum entropy method (Press, Teukolsky, Vetterling and Flannery 1988). This is better known as the harmonic-mean method of computing the reflection coefficients of the lattice representation of the vocal tract filter, and as such, is guaranteed stable and needs no windowing (Makhoul 1977).

Time-normalised cepstral coefficients

The p LPC coefficients a_n are transformed to p *cepstral* coefficients c_n using the following recursive routine (Furui 1981).

$$c_n = a_n + \sum_{k=1}^{n-1}(1 - k/n)a_k c_{n-k} \qquad 1 < n < p \tag{4.5}$$

The cepstrum is the inverse Fourier transform of the logarithm of the magnitude of the frequency response of the vocal tract filter. The cepstral coefficients are widely used for speaker verification as they offer improved discrimination over LPC coefficients (Atal 1974). The reasons for this are not

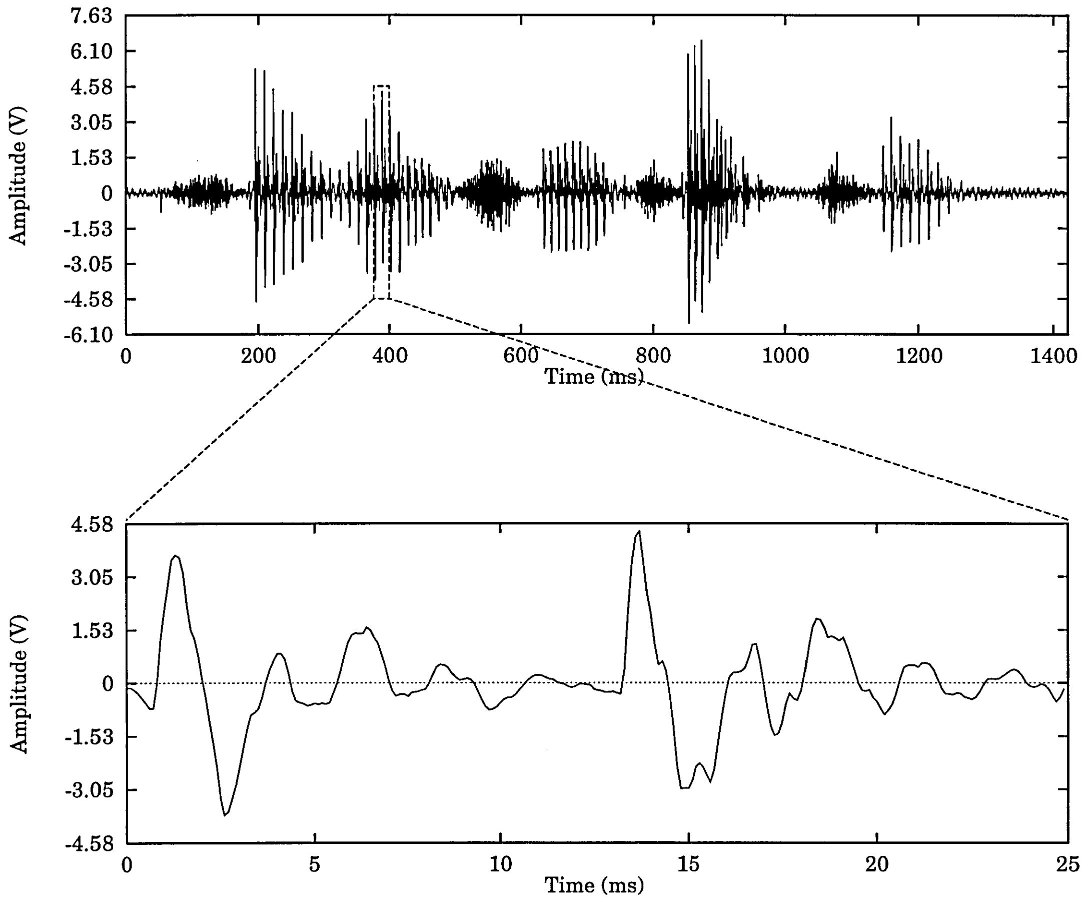

Figure 4.3: Utterances are analysed in 25 ms frames, such as that shown in the lower plot. The similarity between the first and second halves of the frame reflect the periodicity which characterises voiced speech. (The frame is a portion of the /ɪ/ vowel in *Sir Winston Churchill.*) In this case the pitch period is approximately 12.5 ms, corresponding to a frequency of 80 Hz.

well-understood but one explanation is that differences between spectra become more prominent on a logarithmic scale due to the wider dynamic range.

Incorporating the logarithm operation in the frequency domain also allows deconvolution of time-signals by spectral subtraction or equivalently, truncation of the cepstrum. This has direct relevance to speaker verification where the vocal tract characteristics may be separated from those of the excitation, which does not generally carry much speaker-specific information. To a large extent, this deconvolution is already performed by LPC which models the vocal tract regardless of excitation. For this reason, LPC-derived cepstral coefficients are more correctly known as pseudo-cepstral coefficients.

For each frame of the utterance, the 12 pseudo-cepstral coefficients are calculated and stored as a single *vector*, so called because the coefficients may be regarded as the co-ordinates of a point in 12-dimensional space. A vector is generated for each complete frame of the utterance and by summing them and dividing by the number of frames, an average cepstral vector is obtained. Subtracting this average vector from each vector in the sequence (utterance) normalises the signal with respect to channel characteristics such as the microphone response and background noise. Such aspects would be roughly constant over one utterance but may vary between utterances, thus introducing dissimilarities which are unrelated to the speaker's voice. The use of such time-normalised cepstral (TNC) coefficients is widespread and good results have been reported (Atal 1974, Furui 1981) although recently it has been suggested that the cepstral normalisation may be harmful when working with short utterances which do not allow reliable estimation of the channel (Furui 1994, Naik, Assaleh and Mammone 1994).

The transformation of the sampled utterance to a sequence of TNC vectors effectively compresses the speech to about one-tenth of the original size: each frame of 250 integers (2 bytes) is represented by 12 floating-point numbers (4 bytes). The elimination of redundant information allows the comparison process to focus only on important features, thus offering rapid matching. Also, since reference tokens are formed in much the same way – usually combinations of TNC vector sequences, as described in Chapter 7 – storage capacity is crucial where such identity tokens may be stored on a card.

4.2 Comparison process

The key task in the verifier is the comparison of a test token with a reference token. This verifier uses the dynamic time-warping (DTW) method although as mentioned in Chapter 2, there are several other applicable methods, such as vector quantisation (VQ) and hidden Markov modelling. A good introduction to DTW is given by Owens (1993, pages 140–146) and variations of the technique are detailed by Myers *et al* (1981). Appendix B describes mathematically the implementation used for this project.

DTW gives a measure of the similarity between two speech signals and allows the verifier to decide if they have been spoken by the same person. The similarity is assessed in terms of the geometrical *distance* between corresponding pairs of feature vectors. However, a simple frame-by-frame comparison is not sufficient because variations in speaking rate could mean that the nth frame of the test token does not really correspond to the nth frame of the reference token, even if the texts and speakers are the same. Each TNC vector only represents 25 ms so even slight variations in speaking rate – unnoticeable to a human listener – could cause errors.

Simply expanding or compressing the time axis of one utterance to match that of the other is usually not sufficient because the variations in speaking rate are not linear. DTW provides a dynamic solution by performing localised compression or expansion as required. In this application, the tokens are not actually compressed or expanded but the frame-wise distances are calculated as if they had been.

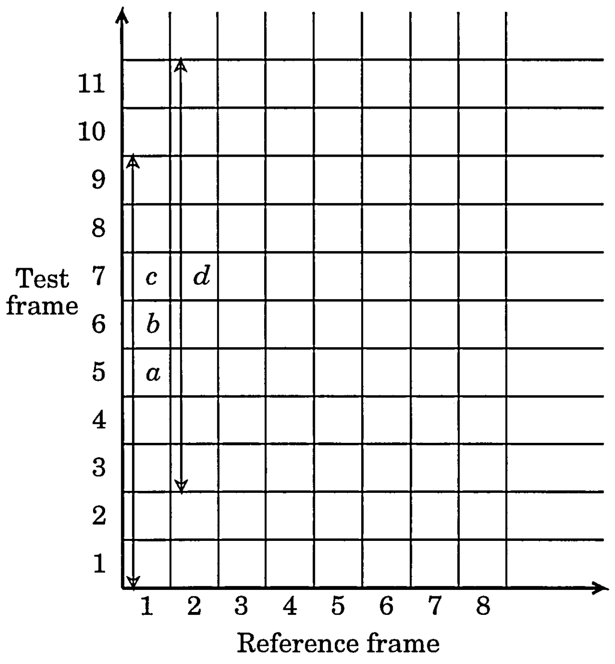

Figure 4.4: The initial stages of DTW. Each reference frame is matched against a test frame by searching in a range equivalent to 225 ms. The best match within this range sets the centre of the search range for the next reference frame, e.g. if c is the best match in column 1, then the search in column 2 is centred on d.

DTW is usually described using a grid of points such as that shown in Figure 4.4. The point (i, j) represents the distance between frame i of the reference token and frame j of the test token. The aim of DTW is to determine an optimal path – given certain constraints – through the grid. This *warp path* describes a mapping between the test and reference tokens designed to maximise their similarity. The process begins by comparing the first reference frame to the first few test frames. In theory, if the texts are the same, the first frames will already coincide but in practice this is rare due to the approximate nature of the start-point detection. Following Bernasconi (1990), the initial search range was chosen to be 9 frames (equivalent to 225 ms), i.e. frame 1 of the reference token is compared with frames 1–9 of the test token, as shown in Figure 4.4. The closest matching frame is then chosen as the centre of the range to be searched to match the second frame of the reference token. For example, the lowest distance in column 1 may be at the point marked c which means that frame 7 of the test token best matches frame 1 of the reference token. Frame 2 of the reference token is then compared to 9 frames of the test token, centred on frame 7, i.e. frames 3–11.

In searching column 2 for the lowest distance, the distances in column 1 are taken into account because the aim is to find the minimum distance path through the entire grid. However, certain slope constraints are applied to ensure that the warp path is monotonic: otherwise, some backwards talking would be implied! In common with Bernasconi (1990) and others, the slope constraints used are those suggested by Itakura (1975). In Figure 4.4, these conditions allow point d to be reached only via points a, b or c. Also, no more than one repetition of a single frame (e.g. c to d) is allowed.

The process continues until either the last reference or last test frame has been matched. The warp path may then be obtained by working backwards from this point through the grid to the start, passing through local minima. However, in this application, the important feature of DTW is the overall distance between the two utterances, referred to as the *score*. This is equal to the accumulated distance calculated at the last local minimum in the grid, normalised to the number of frames in the reference token. According to the principle of dynamic programming, the final local minimum is guaranteed to be the end of the best warp path within the given constraints.

This DTW method is essentially the *unconstrained end-points, local minimum* (UELM) method (Rabiner, Rosenberg and Levinson 1978). Other methods exist but UELM has the advantages of a fixed maximum number of frame comparisons and no restrictions on the relative durations of test and reference. Most importantly, the need for exactly end-pointed utterances is relaxed somewhat. The major disadvantage is that the search for local minima limits the search range for the globally optimum path. However, this is unavoidable without searching the entire grid.

4.2.1 Modifications to the dynamic time-warping method

Recently, variations on the DTW theme have been published which in future work, could be applied to the current method.

Since the warp path is a straight line at 45° to each axis when both utterances are exactly the same, it has been suggested that the level of geometric deviation from this path could indicate the mismatch between utterances. Booth, Barlow and Watson (1993) used various properties of the warp path as well as the score to identify a speaker. However, the increase in accuracy is slight: the extra computation is not.

Irvine and Owens (1993) used a verifier similar to this one and found that warp paths with unconstrained slopes produced better matches than the constrained case. This simplification was justified by noting that in a text-dependent verifier the local deviations from the ideal warp path are expected to be small and that the constraints may prevent finding the best match when the end-point detection is less than optimal.

Angus and Whitaker (1987) suggested a simplification to improve speed. Frame comparisons are aborted when the calculated distance exceeds some threshold; a fixed, penalty distance is used instead. The assumption is that high-distance points in the grid will not be included in the path so calculating precise details is a waste of time. However, finding a robust, reliable computation-cancelling threshold may be difficult.

4.3 Deciding to accept or reject

The output from the DTW process is a score, a number corresponding to the dissimilarity of the two voice patterns: zero for an exact match, higher scores indicating progressive degrees of mismatch. The decision to accept or reject the unknown speaker depends on whether the score is below or above some threshold.

As mentioned at the beginning of this chapter, most of the experiments presented here do not make an accept/reject decision; rather, score distributions for the genuine speaker and impostors are compared. However, certain results to be discussed in Chapter 7 do make use of error rates, for which a threshold is required.

4.3.1 Furui's threshold

Previous research has not really produced a sound method of determining thresholds. For example, Furui (1981) suggested calculating the threshold for speaker j, T_j, using the formula:

$$T_j = a(\mu_{\mathrm{imp},j} - \sigma_{\mathrm{imp},j}) + b \tag{4.6}$$

where $\mu_{\mathrm{imp},j}$ is the mean of the distribution of impostor scores for speaker j, and $\sigma_{\mathrm{imp},j}$ is the standard deviation of the same distribution. For each speaker, values for a and b are found such that the resulting threshold produces equal rates of false-acceptance and false-rejection errors. These *local* constants are then averaged over all speakers to give *global* constants, with which each speaker's threshold is re-calculated.

Furui's formula contains two empirically-determined constants a and b, which are rather arbitrary and whose final values are calculated with the assumption that the relationship between the score distributions of genuine speaker and impostor is the same for all enrolled speakers. Many studies, e.g. Thompson and Mason (1994), have found that this is not the case, with certain enrolled speakers being more 'identifiable' than others. (They are sometimes referred to in the literature as 'sheep', as in the phrase "separating the sheep from the goats".)

4.3.2 Thresholds based on ranked impostors

For this project, a new technique based on the ranking of impostors for each speaker is used to generate speaker-specific thresholds. This would allow a strategy of grouping together enrolled speakers of similar characteristics with a view to optimally separating them (described in Chapter 9). A similar method was employed by Booth *et al* (1993).

For any given speaker, each impostor's average score (over all verification attempts) was stored and used as a sort key to rank the impostors in descending order of similarity to the genuine speaker. This produces a list of impostors most likely to be falsely verified as that speaker. Speaker-specific thresholds are then set to be equal to the nth closest impostor's average score, as shown in Table 4.2. This method provides a mechanism for biasing the verifier toward either false acceptance or false rejection by increasing or decreasing n.

It should be noted that although n is essentially an index into a list of impostors, negative values of n are allowed: the corresponding threshold is calculated by subtracting the closest impostor's average score from that of the nth closest impostor. The difference is then subtracted from the closest impostor's average score to give the threshold.

```
IMPOSTOR.AVG
    300
    320
    350
     ⋮
```

$\Longrightarrow$

n	calculation	threshold
-3	$300 - (350 - 300)$	250
-2	$300 - (320 - 300)$	280
-1	$300 - (300 - 300)$	300
1	-	300
2	-	320
3	-	350

Table 4.2: For each speaker, the file `impostor.avg` contains a list of average impostor scores, sorted in ascending order. The ranked impostor method determines a threshold by indexing into this list. A 'mirror image' of the differences between the first and nth lines of the list list allows negative indices.

Repeating experiments with varying threshold levels can produce charts (see Chapter 7) showing the interaction of false accept and false rejection, which would allow the system tolerance to be tuned according to the application.

The definition of error rates and alternative methods of assessing the verifier performance are given in Chapter 5.

Chapter 5

Assessing Performance

As mentioned in Chapter 4, verification is based on the degree of mismatch between test and reference tokens, reflected (in this project) by the DTW score. An operational verifier would decide to reject or accept the speaker by comparing this score with some threshold. Two possible errors may occur: false acceptance (FA) or false rejection (FR). Varying the threshold changes the probability of occurrence of either error, at the other's expense.

To compare the performance of different verifiers, or different configurations of the same verifier, the *equal-error rate* (EER) is often used. This is derived by setting the threshold at such a point that the probability of FA equals that of FR. However, this can sometimes give a distorted evaluation because the error rates are unreliably estimated with small test databases.

This chapter begins by examining the EER before deriving alternative performance metrics which are then demonstrated to be more suitable for predicting the verifier performance from limited test data. These new methods will dominate assessments of verifier performance in later chapters where experimental results are reported. Also included will be a discussion of the *jackknife* technique of artificially enlarging a small data set.

5.1 Traditional assessment

A general statement of the accuracy of a verification system must cover both FA and FR errors because it is possible to bias the decision towards minimisation of one at the expense of the other. Hence, the EER is often quoted but this implies the existence of some threshold value of DTW scores. That is, an identity claim from an individual is accepted (or rejected) if the verification score is less than (or more than) the threshold. This can be demonstrated by examination of the interaction of the FA and FR rates as the threshold is varied. A typical case is illustrated in Figure 5.1. Low thresholds limit the acceptance rate – false or otherwise – and the rejection rate is correspondingly high. The converse is true when the threshold is high. The error rate at the intersection of the two curves is the equal-error rate.

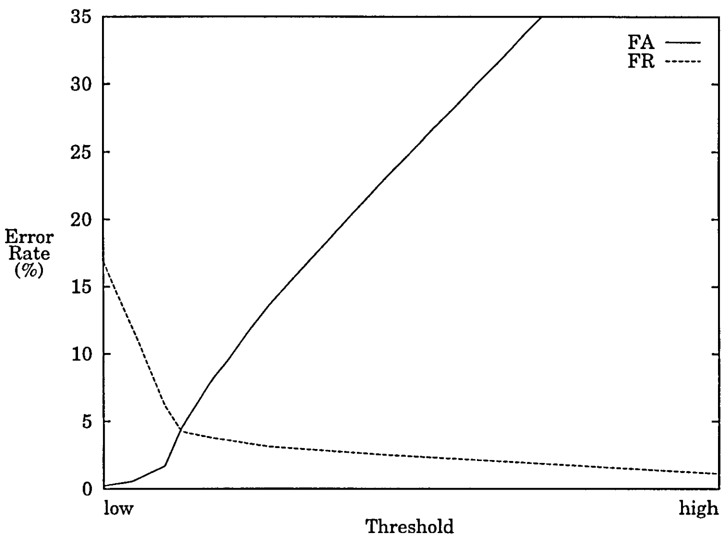

Figure 5.1: Typical interaction of the FA and FR error rates as the accept/reject threshold is varied. The intersection marks the equal-error rate; in this case, about 4%.

Strictly speaking, this point represents the equal average-error rate because the figure is arrived at by taking average FA and FR rates across the whole speaker set, for various thresholds. This is not the same as finding the EER for each individual then presenting the average; that would be the average equal-error rate. Indeed, at the EER point in Figure 5.1, there is no guarantee that the FA and FR rates are equal, for any of the individual speakers. However, this approach is much easier; given any threshold, the FA and FR rate can always be calculated. Finding the average EER would entail finding the true EER for each speaker, which itself is not easy because it is an iterative process: estimating the threshold and assessing the error rates until they are equal. In fact, the error rates cannot ever be guaranteed to be equal because the FR rate will be quantised coarsely due to the limited number of self-test scores (15 for most of the experiments in this project).

5.1.1 Speaker-specific thresholds

Due to the inherent variability of individual voices, a single, global threshold cannot be guaranteed optimal for every speaker; instead, individual thresholds are needed. This is because the typical range of scores for one speaker may not be the same as for another speaker. For instance, for speaker 053 the self-test scores range from 100–180 and the impostor scores range from 180–580; yet for speaker 054 the corresponding ranges are 50–100 and 110–450. An absolute threshold, such as 105 or 180 say, would be inappropriate for at least one of these speakers.

To illustrate the effect of universally applying an absolute threshold, consider Figure 5.2 which shows the correlation between speakers in the large database, according to the tests described in Chapter 7. Referring to the horizontal and vertical axes as x and y respectively, each point (x, y) represents the average score for speaker y 'impersonating' speaker x. However, only points representing scores less than 180 are shown. (The threshold value of 180 was chosen to show only a certain amount of detail and was arrived at empirically.)

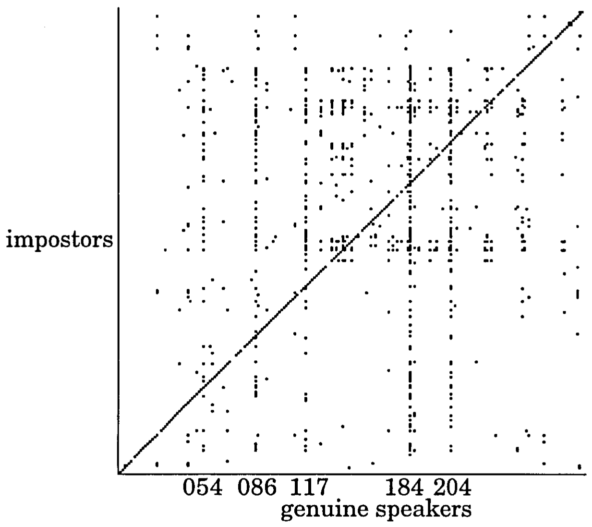

Figure 5.2: Each co-ordinate represents the correlation between a pair of speakers: a black dot is shown if the average score for one speaker ('impostor') impersonating another ('genuine') is less than 180. The varying levels of detail for different genuine speakers indicates that such a universally applied threshold is inappropriate.

If this universal threshold was appropriate, only the $y = x$ line would be apparent but there are plenty of other correlations shown in Figure 5.2. Indeed, there are 5 prominent vertical stripes relating many impostors to speakers 054, 086, 117, 184 and 204. However, it should not be be concluded that these speakers are easily impersonated by the others; although each point in the stripe indicates that an impostor scored (on average) less than 180, in all of these cases the speaker's own score is the lowest. One of the results of the experiments described in Chapter 7 is an ordered listing of the speakers, ranked according to how reliable verification is for them. According to the listing, only 1 of these 5 speakers is amongst the worst of the test set; the reason for the stripes is simply that the ranges of scores (both genuine and impostor) are lower.

Conversely, for some genuine speakers, there is no $y = x$ dot in Figure 5.2. Again, this is due to the unsuitable threshold; for these speakers even their own average scores are higher than 180. Checking the ranked listing reveals that most of the speakers are not especially poor in terms of verifier performance and this again confirms the need for speaker-specific thresholds.

The reason for the variation in typical score ranges amongst speakers is not clear. It is to be expected that one speaker's range of self-test scores will differ from that of another speaker because this is a typical effect of sampling the population. However, it is surprising to find that the range of impostor scores should somehow be linked to the genuine scores; it is not obvious why the impostors *en masse* should get lower scores when impersonating different people, as illustrated in Figure 5.3. Basing the reference template on atypical utterances is a possible explanation. Chapter 7 examines various methods of constructing the template although not this particular aspect which would need further work to investigate.

Given the need for speaker-specific thresholds, a mapping to these from an apparently linear scale is offered by the ranked impostor method (described in Chapter 4) and this allows a permissible error-rate strategy to be set globally but vary locally. The majority of error rates quoted in this report are assessed on thresholds based on this method.

5.2 Score distributions

Describing the verifier performance in terms of error rates has the disadvantage that the method of generating the threshold is implicitly included in the assessment. Also, different verifier configurations may all report zero errors for

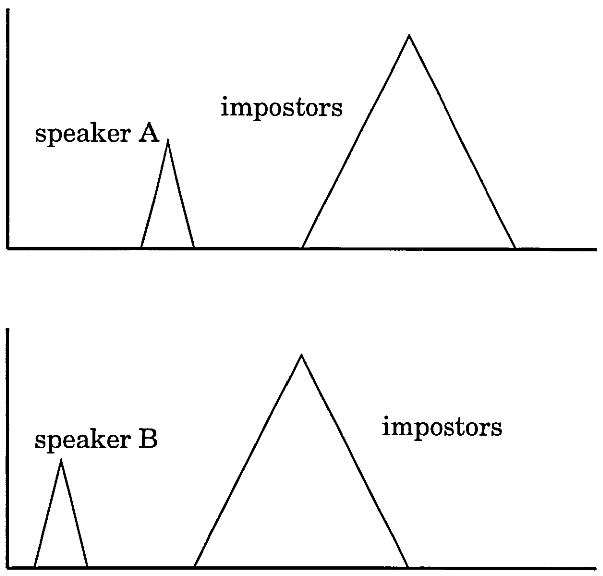

Figure 5.3: The gap between (abstract) genuine and impostor distributions is the same for speaker A as it is for speaker B, although the ranges of scores are different. The impostor distribution seems to shift *en masse* to maintain the distance from the genuine distribution.

a speaker although the configurations are not all equally good: the probability of error really needs to be considered. This is especially important when testing with databases composed of restricted populations. For an ATM application, the target customer population is so large that even the large database (described in Chapter 3) falls into this category.

An alternative approach is to consider the underlying statistics of the distributions of verifier scores for the speaker under test (designated genuine) and the other speakers in the test population (designated impostors). A typical pair of distributions is shown in Figure 5.4 which was generated as a result of one of the experiments of Chapter 7 using the *Sir Winston Churchill* utterances from the large database.

The distributions are shown as histograms with each column representing the number of scores observed in a range of 25 units. (The reader should recall from Chapter 4 that the verifier score comes from the DTW distance measure which is zero for a perfect match and higher for progressive degrees of mismatch.) Most of the experiments with the large database produce a much smaller distribution for the genuine speaker than for the impostors, making it difficult to see if they overlap. In Figure 5.5 the data of Figure 5.4 are reproduced on a larger scale

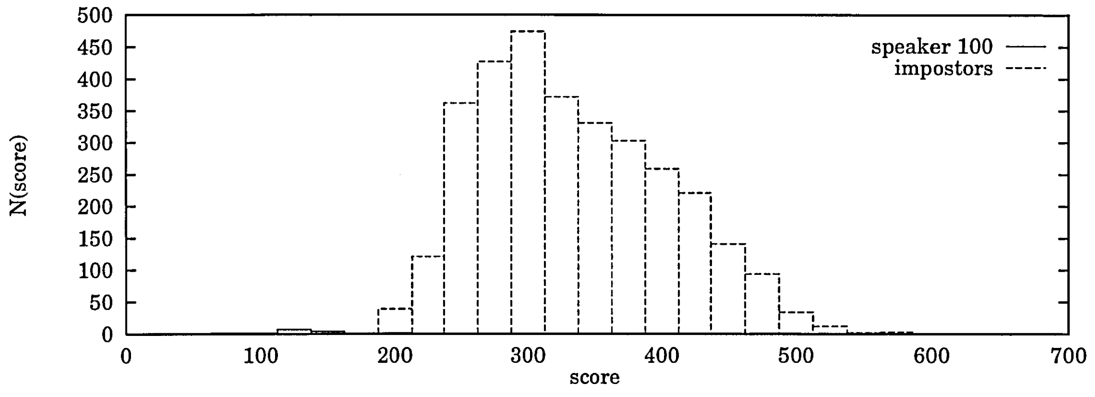

Figure 5.4: The score distributions for speaker 100 are typical in that they overlap slightly, although this is not clear from the graph.

so that the shape of the genuine distribution is clearer and the overlap is more obvious.

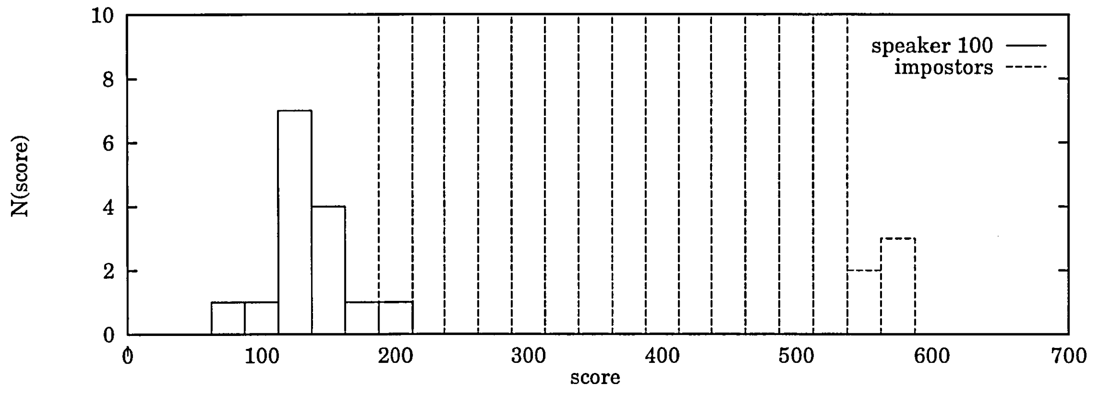

Figure 5.5: Typically, the distributions of genuine and impostor scores overlap slightly, indicating that setting a threshold anywhere would cause at least one error, either FA or FR.

The overlap is shown as columns for both the genuine and impostor scores in the same range (centred on 200). This would not always guarantee that the two distributions actually overlap, merely that both have values in a common range, but inspection of the raw data confirms overlap in this case. (There are 4 impostor scores lower than the highest genuine score.) Even if there was no actual overlap, the *probability* of overlap is high, given that there are 39 impostor scores in the same range as the highest genuine score.

One of the best speakers, in terms of verifier performance, is speaker 239. Figure 5.6 (again, magnified for clarity) shows the corresponding score distributions which are clearly separate, so that a threshold set anywhere between the maximum genuine score and the minimum impostor score would produce zero errors.

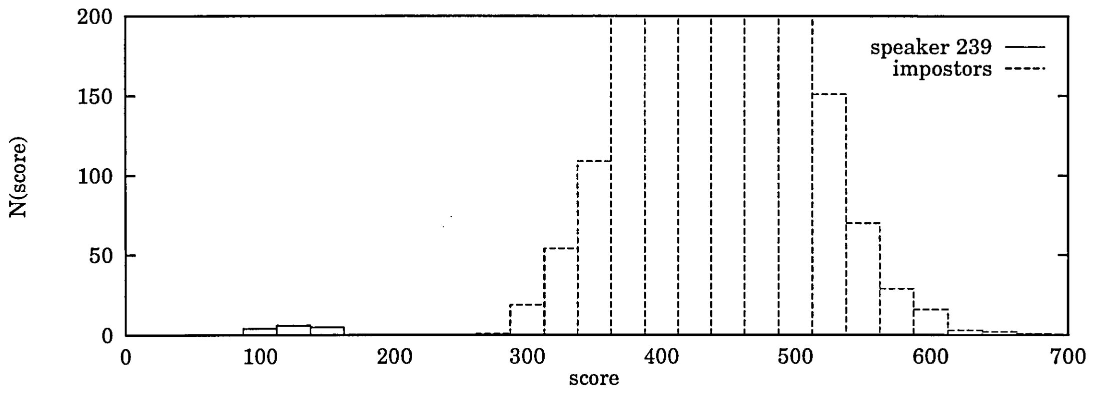

Figure 5.6: The verifier performs well for speaker 239, as demonstrated by the separation between the genuine and impostor distributions.

Besides the separation between distributions, the shape is also of interest. For speaker 100 the impostor distribution is skewed left, indicating that the majority of impostors are closer to, rather than further from, the genuine speaker; for speaker 239 the impostor distribution is almost normal.

At the other extreme, speaker 123 was among the worst of the test population, as illustrated in Figure 5.7. Again, the scale has been altered to show the two genuine speaker scores above 400, much displaced from the rest of the genuine distribution. Such outliers indicate stark differences between test and reference utterances, possibly due to artifacts of the recordings (see Chapter 3) or poorly constructed reference templates (see Chapter 7). (Speaker 123 is used in a case study of correcting for outliers in Section 8.2.1.)

5.3 New methods

Most of the experiments in this project avoid setting thresholds and estimating error rates since these aspects are important only during the final stage of development of a verifier. Recently, other researchers have expressed a similar opinion (Forsyth, Bagshaw and Jack 1994, Oglesby 1994). Instead, to estimate the verifier performance in its eventual application, the main focus is on the relative position and shape of the genuine and impostor score distributions, particularly the degree of separation and/or overlap for each speaker. To help in

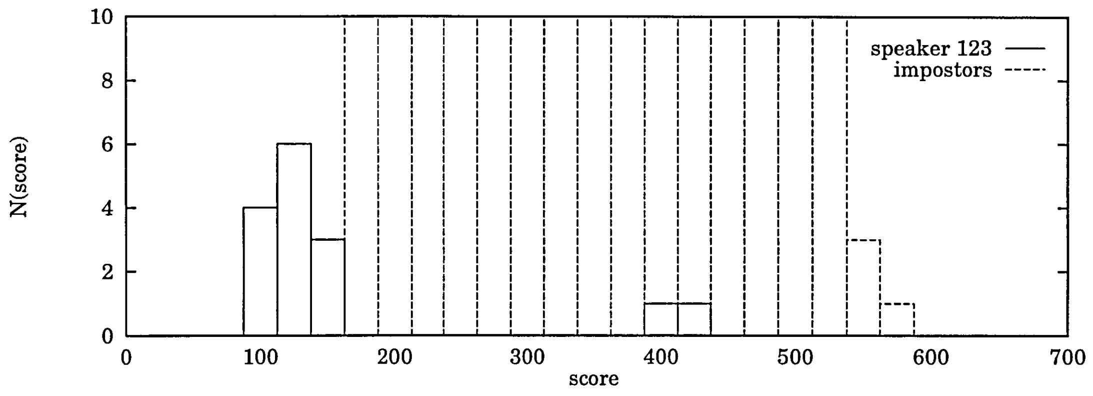

Figure 5.7: For speaker 123 the verifier performs poorly, indicated by two genuine scores well within the impostor range.

this assessment, two new figures of merit are employed: a version of the classical d' (borrowed from signal detection theory) and an 'overlap' function, F.

5.3.1 Using d'

The d' measure, from classical signal detection theory (Green and Swets 1966), assumes the existence of two response distributions (signal and signal-plus-noise, corresponding here to genuine and impostor) which are Gaussian and equal variance. Then:

$$d' = \frac{|\mu_s - \mu_{sn}|}{\sigma} \tag{5.1}$$

where μ_s and μ_{sn} are the means of the signal and signal-plus-noise score distributions respectively, and σ^2 is the common variance. In the case of DTW scores for genuine speakers and impostors, the assumptions are invalid but d' seems nonetheless to provide a reasonably consistent measure of separation. The following modification of the original equation (5.1) compensates for the inequality of variances:

$$d' = \frac{\mu_{\text{imp}} - \mu_{\text{gen}}}{S} \tag{5.2}$$

where μ_{imp} and μ_{gen} are the means of the impostor and genuine distributions respectively, and S is the geometric mean of the standard deviations of the two distributions, given by $\sqrt{\sigma_{\text{gen}}\sigma_{\text{imp}}}$. Unlike the situation in signal detection theory, d' can be negative in the (highly undesirable!) case that $\mu_{\text{gen}} > \mu_{\text{imp}}$.

As mentioned before, the use of d' implicitly assumes that both distributions are Gaussian but this is generally untrue. The genuine distribution usually only contains 15 scores which is not enough to properly estimate its shape. Also, when any of the females in the small test population is designated as the genuine speaker, the impostor distribution is bimodal – the lower mode being attributable to the female impostors. Although it may be expected that the female impostors would score lower than the males, such stark division is surprising. Although the author can detect no striking similarities amongst the five female speakers, perhaps the verifier can. This would explain why the effect is not repeated with the large database although another factor could be the ratio of males to females of about 2:1. Nevertheless, the d' figure is less reliable using speech from the small database when the genuine speaker is female.

5.3.2 Overlap function (F)

A simpler, alternative measure of verifier performance, F, is based on the amount of overlap of the two score distributions. This is an easily-computed, *ad hoc* figure which has a valid range of $[0, 1]$, although this is split into two equal sub-ranges to cover the distinct cases of overlap and non- overlap, or (total) separation. F is computed differently for each case (labelled F_{overlap} and $F_{\text{non-overlap}}$) but increases linearly with verifier performance over the entire range. F_{overlap} is inversely related to the probability of an impostor's utterance producing a score that is within the range of the genuine speaker's scores:

$$F_{\text{overlap}} = 0.5 - \frac{f}{2N_i} \tag{5.3}$$

where f is the number of 'failures' in terms of an impostor score being within the range of genuine scores while N_i is the number of impostor trials. In cases of non-overlapping distributions, F expresses the separation between them as:

$$F_{\text{non-overlap}} = 1 - \frac{G_{\max}}{2 \cdot I_{\min}} \tag{5.4}$$

where $G_{\max}(\geq 0)$ is the maximum genuine score and $I_{\min}$ is the minimum impostor score. The factor of 2 in equations 5.3 and 5.4 sets F to be 0.5 at the overlap/non-overlap boundary.

5.3.3 Ambiguity rate

Some of the results in Chapter 6 are expressed with *ambiguity rates* to compensate for the inability of d' to definitely indicate overlap of the score distributions. The ambiguity rate gives the ratio of scores within the *ambiguous zone*, i.e. between I_{min} and G_{max}. Because this measure is based on the extreme points of the distribution, it can appear to overstate the extent of overlap. For instance, a single high genuine score within the range of impostor scores spoiling otherwise distinct distributions would have a very high ambiguity rate. However, with only 15 scores to estimate the genuine distribution, single outlying scores cannot be neglected.

5.4 Jackknifing the genuine distribution

In common with many studies, the speech data used in this project suffers from a lack of recordings from each speaker; when designating one as genuine there are only a few utterances with which to estimate the genuine score distribution. One way of compensating for this is the *jackknife* technique, described by Miller (1974), which may be used to expand the number of apparent observations of some variable. In particular, experiments (such as those in Chapter 7) using the digits from the large database can benefit from this technique because they were only recorded 10 times each, and 5 of these are used as reference data.

Several variants of the basic method exist but the application in this project is as follows. With 10 genuine tokens $G_1, \ldots, G_{10}$, each one is left out in turn and tested against every possible reference template that can be made from the remaining 9 tokens. For example, since there are $\binom{9}{5} = 126$ different combinations of 5 tokens from a pool of 9, when G_1 is left out, it is tested against templates $R_1, \ldots, R_{126}$, which are formed from $G_2, \ldots, G_{10}$. The process is repeated by testing G_2 against $R_{127}, \ldots, R_{252}$ which are combinations of $G_1, G_3, \ldots, G_{10}$ and so on. The benefit of jackknifing is that the genuine distribution can be estimated with 1260 scores instead of 5 but in practice the 126 scores for each designated test token tend to be very similar to each other but not always to those of another test token. In effect, jackknifing produces 10 sub-distributions (each of 126 scores) which are not guaranteed to blend smoothly into a single genuine distribution.

5.5 Comparison of the new methods

The d' rating is used extensively throughout this report since it describes the shape and separation of the distributions (neither neglecting nor emphasising outliers) as a single figure. The F rating is mostly used in Chapter 8 where the experiments call for repeated assessments of performance within a short time. It is a second choice to d' when dealing with the small database because the rating is noticeably quantised due to the number of impostor scores appearing in the denominator. It has the advantage that separation or overlap is indicated by being, respectively, above or below 0.5. The ambiguity rate is rarely quoted but clearly signals undesirable score distributions which d' may fail to convey. In Table 5.1 all three ratings are given for the score distributions shown in Figures 5.4–5.7.

Figure	d'	F	Ambiguity rate (%)
5.4 & 5.5	4.79	0.499	1.56
5.6	10.1	0.730	0.0
5.7	1.81	0.050	89.7

Table 5.1: Comparing assessment measures of the scores distributions from earlier figures.

Chapter 6

Utterance Content and Duration

In designing a verifier for widespread public use, two main issues must be addressed: security, and convenience to the user. Both aspects must be considered when choosing the phrase to be spoken in a text-dependent verifier, such as that being used in this project. In terms of security, several papers (Soong *et al* 1985, Velius 1988) have demonstrated that the verification text strongly influences reliability by recording different error rates for the same verifier and same speaker set but using different texts. However, it is not clear if this differential is due to the phonetic content and prosodic effects of the words or simply to their varying durations.

From everyday experience of recognising voices on the telephone, it is clear that longer utterances make the task easier. Also, it is known that utterance duration is an important factor in text-*independent* verification (Higgins and Wohlford 1986), but for maximum convenience to the user it would be desirable to use the shortest text possible so that the overall process is less time-consuming.

This chapter examines these aspects of the verifier with a series of experiments, with a view to assessing the feasibility of using randomly-prompted short utterances (such as the digits) as the verification texts.

6.1 Initial experiments

To test the effect of varying utterance duration on the verifier performance, recordings from the small database (see Chapter 3) were used. This database consists of 10 speakers saying the phrase *ABCDEFG*, which allowed the creation of texts of variable length by appropriate manual editing. It should be noted that an exactly linear measure of duration was sacrificed in order to preserve the continuity of natural speech, by segmenting at word boundaries.

Each experiment is a collection of *trials*, each of which consists of a number of *tests*. Each test compares the reference token for a specific person (designated as 'genuine') with another token, which may be from the genuine speaker (a *self-test*) or from the other speakers (an *impostor test*).

6.1.1 Experiment 1: forward segmentation

To obtain progressively longer tokens, the first n letters were extracted from the phrase, i.e. A, AB and so on, until the full-length utterance was used. For each designated genuine speaker there were 7 trials, corresponding to the 7 different utterance lengths. Each trial comprised 15 self-tests and 135 (15×9) impostor tests. In each trial, the statistics of the distributions of genuine and impostor scores were recorded and provide the basis for analysis of the verifier using the d' figure, as described in Chapter 5. Figure 6.1 illustrates, for a typical speaker (JP), the changes that take place in these distributions as the utterance length is increased.

The first plot in the sequence of Figure 6.1 shows that both distributions are very narrow (apart from a single outlying impostor score), and their close proximity indicates poor discrimination, reflected in the low d' rating. However, the dynamic range of scores produced by the DTW process is small when short utterances are used and the fact that most of the genuine scores are below most of the impostor scores shows that some inherent characteristics are being detected.

When two words are used (AB), the impostor distribution begins to resemble a normal distribution but the shape of the genuine distribution remains the same as before, although it has shifted higher up the score axis. The shape is skewed left which is welcome because it means that most of the self-tests were categorized as genuine with a high degree of confidence: only a few produced scores high enough to be confused with impostors. These positive aspects produce a d' of 4.5 which is often high enough to suggest complete separation of the genuine speaker from the impostors, although not in this case.

Separation, however, is achieved using three words (ABC), and further skewing of the genuine distribution results in a d' of 5.5. This falls back to 5.0 when using four words ($ABCD$) and the graph shows an overlap once more, although only one impostor score is in the overlapping area. However, the

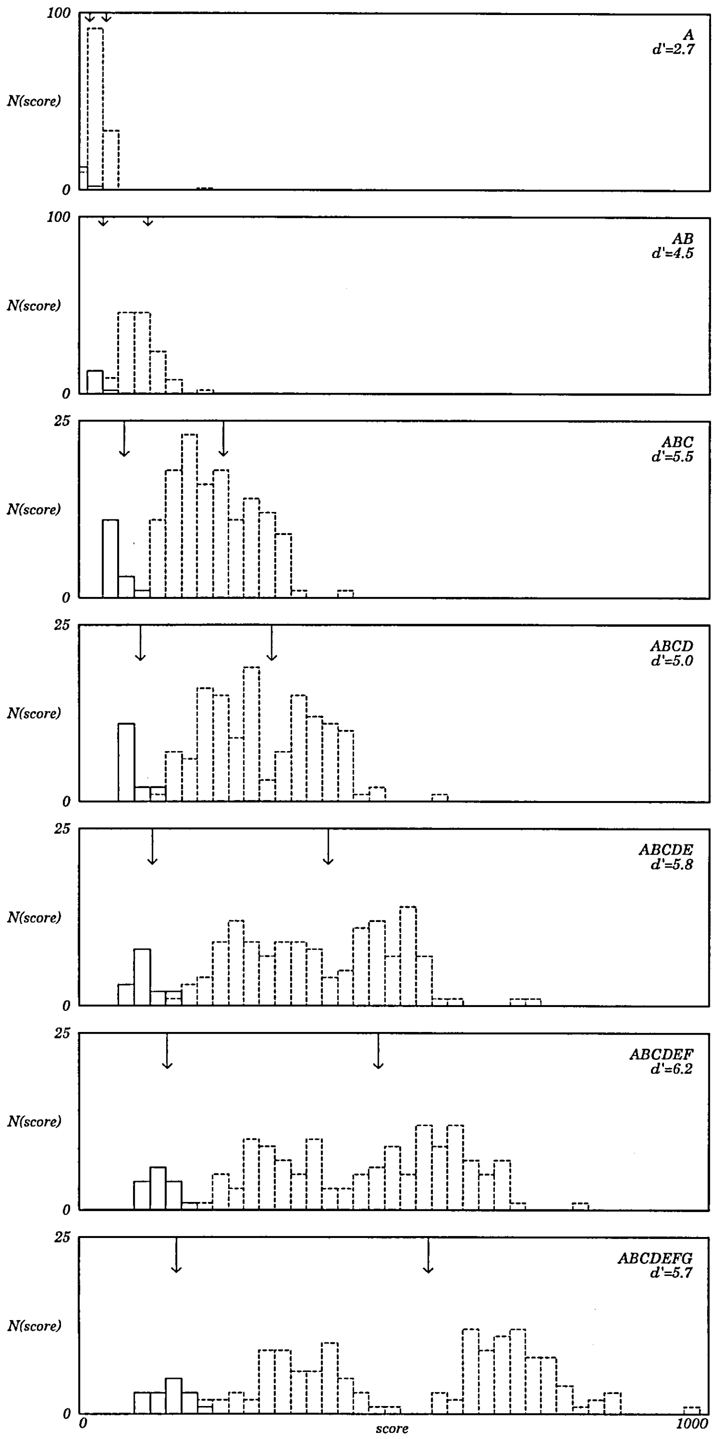

Figure 6.1: Each plot shows the verification text (top-right) and the resulting score distributions for the genuine speaker (solid lines) and the impostors (broken lines) and their means are indicated by arrows. The d' calculated for each pair is also given. Note that the first 2 plots use a smaller scale than the rest. The sequence shows a general trend of better discrimination with longer utterances, illustrated by the diverging means and the effective rejection of an increasing proportion of impostors.

impostor distribution has changed significantly: it has become bimodal. This is typical for a female genuine speaker (as in this case), and, as discussed in Chapter 5, the d' rating is slightly less reliable. In terms of the number of impostor scores close to the highest genuine scores, the speaker discrimination is still improving. This is reinforced by the continued upward shift in the impostor mean.

For five words (*ABCDE*), the d' climbs again to 5.8 although the level of overlap is similar to the previous case. Again, the shape of the impostor distribution plays a part in this because the trough around the mean is less stark than the previous case so the d' is free to rise as the separation between the genuine and impostor means increases. This continues with the six-word text *ABCDEF* and because the trough is no longer centred around the mean, the highest d' of the series is attained. This situation reverses again in the final graph where the full utterance was used: the impostor distribution is now completely bimodal and the trough is close to the mean, resulting in the d' dropping again.

Over the entire sequence of graphs, three trends emerge:

- the means of the genuine and impostor score distributions increase at divergent rates;
- the proportion of the impostor population at risk of being falsely accepted decreases;
- the shapes of the distributions change: the genuine distribution becomes less skewed and more normal, and the impostor distribution becomes less normal and more bimodal (in this case).

If the first two trends are repeated across the speaker set, it may be concluded that performance improves with longer utterances. Rather than visually analysing the sequences of score distributions for each speaker, the d' obtained using each utterance-portion was calculated and plotted as a graph. Traces for all 10 speakers are shown together in Figure 6.2 to illustrate the general trend and also the variation across the speaker set.

It can be seen that the overall trend is for an increase in d' with increasing lengths of utterance. This is illustrated by the fact that the d' for *ABCDEFG* is higher than the d' for A, in every case. However, the rate of increase varies widely within the speaker set. For instance, the gradient of a straight line between the

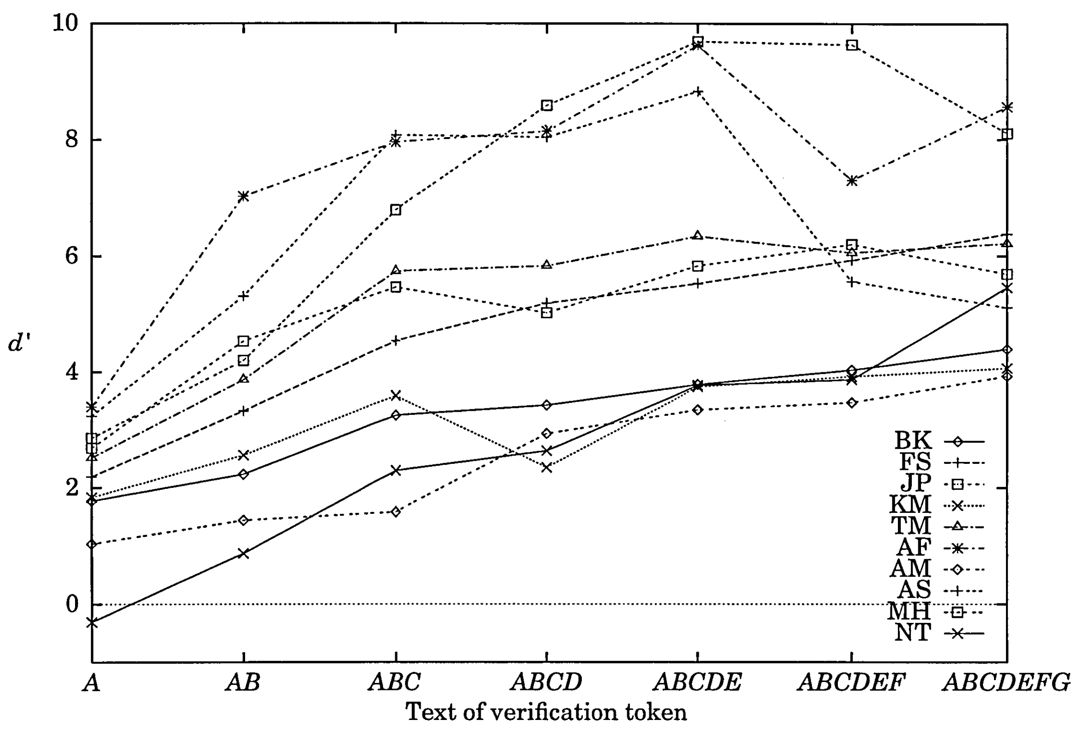

Figure 6.2: Speaker separation (measured by d') as a function of the number of words in the verification token. For all speakers, the overall trend is increasing but monotonic in only four cases (NT, AM, BK and FS). BK–TM are female; AF–NT are male.

first and last results for speaker BK is 0.44 but 0.96 for speaker NT. Similar statistics are not really valid for most of the other speakers because their results do not increase montonically. The most noticeable examples of this are AS and AF, for whom the d' drops dramatically between *ABCDE* and *ABCDEF*. In each case, inspection of the data revealed that the decrease in d' was due to a single atypical genuine score. Listening to the original speech files which produced the offending scores failed to provide any obvious clues as to the essential differences between them and those used in the construction of the reference template. There were no spikes or clicks and the (manually selected) end-points were accurate. Clearly, processing the speech to produce time-normalised cepstral coefficients allows the comparison mechanism to detect differences which the human listener cannot.

As mentioned in Chapter 5, the d' figure can be affected considerably in these cases even though the actual scores are acceptable. For instance, in the case of AS, the single outlying score was still substantially below the lowest impostor score. However, the reliability of d' as a measure of separation is not undermined by these cases: values of 5.5 and 7.3 (for AS and AF respectively) still suggest clear separation. Chapter 7 describes methods that may be employed to de-emphasise such isolated atypical utterances.

It is not clear why the rates of performance-gain should vary so widely across the speaker set. The implication is that for some speakers, adding more text adds less discriminating material than for others. This could be because of the speed of talking: three words for one speaker may take the same time as five words for another. However, inspection of the hand-segmented speech files does not support this theory. Although the slowest speaker on average took 25% longer than the fastest speaker to say the full utterance, most of the other speakers are near the middle of this range. The d' and average duration (over 15 utterances) for each speaker (saying the full phrase) is presented in Table 6.1 and shows no direct correspondence between speaking rate and d'.

subject	d'	average duration (s)
BK	4.39	2.15
FS	6.38	2.55
JP	5.69	2.3
KM	4.07	2.05
TM	6.22	2.225
AF	8.56	2.3
AM	3.93	2.225
AS	5.11	2.35
MH	8.11	2.35
NT	5.45	2.325

Table 6.1: For each speaker saying *ABCDEFG*, the speaker separation (measured by d') and duration (in seconds) averaged over 15 repetitions, is shown.

Five of the seven words in the verification text are taken from the E-set – they all feature the /i/ vowel. It seems that for some speakers, repeating the sound several times adds no new information and hence increasing (this) utterance duration makes little difference. This could be due to an inherent variability which does not significantly decrease the confusability with the other speakers. In other words, regardless of the number of examples of their speech, no common theme emerges. In these cases, it is also likely that the reference template was unable to characterise the speaker. A corollary of this is that less animated speakers will not produce markedly higher scores with longer utterances. The DTW process would accumulate only small local distances due to the consistent /i/ sound. This could explain the rising plot for AF (widely acknowledged as the possessor of a deadpan voice) and for MH and AS (the author and project supervisor respectively) who may have subconsciously spoken in a consistent manner.

6.1.2 Experiment 2: reverse segmentation

The finding of experiment 1 that the verifier's performance improves with increasing utterance length could, in theory, be explained as an artifact of the position of specific words within the phrase. For instance, the later words in the phrase may carry more speaker-specific information than the earlier words. To test whether the performance depended on *content* rather than length, segmentation was also done starting from the phrase-end and producing progressively longer utterances by including only the last n words. (For example, *G*, *FG*, *EFG*, etc.) The results of this experiment are similar to those for the forward segmentation: an overall trend of improved performance with increased utterance length. Figure 6.3 shows the average d' values for both forward- and reverse-segmented utterances. For comparison, Figure 6.4 shows the corresponding average values of the overlap function F. The close agreement of results obtained with the two different measures is evidence of the improving effect on verifier performance of longer utterances.

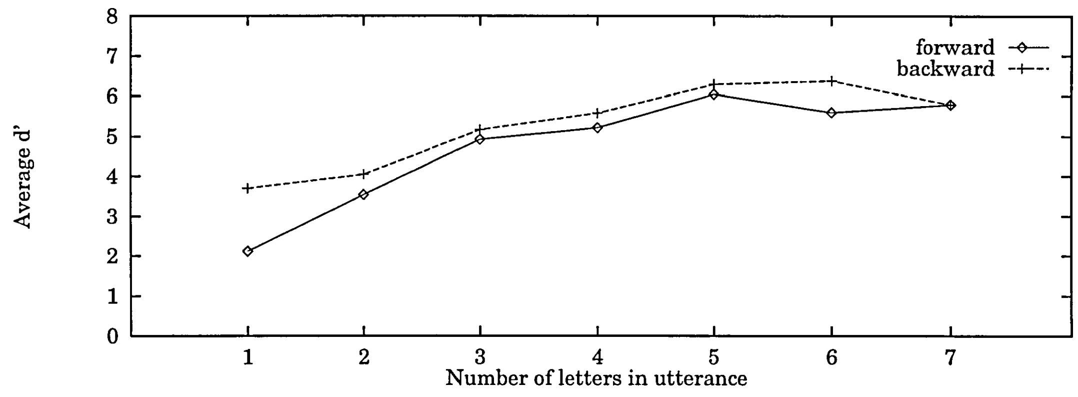

Figure 6.3: Speaker separation (as measured by average d') as a function of the number of words in the verification token, for both forward- and reverse-segmented utterances.

The main difference between the results of this and experiment 1 is that reverse segmentation produced a slightly better one-word performance. One explanation

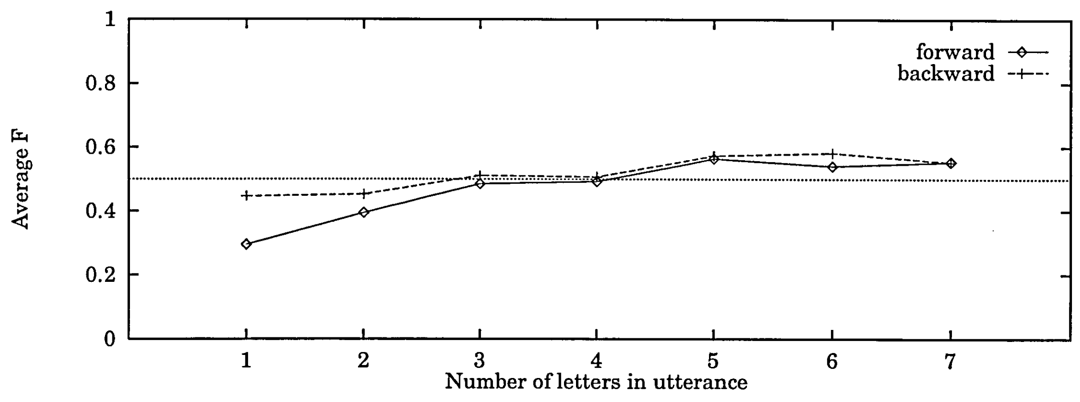

Figure 6.4: Speaker separation (as measured by average F) as a function of the number of words in the verification token, for both forward- and reverse-segmentation.

might be that G (/ʤi/) with its two phonemes (and coarticulation between them) contains more speaker-specific information than the single-phoneme A. (Spoken in isolation, A tends to be a diphthong, but when followed by B, as in these recordings, this is less apparent.) Su, Li and Fu (1974) suggested that coarticulated sounds were good for verification because the movement of the articulators was rapid and difficult to modify consciously. Another possibility is that the first word in any connected-word utterance is not good for verification, perhaps as an effect of a high initial outflow of breath. In any event, the average performance using single-word tokens is poor.

6.2 Persevering with single-word tokens

Experiments 1 and 2 demonstrated that performance with single-word tokens is poor but the potential benefits of using such tokens warrant further investigation. A scheme whereby the ATM could prompt users to say single words chosen randomly from a finite set would be attractive for two reasons.

Firstly, the introduction of an unpredictable element in the verification text increases security against fraudulent access. In a text-dependent verifier such as the one used in this project, the risk exists that someone could 'impersonate' an enrolled speaker by simply using a recording of the text. Normally this would not pose a major threat since the frequency characteristics of the recording and playback devices would distort the 'utterance' sufficiently for the verifier to reject it. However, for the ATM application, the existence of multiple access points (all the machines on the network) requires that the verification system is able to disregard the frequency distortions introduced by using different microphones. As described in Chapter 4, this is achieved through the use of time-normalised cepstral coefficients; the associated security risk would be lessened with random texts.

Secondly, a *sequential* method (Bielby 1987) may be used where the machine continues to prompt for words from a pre-chosen set until the decision to accept or reject the speaker may be made with a specified confidence. This would allow banks to alter the level of security for different speakers or different locations, as appropriate. Convenience to the user could also be increased because, based on the results of experiments 1 and 2, for some speakers, as few as two words may be needed to verify identity.

6.2.1 Experiment 3: concatenated templates

With this scheme in mind, an experiment was devised in which the test data consisted of a single, connected utterance but the reference data consisted of concatenated templates for the *individual* words. This scheme does not incorporate the sequential process described above but instead assumes that the user would be prompted for a fixed number of words, randomly chosen but presented together. The use of individual templates allows a truly random selection of words for the verification phrase without the prohibitive capacity requirements of storing templates for every possible combination of words in the set. Presenting these words together as a single phrase is convenient for the user and less error-prone than prompting for and capturing several short utterances.

This experiment was conducted using new recordings from 5 of the 10 speakers saying each of the discrete words A to G five times. $ABCDEFG$ was used as test data matched against the concatenated individual templates $A+B+\cdots+G$, as illustrated in Figure 6.5.

The results (shown in Table 6.2) were very poor, none of them exhibiting any worthwhile degree of separation of genuine speaker and impostors. Given the findings of experiments 1 and 2, it might be expected that the verifier would perform poorly because it is merely combining several single-word results, each of which is itself poor. However, another possible reason could be that the DTW process is sensitive to the presence (or absence) of inter-word silence in

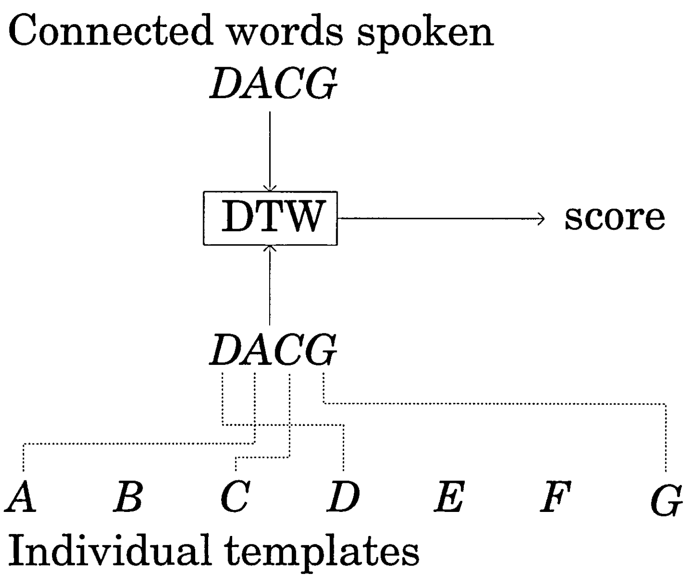

Figure 6.5: In experiment 3, connected-word utterances are matched to a concatenation of templates for the individual words. The general scheme is indicated here for an example text *DACG* but the text used in the experiment was *ABCDEFG*.

the reference template if it is not duplicated in the test token. Although the individual templates were constructed from hand-segmented utterances, the joins in the concatenated reference token cannot be guaranteed to match those of the test token. Coarticulation between words in the test data further compounds this problem. In an attempt to exclude these effects, a fourth experiment was devised.

6.2.2 Experiment 4: discrete utterances

For the previous experiment, discrete words formed (by concatenation) the reference data; the test data were the full-length utterances as in experiments 1 and 2. In this experiment, the discrete-word recordings provided *both* the test and reference tokens. This is illustrated in Figure 6.6.

After manual end-point detection, the first of the 5 tokens of each discrete word was taken as reference for that speaker. Thus, for each word and each speaker, there were 4 genuine and 20 impostor tests. The scores for these comparisons were totalled for the 7 discrete words A to G, and d' and F values calculated. Table 6.3 shows the resulting d' values (the results for F were essentially the same) and compares them with those found in experiment 1.

subject	d'	
	Expt. 1	Expt. 3
MH	8.11	-0.63
JP	5.69	3.41
FS	6.38	3.43
AS	5.11	3.66
AF	8.56	2.89
average	6.77	2.55

Table 6.2: In experiment 3 connected-word utterances are matched against composite utterance-length templates made by concatenating templates for the individual words. This diminishes performance, as seen by the lower (average) d' ratings compared with (non-averaged) d' ratings for experiment 1.

subject	d'	
	Expt. 1	Expt. 4
MH	8.11	12.05
JP	5.69	7.16
FS	6.38	5.64
AS	5.11	6.15
AF	8.56	5.95
average	6.77	7.39

Table 6.3: d' resulting from summing scores for discrete-word matches (experiment 4) and by matching continuous utterances (experiment 1).

These results show that separation attainable by summing scores for discrete-word matches can at least equal that obtained with continuous-word templates and test data. This tends to confirm the interpretation of experiment 3 given above, i.e. that word-boundary differences between test and reference data can cause severe problems for verification. It could be argued that the good results of experiment 4 are due to the removal from the data set of the closest impostors for certain speakers: this experiment used only 5 of the original 10 speakers. However, the same condition applied in experiment 3, with considerably poorer results. Overall, the result is encouraging since it indicates that a practical system which operates by prompting the user for a number of discrete utterances in random order could work well.

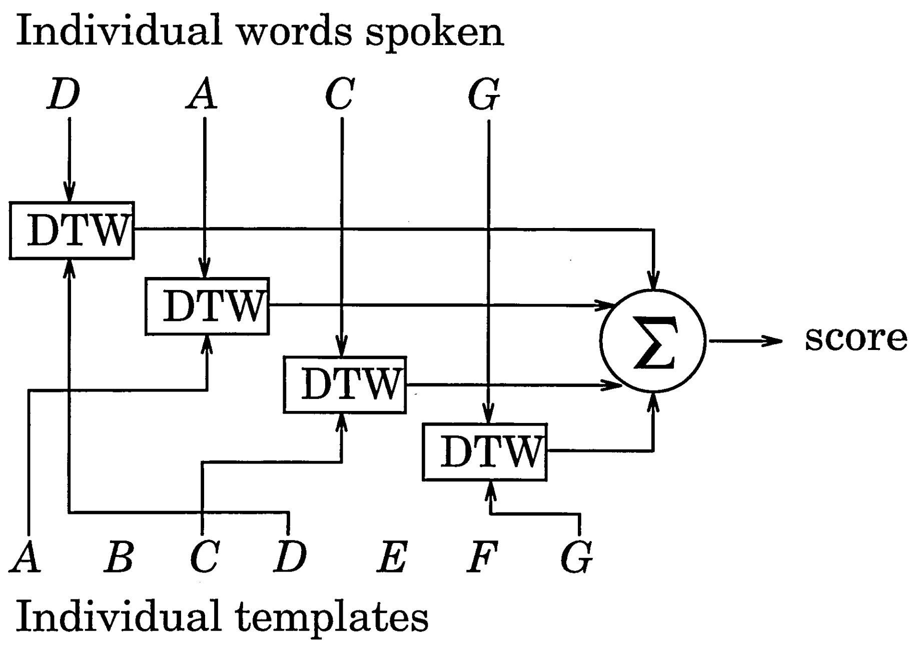

Figure 6.6: In experiment 4, a series of word-length utterances are matched to templates for the individual words. The scores are summed to give an overall score for the phrase. An example phrase *DACG* is shown here although the phrase *ABCDEFG* was used in the experiment.

6.3 Extensive tests (large database)

The findings of the previous experiments are encouraging but not convincing. The main detractions are a lack of subjects — at most, only 5 speakers of each gender — and impostor tests which greatly out-number self-tests. Consequently, the reliability of performance measures based on distribution-pairs suffers. To counter these negative aspects, further experiments were conducted using the large database (described in Chapter 3). The database comprises 10 recordings of the digits *one, two,* $\cdots$, *nine* plus the words *zero*, *oh* and *nothing* by each speaker. Since the verifier is to be used in a financial context and current users are accustomed to typing identification numbers, the digits seem to be the natural choice for the verification vocabulary. Indeed, many published studies have used the digits as a test set (Buck, Burton and Shore 1985, Rosenberg and Soong 1986, Irvine and Owens 1993).

Each speaker was 'impersonated' by over 200 impostors and the jackknife technique (described in Chapter 5) was applied to compensate for the lack of self-test data. The increased volume of data allows more general conclusions to be drawn but there is a price to be paid in terms of computation and storage requirements. With 25680 ($12 \times 10 \times 214$) utterances, it was absolutely necessary to use an automated system of silence deletion and, as noted in Chapter 4, doing this robustly is not trivial. Any results in this section are as much a test of the silence deletion technique as of the verifier. Of course, in a practical system, both would be crucial to highly accurate verification.

6.3.1 Experiment 5: relative usefulness of digits

It has been suggested that certain digits are more useful for speaker verification than others (Soong *et al* 1985). If this were true, it might be possible to selectively weight some digits so that the 'random' process would include them in the verification phrase more frequently. To test this, the d' figure was calculated for each speaker for each digit.

Since there are only 10 utterances of each digit by each speaker, the jackknife method was used to obtain a more general idea of the genuine speaker score distribution. This method is described more fully in Chapter 5 but, briefly, each utterance is taken in turn and tested against every possible reference template that could be constructed with 5 of the other 9 utterances. This creates 126 scores for each of the 10 utterances, 1260 genuine scores in total. The distribution of these scores is considered as the genuine distribution and its statistics (mean and standard deviation) are used in the calculation of d'.

Applying the jackknife technique to the 10 genuine utterances effectively produces 1260 reference templates for that speaker. Producing the impostor distribution by testing every impostor utterance against all of these templates would generate almost 3 million scores, and is therefore impractical. Instead, it was decided to choose just one of the genuine speaker's templates: the template which produced a self-test score nearest to the average. Thus, for each speaker and digit, a d' was calculated based on the distributions of 1260 genuine scores and 2130 impostor scores. (The scores themselves were written as text files, compressed and archived to be used in later experiments.) These figures were averaged across all speakers to produce an average d' for each digit. These are shown in Table 6.4, arranged in descending order.

The d' ratings in Table 6.4 indicate that the best performance is achieved using the word *zero*; *six* also works well. There are then six words which offer

digit	avg. d'
zero	2.83
six	2.63
seven	2.26
nine	2.25
five	2.19
eight	2.19
two	2.07
nothing	2.03
three	1.89
one	1.80
four	1.65
oh	0.81

Table 6.4: The average d' across all speakers in the large database for each digit. As expected, no single digit provides enough speaker-specific information to completely separate the genuine speaker from impostors (indicated by $d' \ll 4$). However, there is a clear difference between best and worst.

equally average performance followed by three below average words. Finally, the word *oh* is clearly the least useful, with an average d' of 0.81; this again is a demonstration that short utterances offer little discrimination between speakers.

As can be seen from Table 6.4, the d' for every digit is considerably less than 4.0, the figure which generally corresponds to complete separation of the genuine speaker from impostors. This reinforces the findings of the previous experiments that a verification phrase consisting of a single digit would be insufficient. However, when the same results are plotted as a graph (Figure 6.7), it can be seen that the average d' figure is, in most cases, concealing the wide range of performance available for each digit.

In Figure 6.7, for each digit, a point is plotted with a y-coordinate corresponding to the d' for each of the 214 speakers. The concentration of most points in the middle of each range indicates that for most of the digits, the performance for most of the speakers was similar. The appearance of a few outlying results confirms the analysis of the previous experiments that the performance varies widely across the population. For most digits, the outlying results with a $d' \gg 4$ suggest that complete separation is possible with only a single digit as a verification text. Also, analysis of the data shows that these outlying results, good and bad, are

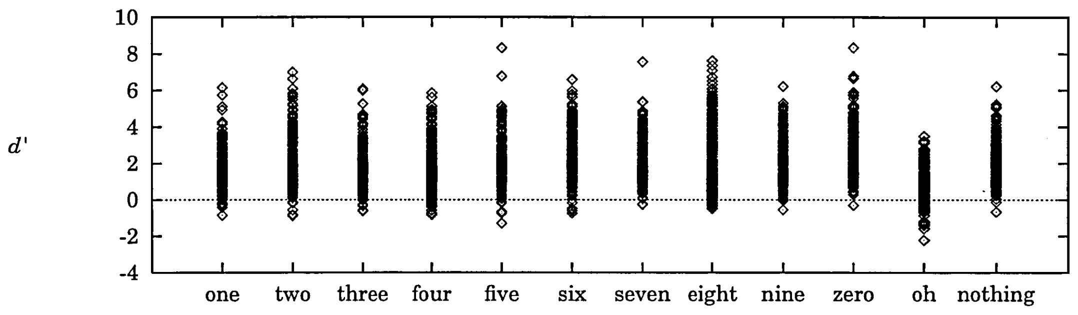

Figure 6.7: Each vertical line of points above the name of a digit represents the d' for each speaker for that digit. For all but one of the digits (*oh*), a few isolated observations of $d' \gg 4$ suggests that for some speakers, pronounced separation is possible with only a single digit.

not exclusively attributable to the same speakers: certain digits are better than others for individual speakers. This development suggests an enhancement to the envisaged scheme: the verification texts could be prompted in a speaker-specific, weighted-random manner.

Checking the silence deletion algorithm

As mentioned in Chapter 3, the quality of the recordings varies widely across the database and this creates problems for the automatic location of the silence-to-speech transition. Before proceeding further with experiments using the digits, it was decided to check that the silence removal process had been sufficiently accurate: effective but without removing parts of the speech as well. To do this, a small subset (8 speakers) of the large database was taken. These speakers were chosen on the basis that the signal-to-noise ratio of their recordings was high, and comparable to each other's. This ensured that the silence deletion process would affect the captured utterances in roughly the same way, in effect, a best-case scenario for applying the process to this database.

Scores of this subset were extracted from the score archives described in Section 6.3.1. A new figure of d' was calculated for each speaker; again, 1260 scores make up the genuine distribution while 70 (7×10) scores are used to compute the statistics of the impostor distribution. The new d' ratings were averaged across all 8 speakers for each digit and are shown in Table 6.5.

Table 6.5 shows no improvement over the results with the entire database,

digit	avg. d'
four	1.47
seven	1.36
two	1.07
nothing	0.98
five	0.88
six	0.75
eight	0.70
nine	0.65
zero	0.59
three	0.57
oh	0.32
one	0.28

Table 6.5: The average d' for each digit using a subset of the large database, originally selected as a best-case scenario for the silence deletion algorithm as applied to these recordings.

given in Table 6.4. This is surprising since this was meant to be the best-case scenario. There are a few possible factors which could explain this finding. One is that the signal conditions play a large part in the comparison and since these recordings were all of the same quality, there is substantial confusion amongst them. This theory could also extend to the fact that all of the speakers are male, most of them with local accents. The fact that the number of scores contributing to the genuine distributions greatly out-numbers those of the impostors is also significant because of the possible distortion of the jackknife method.

The jackknife process is used to expand the apparent number of observations of the self-test score in order to obtain a reasonable estimate of its distribution, or relative frequency. In this case, this involves matching each utterance against every possible reference template not containing that utterance. This produces 10 sub-distributions, each made up of 126 scores. Examination of all 1260 self-test scores for a few speakers reveals that amalgamating these sub-distributions does not necessarily produce a smooth blend. For example, Figure 6.8 shows all 1260 self-test scores for speaker 008 saying *five*. Vertical lines mark the boundaries of each of the 10 groups of scores. The contributions from 2 of the utterances (6 and 7) are clearly visible as being different from the other 8. Since an attempt is being made to characterise this speaker, in practice it might be allowable to discard these two uncharacteristic utterances. If this is done for this speaker, a great improvement in the corresponding *d′* is obtained: in the subset experiment the *d′* rises from 0.65 to 1.85. Eliminating the poor-scoring utterances reduced the mean and standard deviation of the genuine distribution.

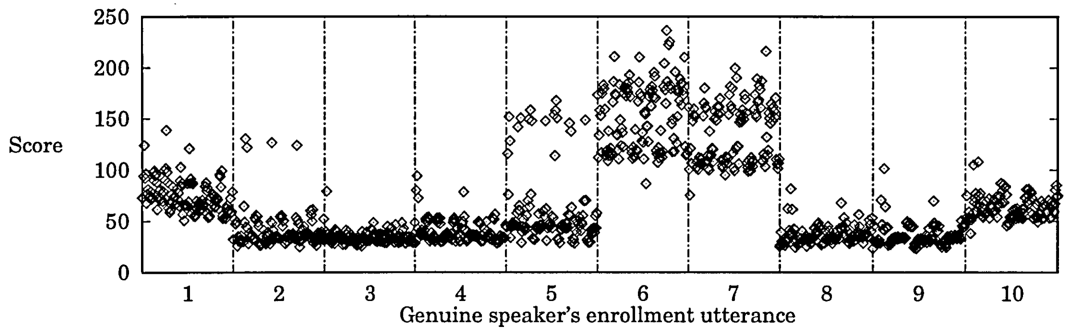

Figure 6.8: The jackknifed genuine distribution is composed of 10 sub-distributions; the boundaries are marked with vertical lines. It is clear that bands 6 and 7 are different from the others.

Inspection of jackknifed distributions for a few other speakers and utterances revealed similar patterns, to lesser extents, but not in all cases. These findings raise questions about the validity of the jackknife process although it must be remembered that without it, the genuine distribution would have to be estimated with as few as 5 scores, which would be even less reliable.

6.3.2 Experiment 6: summation of single-word sequences

The results of experiment 5 reinforce the theory that single-word tokens generally cannot be used successfully as the verification text. Despite this, experiment 4 found that summing the individual scores over enough words could produce complete separation. It was decided to test the validity of this finding by repeating experiment 4 using the large database. The scores for the individual words were already available, having been calculated in experiment 5 and stored in archive files. In theory it would be possible to generate a *d′* rating for each speaker 'saying' any string of the 12 digits but in practice, the time and volume of results are prohibitive. To minimise the computation and simplify the analysis, three constraints were imposed.

- The speakers to be designated as *genuine* would be picked at random.

- All 12 digits would be used, selected randomly but excluding repetitions within each string, i.e. each string was one of over 479 million permutations.
- Instead of using all 1260 jackknife self-test scores, a single sub-distribution (of 126 scores) was chosen (randomly) to represent the speaker. As mentioned previously, the 10 sub-distributions for a given speaker and digit are not guaranteed to have similar statistics so, for the sake of completeness, they should all be considered; random selection is an economical alternative.

The program was run for 5000 iterations, which should give, on average, about 20 results per speaker. No efforts are made to prevent the same digit-string being used repeatedly for any given speaker; in the unlikely event of this happening, it is probable that a different sub-distribution would be selected for use as the genuine distribution. An example iteration is shown in Figure 6.9.

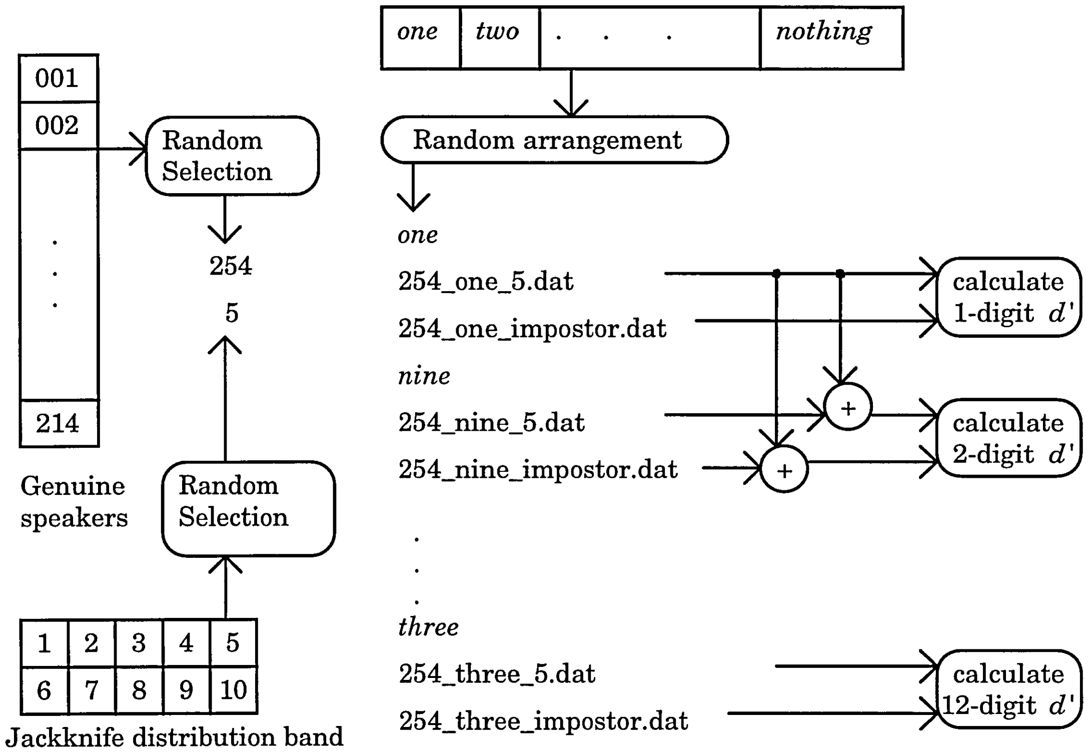

Figure 6.9: In experiment 6, the genuine speaker is selected randomly as is one of the speaker's 10 sub-distributions. The d' is calculated for progressively longer portions of the randomly-ordered string of digits.

For the first digit in the random permutation, the appropriate archived score files are retrieved and d' is calculated. The score files for the second digit are then added to those of the first and the d' for 2 digits is calculated. Addition of the score files takes place on a line-by-line basis i.e. the first genuine score for the first digit is added to the first genuine score for the second digit, and so on. The distribution statistics of these accumulated scores are re-calculated after every digit. The process continues until all 12 digits are used, at which point a d' rating will exist for a single-digit, a 2-digit string and so on up to a 12-digit string. Two lines of information were appended to a results file after every iteration: the speaker number, the sub-band of the jackknifed genuine distribution, the randomly-ordered digit string and the d' computed for each length of string. Table 6.6 shows a portion of the results file in a tabular format. The abbreviations z, o and n represent *zero, oh* and *nothing* respectively.

254	1	9	4	5	z	o	n	6	8	2	7	3
/5	-0.16	0.53	0.66	1.40	1.68	2.49	3.60	3.89	4.48	4.68	4.91	4.40
116	5	1	z	2	8	n	7	4	6	o	9	3
/5	8.19	7.69	6.04	7.89	8.78	10.5	10.6	10.5	11.3	11.2	11.0	11.0
262	5	6	8	o	3	7	2	z	n	1	4	9
/2	5.95	7.93	8.17	9.40	12.4	12.9	13.4	12.7	14.3	15.3	16.3	17.2

Table 6.6: A few examples of the results obtained in experiment 6. The words *zero, oh* and *nothing* are abbreviated z, o and n, respectively. Most cases show a general increase in d' with more digits in the string, although the rise is not guaranteed to be monotonic. The d' rating of 17.2 for speaker 262 indicates that stark separation is possible with this scheme.

All of the results were averaged to give a general indication of the speaker separation for each number of digits in the string. Figure 6.10 shows the steady increase in d' as more digits are added to the string.

Error bars are included in Figure 6.10 to show the wide range of performance across the speaker set which was also apparent in the small database tests. Although the maximum levels do not change substantially with different lengths of digit-strings, the minimum level increases to the point where, with 12 digits, the lowest d' is just below zero. However, this still represents poor performance, probably indicating speech data corrupted by noise or consistently poor end-point detection or a jackknife distortion. Such distortions are also probably accountable for d' ratings approaching 20, which indicates exceptionally distinct score distributions. An example of the clear separation indicated by some of

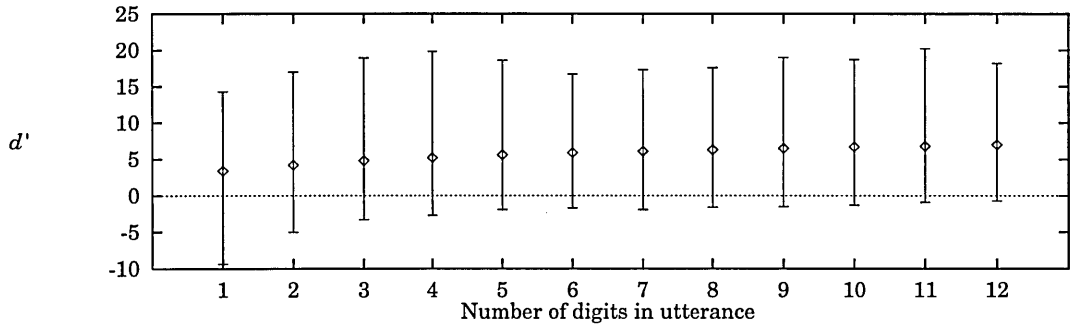

Figure 6.10: Summing the scores for individual word matches increases the discrimination of the verifier, on average.

the results in experiment 6 is shown in Figure 6.11 which illustrates the score distributions when the d' is 17.

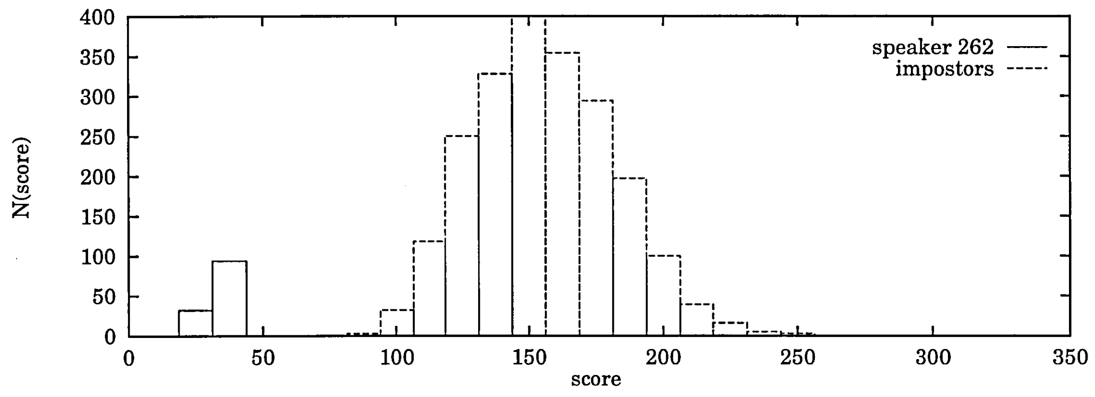

Figure 6.11: An example of the discrimination achieved occasionally in experiment 6 where the genuine scores are taken from a single sub-distribution of the jackknifed data. In this case (speaker 262 saying *nine*), the stark separation is reflected in the exceptionally high d' rating of 17.

It transpires that the curve of Figure 6.10 can be closely modelled by a function G of the number of digits, n, of the form:

$$G(n) = a + be^{cn}$$

Using the Mathcad package, values of $a = 7.75$, $b = -4.95$ and $c = -0.159$ were determined empirically and have been found to fit the curve very well. This is shown in Figure 6.12 which focuses on the $[3, 7.5]$ range of d' to show the closeness of fit.

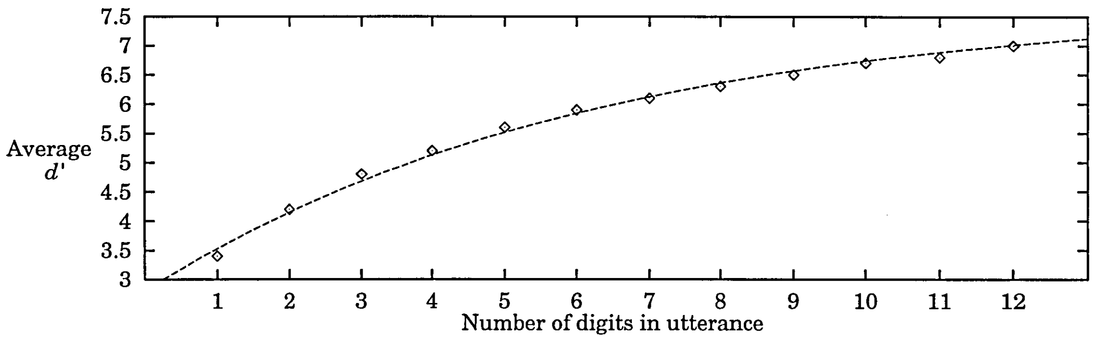

Figure 6.12: An exponential curve (broken line) very closely fits the points produced in experiment 6.

This model suggests a theoretical upper limit ($d' = 7.75$) on the verifier performance for an average speaker. This represents good separation but would require an infinite number of digits! Even 12 digits would not be a practical length of utterance. Perhaps of more interest is the fact that with only 3 digits the average speaker would be just separable ($d' \approx 4.5$) from impostors.

6.4 Conclusions

Utterance length has been shown to have an important effect on speaker separation in text-dependent verification. Using continuous-word test and reference data, separation increases with utterance-length, although not at the same rate for all speakers and not always monotonically. The strong presence of the E-set in the verification text may be distorting the results of experiments using the small database.

In a practical system, it would be attractive to prompt the user for random-order utterances from a small vocabulary. Ideally, the words could be presented together so that the customer need only make a single continuous utterance. Experiment 3 investigated this but failed to find the scheme feasible. This is apparently because of the problem that word-boundary effects pose to the DTW verification technique. Simply totalling scores for matches between discrete-word reference and test data, however, was found to work well, in experiments using both databases.

This apparently contradicts the finding that single-word tokens are of little use in verification; simply accumulating poor scores would not be expected to produce a good score. However, to some extent, this is the basis of DTW, where individual local distances are summed over the entire token (albeit in an optimal manner). The incremental differences between the genuine speaker and an impostor eventually add up to a sufficient amount to discriminate between them. A possible explanation of this is that for any frame-to-frame comparison there is only a *probability* that the distance will be low if both frames are from the genuine speaker. In general, there is a smaller probability of small distances between frames of the genuine speaker and an impostor. So, increasing the number of observations increases the discrimination. Future work could examine this in more detail, possibly using the continuous *Sir Winston Churchill* utterances but assessing the discrimination on a frame-by-frame basis.

Chapter 7

Constructing Reference Templates

The experiments described in Chapter 6 are based on a comparison between a test token and a reference template for the claimed identity. For the verifier to be consistently successful, it is vital that the reference templates are constructed properly from the training data. This chapter examines the following four methods of building such templates and assesses the corresponding verifier performance in each case:

- Furui (1981) method
- Bernasconi (1990) method
- mean separate score method
- majority decision method

The first two methods combine the original training data to form a *single* reference token; the latter two retain all the individual training utterances as distinct tokens which are used at verification time. The obvious disadvantages are the extra time needed to do five verifications instead of one, and the additional memory required to store the template data. These may be offset by greater flexibility in both the decision mechanism and any scheme which may be introduced for updating the templates to account for long-term variations in the customer's voice. Also, storage may not be a major issue as the verifier would be incorporated into the next generation of ATMs which would use 'smart' cards with sizeable on-board memory.

7.1 Methods

For both the initial (small database) and the extensive (large database) experiments, the first 5 of each speaker's 20 utterances were used to construct a template data set. These are then used as the reference against which all test tokens (generated from the remaining 15 utterances of every speaker) are compared. No test token is ever used in the construction of any reference template. This regime was designed to minimise the complexity of the experimental arrangements and it is clear that the suitability of the first 5 utterances as reference material cannot be guaranteed. Although the purpose of constructing a reference template is to characterise the speaker, the lack of multi-session recordings may cast doubt on the effectiveness. Single enrollment sessions are very desirable to minimise inconvenience to the customer.

The template data set may consist of a single token or a collection of several tokens, depending on the method chosen.

7.1.1 Method 1: Furui

Furui has worked extensively on speaker verification and his paper (Furui 1981) described a text-dependent DTW-based verifier which has served as the basis for many other researchers, such as Soong and Rosenberg (1986) and Gaganelis and Frangoulis (1991), including Bernasconi (see Section 7.1.2). It therefore seemed reasonable to include Furui's method in this experiment. In his paper, Furui described a straightforward method of constructing a reference template, as illustrated in Figure 7.1. The first of the 5 reference utterances is designated the initial template, to which the second utterance is then time-aligned by DTW. An average of the two patterns is then taken to produce a new template, to which the third utterance is time-aligned. Again, an average is taken, and the process repeated until all 5 utterances have been combined into a single template.

Note that the length of the final template is equal to that of the initially-chosen utterance, irrespective of whether this utterance is typical in respect of its length.

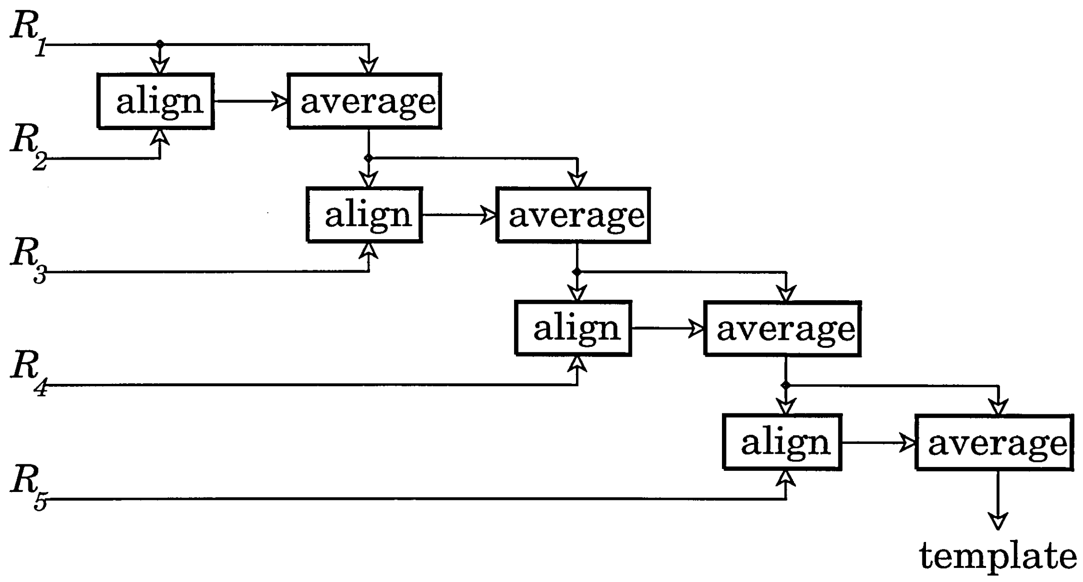

Figure 7.1: Furui's method matches the test token T with a single template formed by successive time-alignment and averaging procedures applied to the reference tokens $R_1, \ldots, R_5$.

7.1.2 Method 2: Bernasconi

The verifier used by Bernasconi (1990) is very similar to that developed by Furui, although the means of template construction is different, as shown in Figure 7.2. Bernasconi's method calculates DTW distances between all pairs of the original utterances so that, in the present case, each of the 5 utterances is compared to the remaining 4. The utterance with the lowest accumulated distance relative to remaining 4 is taken to be the centroid of the group. The remaining 4 utterances are then time-aligned to the centroid, before averaging all 5 to produce the template.

7.1.3 Method 3: mean separate score

One possible criticism of the first two methods is that they attempt to characterise the speaker with a single template based on five tokens. What if one of the reference tokens was corrupted in some way, perhaps by a cough or stumbling over the words? Combining the original tokens into a single template which is influenced by such a 'bad' utterance may actually reduce the level of characterisation. On the other hand, the existence of a bad reference token may actually

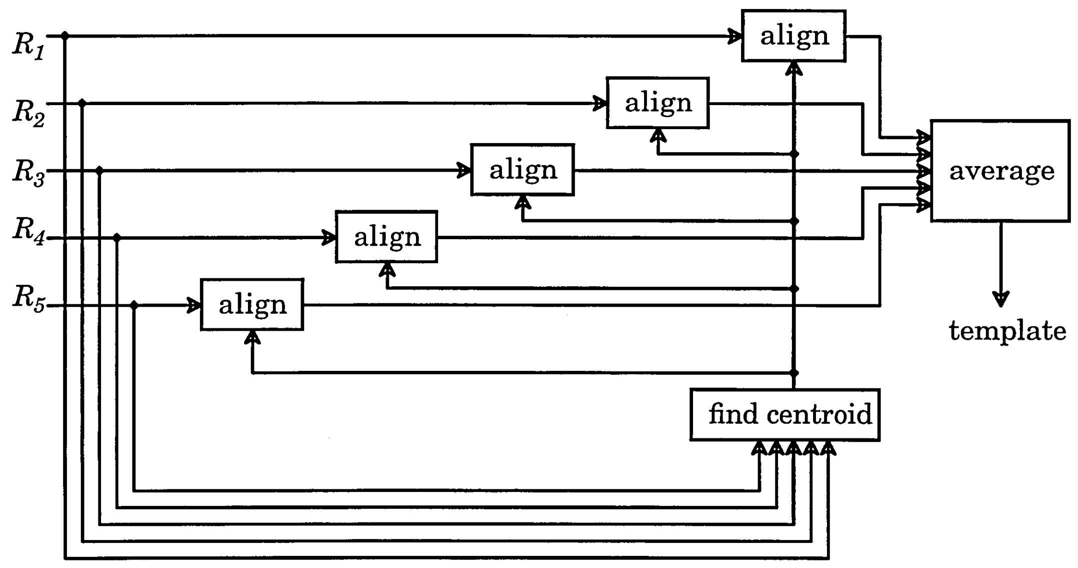

Figure 7.2: Bernasconi's method compares the test token T with a single template based on the reference token which is 'nearest' to all the others (the centroid).

be beneficial because it contains some variation of emotion or voice condition, which, although uncharacteristic of the initial recordings, may manifest itself on other occasions.

For these reasons, it was decided to keep all of the individual tokens separate and compare the test token with each of them. The resulting scores are then averaged to give an overall score. The scheme is illustrated in Figure 7.3.

7.1.4 Method 4: majority decision

By taking the mean of five separate scores, the previous method could be subject to distortion caused by extreme results. For instance, a single reference token could be so bad that it outweighed the other four good tokens, and vice versa. While this is unlikely, it highlights the 'blindness' of method 3 and the possible advantage afforded by a more natural decision-making strategy.

Method 4 offers this by taking the majority of the individual decisions: at least 3 of the 5 scores need to be 'low' for the speaker of the test token to be accepted. A pre-determined threshold for each speaker is used to decide if each individual score is low enough to be accepted. Tests with the small database use thresholds calculated using Furui's formula (4.6) with $a = 3.14$ and $b = 3.14$; with

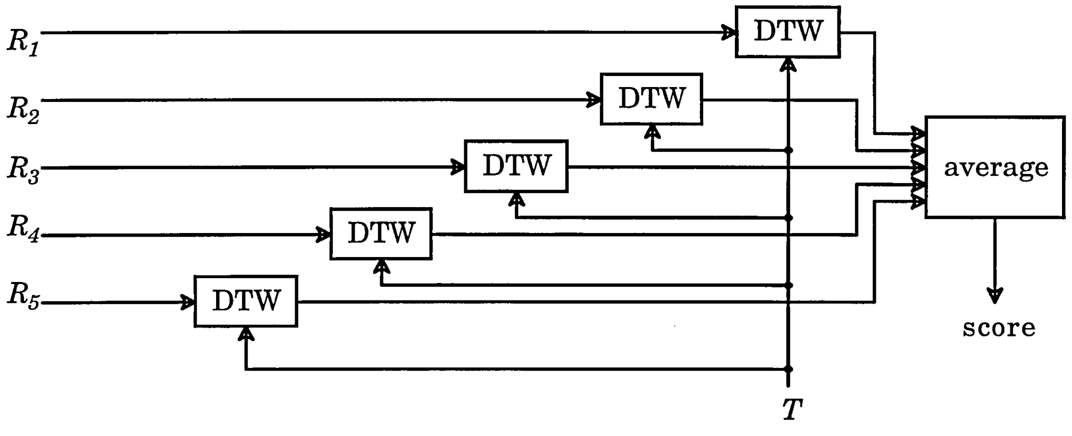

Figure 7.3: The mean separate score method compares the test token T against each of the reference tokens R_1, ..., R_5. The overall score is the average of the individual scores.

the large database, the ranked impostor method (see Section 4.3.2) is employed. The complete majority decision method is illustrated in Figure 7.4.

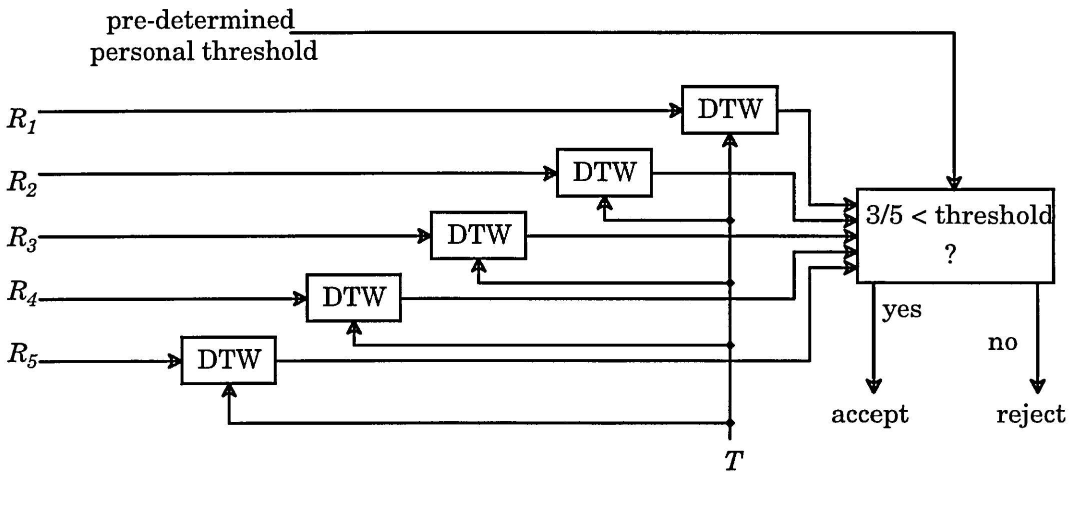

Figure 7.4: The majority decision method matches each reference token R_n with the test token T. The overall accept/reject decision follows the majority of sub-decisions.

The majority decision method formalises the requirements of the mean separate score method in that only 2 bad reference tokens are allowed, regardless of how good the others are. The converse is that 2 good reference tokens will still cause rejection even if the other 3 tokens are only slightly bad.

In order to compare this approach with methods 1–3 on the basis of d', it is necessary to derive some overall score measure for a test utterance so that genuine and impostor score histograms can be constructed. This is done as follows. If the majority decision is 'accept', then the overall score is taken as the average of those (3 or more) individual verification scores which fall below the threshold. On the other hand, if the decision is 'reject', then those (3 or more) scores greater than the threshold are averaged to give the overall score.

7.2 Initial results (small database)

Initial tests of the different methods of template construction used the full-length recordings from the small database (described in Chapter 3) which consists of 10 speakers saying *ABCDEFG* 20 times. For each speaker, the four methods were used to create four templates. In each case, speaker scores were collected and a d' figure computed. These are presented in Figure 7.5. In interpreting these figures, the reader should recall from equation 5.2 that a d' value of 5, for example, indicates that the separation of genuine and impostor distribution means is 5 times the assumed common standard deviation (as assessed from the geometric mean of the individual deviations).

The first 5 speakers (BK–TM) are female while the remaining 5 are male. Although there is an apparent tendency for the males to perform better than the females, the best performing males MH and AS are the author and his supervisor. Further, the best performing female (FS) is a speech therapist. Since 3 of the 10 speakers have experience of speech technology, which seems to influence the results, no firm conclusions may be drawn about gender-related performance differences.

It can be seen that for some speakers (notably, JP, AF and AS), the method of template construction can greatly influence the verifier performance associated with those speakers. With the exception of AS and NT, method 1 gives the poorest results and this is reflected in the low mean d', as shown in Table 7.1. This table also shows the ambiguity rate (defined in Chapter 5) to give an indication of any overlap of the genuine and impostor score distributions. The ambiguity rate is markedly higher (indicating more overlap) for method 1 than for any of the other approaches.

The reason that method 1 performs poorly relative to method 2 is almost certainly a result of the fact that the first utterance is the initial reference,

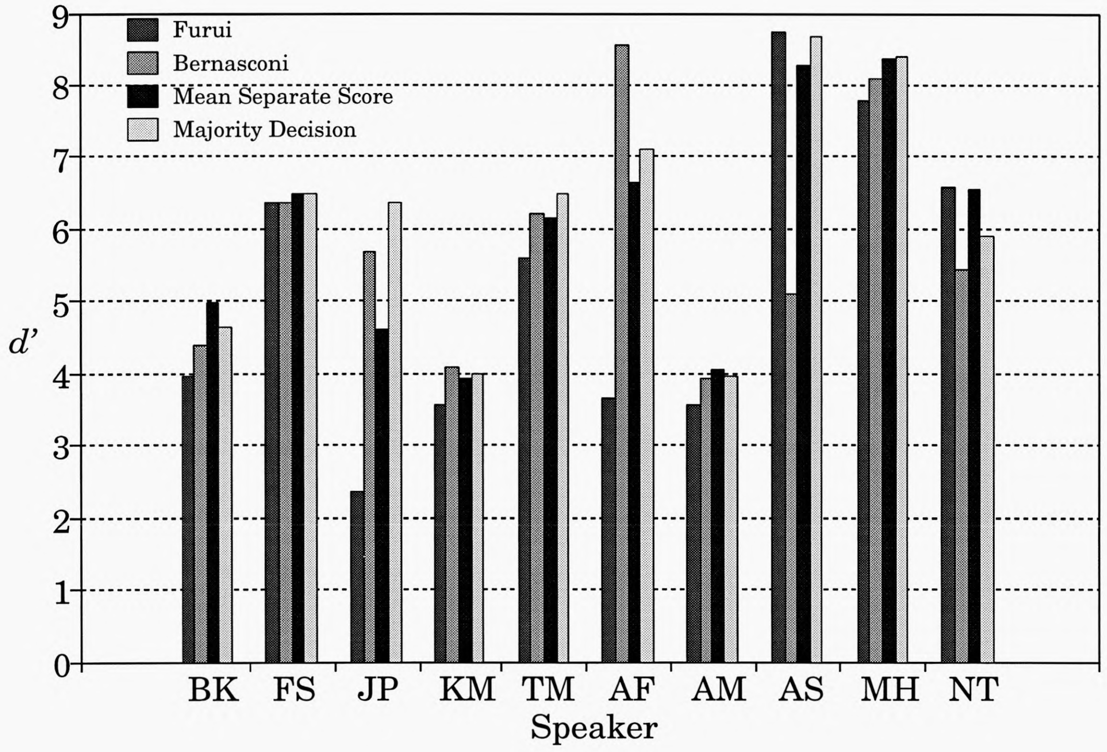

Figure 7.5: The bar charts show the d' calculated for each speaker using 4 different methods of template construction. The first 5 speakers (BK–TM) are female; the remaining 5 (AF–NT) are male. For some speakers, such as AS and AF, the type of template strongly affects performance.

regardless of its suitability. By taking the centroid of the training patterns as the basis for time-alignment, method 2 overcomes this problem. Methods 2 and 3 give roughly comparable results, indicating that choosing the centroid of the group of 5 utterances (method 2) is similar in its effect to averaging scores (method 3). Method 4 produces the best speaker-separation, as indicated by the highest d' averaged over all speakers in Table 7.1.

However, the differences between methods 2, 3 and 4 are not statistically significant, according to the student's t-test (Pipkin 1984). However, using the same test, it can be stated with 90% confidence that any of methods 2, 3 and 4 is significantly better than method 1. This hypothesis is supported by the ambiguity rates in Table 7.1, which suggest that method 1 is less reliable than the others but gives no clear indication as to which is best.

Method 4 (the majority decision) is based on deriving thresholds for each individual speaker. These thresholds offer a basis for estimation of false acceptance and false rejection rates (FA and FR respectively). Table 7.2

Method	Mean d'	Ambiguity Rate (%)
1. Furui	5.2	10.8
2. Bernasconi	5.8	3.47
3. mean separate score	6.0	3.07
4. majority decision	6.2	3.20

Table 7.1: Comparing methods of template construction in terms of d' and ambiguity rate. Method 1 suffers from an emphasis of the first training utterance, regardless of suitability. The other methods are (statistically) significantly better.

shows such FA and FR figures both for the individual matches (to each of the 5 templates) and for the overall score based on the majority decision. This table illustrates, as one might expect, the disparity of performance of the individual training tokens. Also, it shows the gain to be made using the majority decision – in that this rule generally does only a little worse (if at all) than the best-performing reference, but significantly better than the average across reference tokens. For instance, for speaker NT the average FA across the 5 reference tokens is 5.4%, but using the majority decision it is 0%. For the same speaker, the FR averaged across the 5 tokens is 8% while the majority decision reduces this to 1%. The sole exception to this generalization is the FR for speaker AM which averages 12% across the 5 individual reference tokens, but is 13% using the majority decision.

The expectation that method 2 would achieve a good compromise between accuracy and the computational convenience of a single-token template was borne out by the obtained results. Overall, it is apparent from Table 7.1 that the approaches to constructing a template data set that average the reference data to form a single pattern (methods 1 and 2, with an average d' of 5.5) do less well than those that retain the training data in their entirety (methods 3 and 4, with an average d' of 6.1). However, the latter methods entail greater computational complexity in terms of both storage space and processing time at verification.

7.3 Extensive tests (large database)

To find out if the results of the small-scale testing applied for larger populations the tests were repeated using the *Sir Winston Churchill* utterances from the large

Speaker	FA:FR (%)					
	R_1	R_2	R_3	R_4	R_5	Majority decision
BK	5:13	0:20	1:7	0:7	0:47	0:13
FS	0:0	0:0	0:0	0:0	0:0	0:0
JP	1:0	1:0	1:0	6:0	2:20	1:0
KM	0:27	10:0	1:47	9:27	0:27	0:13
TM	1:0	1:0	0:0	4:0	0:0	1:0
AF	0:0	2:0	1:0	0:0	1:0	1:0
AM	4:20	8:7	2:7	8:13	15:13	4:13
AS	0:7	0:0	0:0	0:0	0:0	0:0
MH	1:0	1:0	0:0	0:0	0:0	0:0
NT	27:6	0:7	0:0	0:0	0:27	0:1

Table 7.2: Testing with the small database, false acceptance and rejection rates, FA:FR (%), for all five reference tokens R_1, ..., R_5, and for the majority decision (method 4).

database. These were chosen for two reasons. Being the longest utterances in the database, it seemed likely that the duration effects on performance described in Chapter 6 could be excluded. Also, there are 20 utterances from each speaker which allowed the first 5 utterances to be designated as reference data and the remaining 15 could be used to form test tokens, as in the initial tests. It was felt that 15 self-tests would be sufficient to give a reasonable estimate of the genuine distribution without any jackknifing. For each method of template construction, the following procedure was applied.

Each speaker in turn is designated as genuine and a reference template is constructed. The 15 self-test scores and 3195 impostor scores are recorded and from these data, a d' rating is computed. From each impostor's 15 scores, the average score is calculated and used as a key to sort the list of impostors, as described in Section 4.3.2. This list of average impostor scores can then be used to generate speaker-specific accept/reject thresholds to be used in estimation of the error rates.

The false acceptance (FA) rate is calculated by re-examining all 3195 impostor scores and comparing them with the threshold. The false rejection (FR) rate is assessed similarly but checks the 15 genuine scores. The process is repeated using a few different thresholds. The average FA and FR rates (over all 214 speakers) for each threshold setting are plotted as a graph to show the interaction of the average error rates as the threshold is varied. The intersection of the FA and FR traces can be projected onto the vertical axis to determine the average equal-error rate (EER) and on the horizontal axis to indicate the corresponding threshold setting.

7.3.1 Furui method

Figure 7.6 shows plots of the FA and FR rates as the threshold is varied, according to the average score of the n^{th}-closest impostor. For convenience, the impostor whose average score is used to set the threshold will be referred to as I_t. The crossing point of the two traces shows the EER as approximately 6%, which is achieved when each speaker's threshold is set to the average score of their 6th-closest impostor ($I_t = 6$).

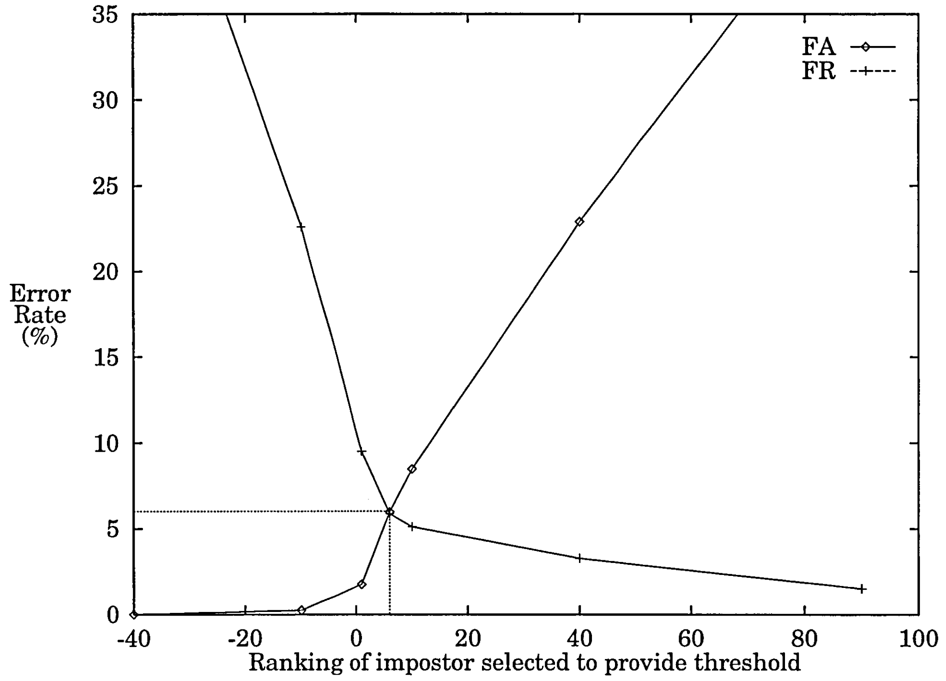

Figure 7.6: Interaction of error rates for the Furui method. The EER is approximately 6% and is achieved by setting each speaker's threshold to the average score of their 6th-closest impostor.

The reader should recall from Section 4.3.2 that low (or negative) impostor rankings map to very high thresholds and tend to cause rejection of all speakers. In this case, the FA rate remains very near to 0% until $I_t = -10$ but at this point the FR rate is still above 20% – unacceptably high. The FR rate only falls to below 10% when $I_t = 1$ and is below 5% when $I_t = 40$ but the rate of decrease is slow, unlike the corresponding increase in the FA rate.

The EER of 6% has been obtained *a posteriori* – after the experiments – but in practice this would be impossible. The sharp rise in error rates either side of the EER point suggests that a system using Furui templates would be very sensitive to poorly-predicted thresholds.

It should be remembered that the interaction of error rates only provides different perspectives on exactly the same data – the distribution of speaker scores remains unchanged and in all cases can be expressed by the same d' rating. In this case, the d' is 4.9, which is an average of the d' ratings for all the speakers in the database. This is close to the corresponding rating of 5.2 using the small database.

The initial tests demonstrated the superiority of the other methods over the Furui method, probably due to the template being based on the centroid of the training data, rather than the first utterance. In these tests, the first utterance of *Sir Winston Churchill* is not the first in the recording session (see Chapter 3) and the negative effects may be less pronounced.

7.3.2 Bernasconi method

When Bernasconi templates are used (see Figure 7.7), the FR rate corresponding to zero-FA is about 17% – better than the Furui method but still poor. The other extreme of performance is actually more likely to be of interest: the FA rate for 0% FR. In this case, for FR rates approaching 2% (which might be acceptable in practice) the FA rates exceed 20%, which would still be a security risk.

The shape and slope of the curves is much the same as those for the Furui method, reflecting the similarity between the two approaches. However, for the Bernasconi method the EER is lower at around 4.5% (achieved with $I_t = 4$) and the average d' of 5.2 is higher. Both of these gains are marginal but together they indicate more reliable performance with Bernasconi templates.

7.3.3 Mean separate score (MSS) method

Figure 7.8 shows the error rate interaction when the mean separate score technique is applied. The picture is similar to that obtained with Bernasconi

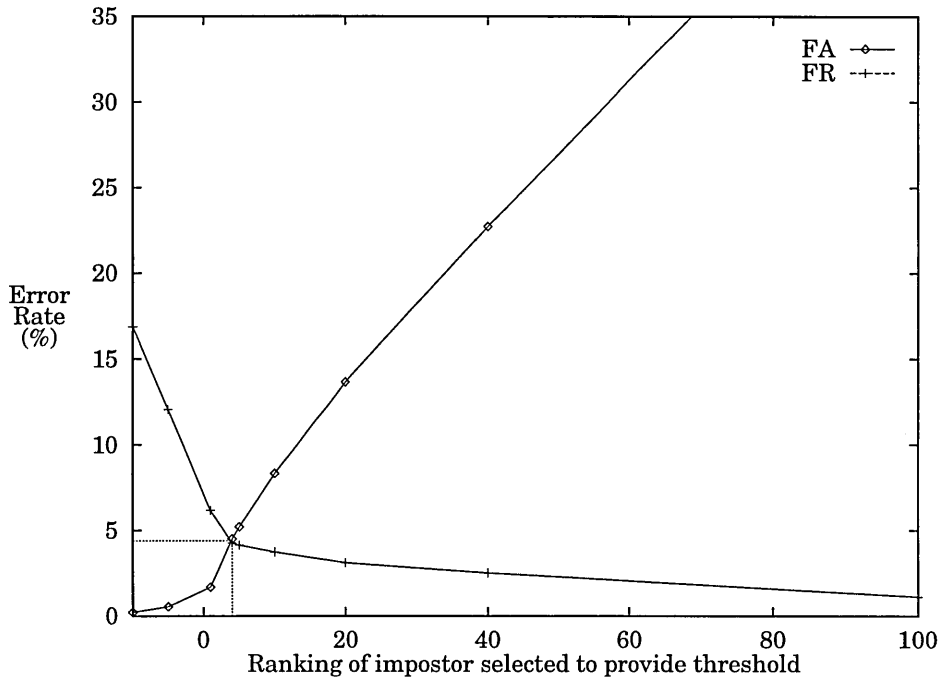

Figure 7.7: Interaction of error rates for the Bernasconi method. The equal-error rate is approximately 4.5% and is achieved by setting each speaker's threshold to the average score of their 4th-closest impostor.

templates and the EER – at just under 5% – is also comparable. This is not surprising; assuming that in most cases all 5 reference utterances are fairly similar, combining them (using the Bernasconi method) should produce a template which is not markedly different from the individual tokens. In such cases, the distances between a test token and the individual reference tokens should be roughly the same. Thus, for speakers who provided consistent reference utterances, Bernasconi's method and the MSS method will produce similar score distributions and this is borne out by the results.

The results for these methods are not identical because there are some cases where the reference tokens are not all alike; one or more of the training utterances being atypical of the group would also account for Furui's method performing poorly although in these tests this is not as evident as in the initial tests. In such cases, Bernasconi's method will effectively prevent any outliers from dominating the overall comparison because they would not be chosen as the centroid and would therefore be subject to considerable alteration as they are time-aligned with the centroid. On the other hand, the MSS method gives each of the individual scores equal priority, thereby allowing the possibility of an outlier corrupting the

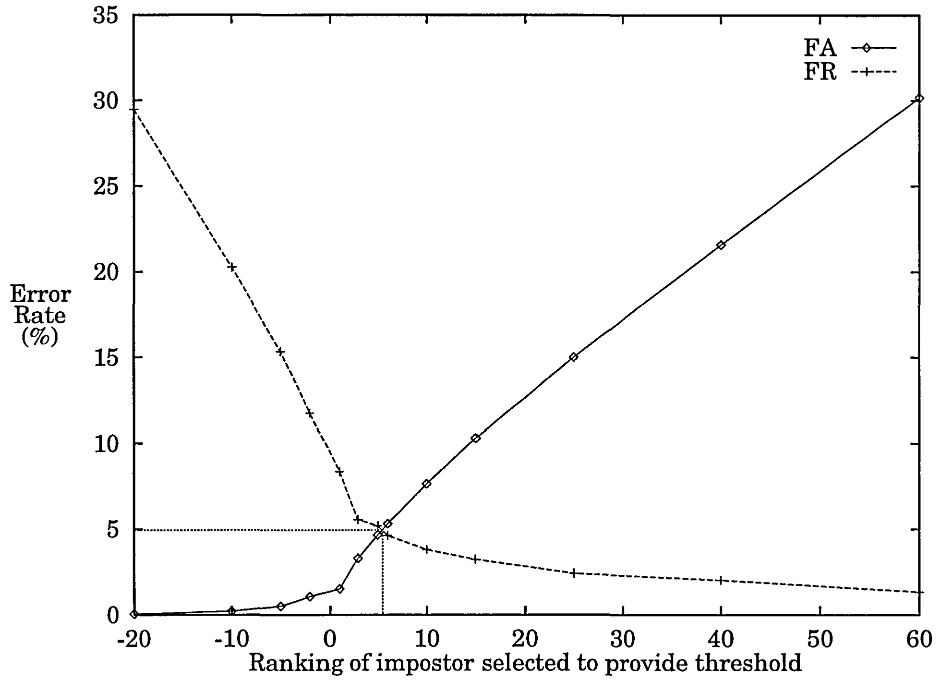

Figure 7.8: Interaction of error rates for the mean separate score method. The equal-error rate is approximately 5% when each speaker's threshold is set to the average score of their 6th-closest impostor.

overall score, although this is unlikely because the averaging process would tend to increase the influence of the grouped scores rather than an outlier.

Figure 7.8 also shows that using the MSS method, the FA rate rises slightly more gently than it does using Bernasconi's method, for corresponding FR rates. For instance, for FR=2% the FA rate is just over 20% but in the case of Bernasconi's template, FA is over 30% for an FR rate of 2%. This highlights a potential benefit of the MSS approach because in any practical implementation, the threshold would be chosen somewhere in the region to the right of the EER point to limit the chances of true customer rejection. All of the error-rate interaction graphs so far have shown diverging rates of change in the error traces in this region: raising the threshold to reduce false rejections causes a disproportionate increase in false acceptances. This effect has been lessened by using the MSS method although it is still present and this highlights the difficulty of pre-determining accurate thresholds.

The MSS method appears to lie between the Furui and Bernasconi methods, both in terms of EER and also average d', which is about 5.0 but the differences are not of great statistical significance (see Section 7.3.6).

7.3.4 Majority decision method

As mentioned in Section 7.1.4, the majority decision method requires a predetermined threshold. For these tests, the ranked technique can provide such a threshold but it may not be reliable because it has to be calculated using one of the other methods. The threshold is based on an impostor's average score which may in fact typically lie in a different range for each template method. For instance, for speaker 192 a typical impostor score might be 400–650 when comparing against Furui templates but this might change to 500–800, say, when using the MSS method. The error rates presented in Figure 7.9 were calculated using the same thresholds as the Furui template method (Section 7.3.1), although for the reasons just stated, it was probably not the best choice. One of the first three methods had to be chosen to provide the thresholds but were the experiment to be repeated, the recommended method would be the MSS method since the range of scores typically matched that of the majority decision approach. Generally, the Furui and Bernasconi methods both produced scores in a different range than that of methods 3 and 4.

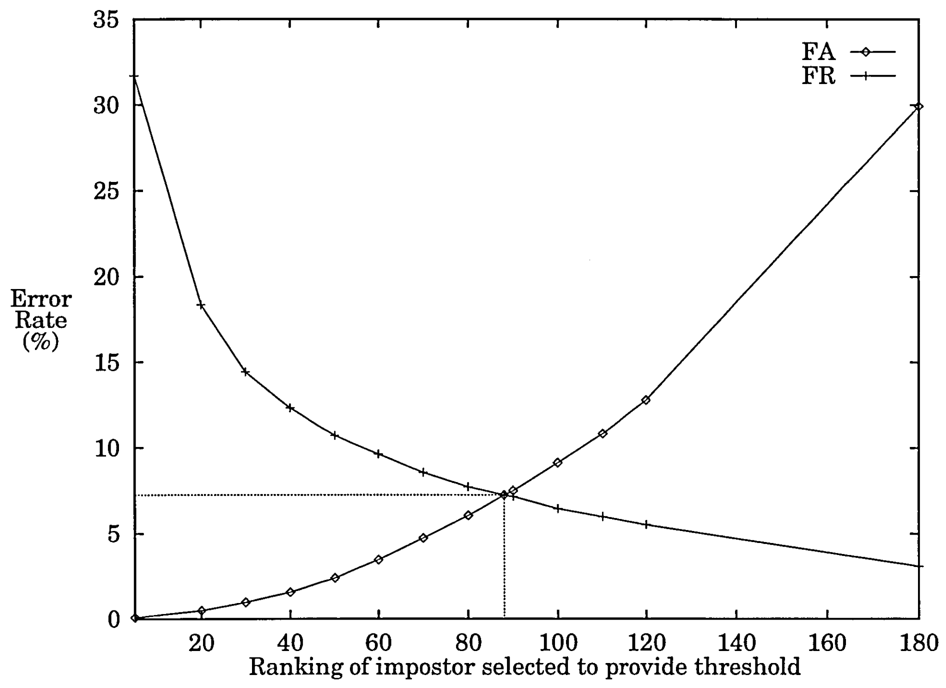

Figure 7.9: Interaction of error rates for the majority decision method but using thresholds taken from ranked impostor files derived using the Furui method. The equal-error rate is approximately 7%.

Interpretation of Figure 7.9 is not easy because the horizontal axis does not mean quite the same thing as it does in the previous diagrams. The slopes of the FA and FR traces seem gentler but this is probably because the horizontal gradations represent relatively smaller differences in the impostors' average score. This is also the reason why the EER point occurs at $I_t = 88$, an apparently much lower threshold than the other methods.

Despite using the 'wrong' method to generate the threshold, the error rates themselves are correct: the threshold is only a number and the percentage of test scores for any given speaker which exceed this number is constant, regardless of how the number is determined. The majority decision method is marginally the poorest of the template methods on the basis of EER which is about 7%. In contrast however, the d' rating is 5.6, the highest of all four methods. Again, however, determining the d' is not as straightforward as before and the figure can be called into question because it is directly dependent on the threshold. As mentioned in Section 7.1.4, the d' is calculated by taking the average of the scores which make up the majority of the decision: 3 or more scores below the threshold for an acceptance or 3 or more scores exceeding it for a rejection. For example, consider the case where the 5 genuine scores are 120, 125, 130, 150 and 160. A threshold of 145 would force an acceptance decision and for the purposes of calculating the d' the score would be 125. If the threshold was then changed to 155, the decision would still be an acceptance but the 'score' would rise to 131.25, which could cause the d' rating to fall, despite the more assured decision (a majority of 4 instead of 3). The d' rating for this method is therefore variable and the figure mentioned above was calculated using thresholds for the EER point.

7.3.5 Three-tries scheme

Following suggestions from NCR that a practical system would allow 3 attempts before rejecting access, the results have been examined to assess the expected false rejection rate for each template method under such a scheme. There is some evidence to suggest that the false acceptance rate would not be increased significantly by allowing repeated impostor attempts because there appears to be only a subset of impostors with voice characteristics resembling those of the genuine speaker. Put simply, most genuine speakers are under threat of 'impersonation' from only a few of their impostors.

To illustrate, Figure 7.10 shows the distribution of speakers who were falsely accepted as speaker 132, using the Bernasconi template method. (After visually inspecting a few speakers' results, speaker 132 was chosen since the effect is clearly visible. Speaker 132 may not be typical in this respect but the results of other speakers demonstrate the same effect to a lesser or greater extent.) Each impostor had 15 attempts and the height of each impulse corresponds to the number of those that were below the threshold. The threshold was set to the average score of the 4th closest impostor for speaker 132 since $I_t = 4$ at the EER point for the whole database when using Bernasconi templates (see Section 7.1.2). All 57 falsely-accepted attempts are due to only 11 of the 213 impostors. This suggests that an impostor's success depends less on the number of attempts than on an inherent similarity to the genuine speaker. The task of identifying, and compensating for, groups of similar speakers at the enrollment stage is examined in Chapter 8.

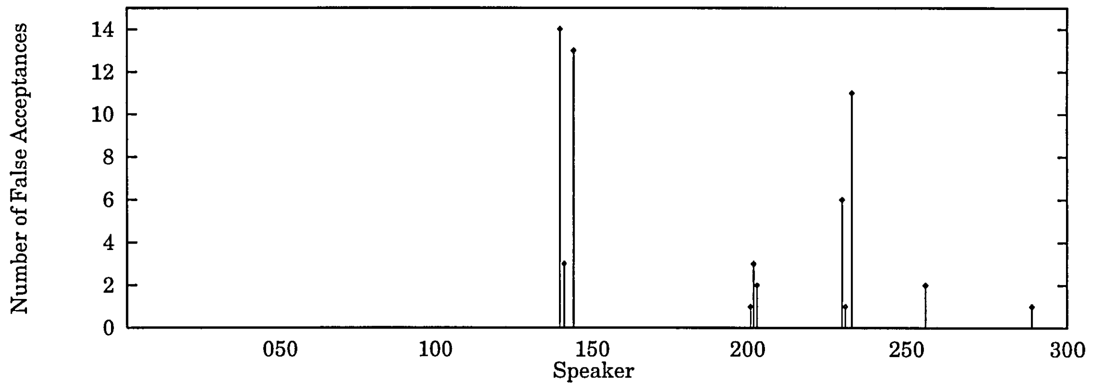

Figure 7.10: The distribution of falsely-accepted impostors for speaker 132, using a Bernasconi template and a threshold set for the database-wide equal-error rate. 11 of the 213 impostors account for all 57 false acceptances.

Using the EER estimated from the graphs in Figures 7.1–7.4, the risk of false rejection using the 3-tries scheme was assessed for each speaker with each of the four template methods. The assessment is based on the number of self-test scores which exceed the personal threshold and would therefore be rejected using a 1-try scheme. For convenience, this will be referred to as n_r. If $n_r < 2$, the speaker would never be rejected under a 3-tries scheme (using this data). If $n_r \geq 3$, then the probability of false rejection P_{FR} is the ratio of the number of possible combinations of 3 from the n_r scores above the threshold to the total number of possible combinations of any 3 scores. This may be expressed as:

$$P_{FR} = \frac{\binom{n_r}{3}}{\binom{15}{3}}$$

Since there is a finite number of self-tests, P_{FR} is quantised (in this case, to 14 levels). The number of speakers in each category is shown in Table 7.3 using each of the template methods. In each case, the vast majority of the database population may be categorised as 'safe' because fewer than 3 of their self-test scores exceed their personal threshold. This is especially pronounced for method 2 (Bernasconi) where 199 speakers (93% of the population) are safe and $P_{FR} > 1\%$ for only 3 speakers. Further work is required to find out if the FA rate indeed remains unchanged (at 4.5%) by the 3-tries scheme. The other methods also show a considerable reduction of the FR rate from their respective equal-error rates but considerably more speakers are 'at risk' than with Bernasconi's templates.

		Template method			
n_r	P_{FR}	1	2	3	4
0–2	0.000	188	199	184	184
3	0.002	10	8	4	7
4	0.009	5	4	7	8
5	0.022	6	1	7	4
6	0.044	2	1	2	3
7	0.077	-	-	1	2
8	0.123	1	-	1	1
9	0.185	1	1	1	1
10	0.264	-	-	-	-
11	0.363	-	-	2	3
12	0.484	-	-	2	1
13	0.629	1	-	-	-
14	0.800	-	-	1	-
15	1.000	-	-	2	-

Table 7.3: Using each template method, the risk of false rejection P_{FR} is calculated based on n_r, the number of each speaker's 15 self-test scores which exceed their own threshold. For each template method, most of the 214 speakers are safe from false rejection when using a 3-tries scheme.

7.3.6 Summary

Table 7.4 summarises the results of the extensive testing and shows, for each method, the mean d', EER (estimated from the graphs of average-error rate interaction) and the average probability of false rejection when using the 3-tries scheme. Presenting averages over the entire database can obscure possibly unacceptable extremes of performance but concise interpretation of the test results is otherwise very difficult, due to their large volume.

Method	Mean d'	Approx. EER (%)	mean P_{FR} (%) with 3-tries
1 Furui	4.9	6	0.57
2 Bernasconi	5.2	4.5	0.14
3 MSS	5.0	5	2.42
4 majority decision	5.6	7	1.09

Table 7.4: A summary of the verifier performance with 4 styles of reference template, after extensive testing. The figures shown are averages across all 214 speakers in the test database.

The means give some indication of the expected performance for a speaker selected at random from the population (assuming the database is representative), although for all 4 methods of reference construction there is a wide variation in performance across the data set. Regardless of method, some speakers produced no errors at all, whilst others always produced errors. As explained in Chapter 3, this could be due to artifacts of the speech preparation procedure, or perhaps an intrinsic confusability of certain speakers with others (reinforcing the "sheep and goats" theory). To illustrate this variation, Figure 7.11 plots the distributions of the d' ratings for the individual speakers for all 4 methods. The dark areas represent densely-packed d' ratings and it can be seen that the Furui method tends to compress these ratings into the most limited range, with only a few outliers at either extreme. The other methods show a single poorly-performing speaker (a data point at the low end of the range, distinct from the rest of the population) and more individuals achieving more emphatic degrees of separation (several data points at the high end of the range). This effect is particularly pronounced with the majority decision method and is reflected in the high average d' of 5.6, clearly separable from the other methods.

In terms of d', the majority decision appears to be more successful than the

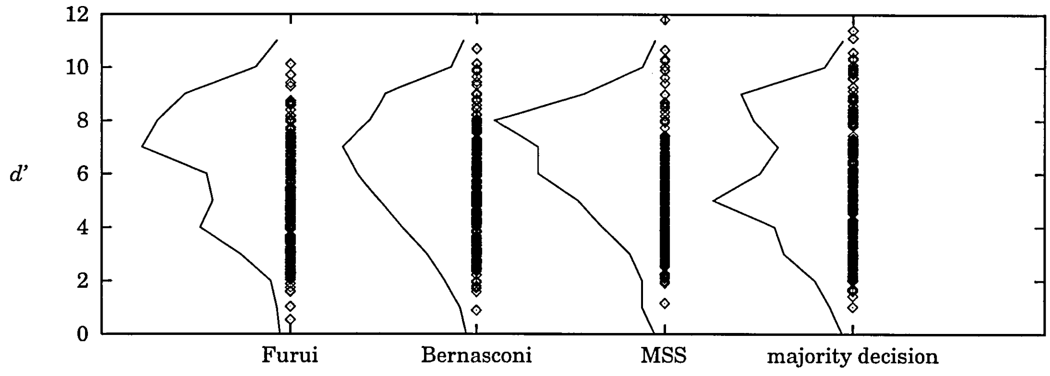

Figure 7.11: Showing the variation of d' across all 214 speakers in the data set. For each reference method the vertical lines are distributions.

other methods, and this is confirmed using tests of statistical significance based on z, the standard normal deviate (Pipkin 1984). It can be stated with over 90% confidence that the method 4 results are superior to those of method 1; the margin over the other 2 methods is less clear, about 70% confidence. Some doubt still hangs over the usefulness of the d' data for method 4 since it is dependent on the threshold but further work is required to test this.

In terms of the equal-error rate, methods 2 and 3 are of very similar ranking and the interaction of error rates for both methods is also similar. This ties in with the results using the small database that these two methods are almost identical, in most cases. The poorer showing for method 4 could be explained by the no-error cases resulting from better score distributions (higher d') than the no-error cases using other reference methods: such cases caused the original need for the d' measure. The EER has the advantage of indicating how many errors really did occur during testing but can be misleading about the general trend, which is more important considering that the scale of these tests is still small compared with the envisaged application.

If a scheme were adopted where a speaker would be allowed 3 attempts before being rejected, the Bernasconi system would give the highest safe-to-unsafe ratio. This is perhaps the most important aspect of all since the implication is that the Bernasconi method successfully characterises the speaker from the 5 enrollment utterances. However, this could be a direct consequence of the test and training data being recorded in the same session.

7.4 Conclusions

Four different methods of template data set construction have been compared; two are established techniques and two were developed for this project, although they are quite intuitive and straightforward.

The results of the initial test phase found superior overall performance when retaining the original reference tokens separate at verification time (Methods 3 and 4), rather than averaging training data to yield a single reference pattern (Methods 1 and 2). The best performing technique (Method 4) applied a majority decision rule to the individual distance scores rather than simply averaging them as in Method 3 (MSS). However, these findings were not statistically significant and so the experiments were repeated with the large database.

The extensive experiments were less conclusive in that no single method performs best for every assessment: method 4 gives the best average d' and methods 2 and 3 produce the lowest EER. However, under a scheme which allows 3 attempts before rejecting access, method 2 (Bernasconi) clearly performs better than the others, with false acceptances below 5% for false rejections below 2%. In practice, this level of performance might be acceptable to the financial institutions.

Although the results for method 1 are not far behind the rest, it may be expected to perform poorly relative to method 2 because the first utterance is always chosen as the initial reference, regardless of its suitability. By taking the centroid of the training patterns as the basis for time-alignment, method 2 overcomes this problem. Methods 2 and 3 give roughly comparable results, indicating that choosing the centroid of the group of 5 utterances (method 2) is similar in its effect to averaging scores (method 3).

Another topic for future investigation is the ease with which the templates of all the methods may be updated to compensate for the ageing process and a variety of moods. Methods 3 and 4 seem ideal for this purpose and could maintain a pool of template data patterns, which could be replaced if 'better' patterns were obtained through ordinary use of the verifier. This would be more difficult for methods 1 and 2 which might become so general that almost anyone would be accepted.

The Bernasconi template is reliable – perhaps not optimal – and will be used in all of the remaining experiments.

Chapter 8

Weighting the Distance Measure

Chapter 4 described the process by which each speaker's voice is represented by a sequence of cepstral vectors. In effect, these vectors describe a path through 'space' which represents the speaker saying a particular utterance. Verification is then a matter of checking whether or not the vectors of the unknown speaker map to points along the path representing the claimed identity. Although it is difficult to apply such concepts to multi-dimensional space, the paths of impostors may perhaps cross, be parallel to, or deviate from the path of the true speaker, to greater or lesser extents. This perspective helps to explain the result of Chapter 6 in that having more data – either longer utterances or more utterances – allows the *typical* difference between paths to be estimated more accurately. Similarly, Chapter 7 found that different ways of defining the true speaker's path (different methods of constructing templates from training data) yield varying levels of accuracy.

Yet another optimisation method is available by emphasising or *weighting* some dimensions of the speaker space (corresponding in some abstract sense to voice features) with respect to others. This is accomplished at the DTW stage, which performs a frame-by-frame comparison of a test token and a reference template. At the frame level, the comparison is based on the distance between the test and reference cepstral vectors, calculated by squaring the differences between corresponding dimensions and summing for all n (=12, in this case) dimensions. The differences in each dimension i may be emphasised according to their perceived importance by multiplying by a weighting factor, w_i. The general expression for the local distance d between a test frame t and a reference frame r

is then:

$$d = \sum_{i=1}^{n} w_i(t_i - r_i)^2 \tag{8.1}$$

Previous speaker verification studies using weighted cepstral distance measures, such as those summarised by Velius (1988), have been successful in reducing error rates. However, the derivation of the weights has been based on assumptions rather than using an optimisation procedure to yield the best practical results. This chapter begins with experiments using the small database to compare the following weighting methods:

- inverse-variance function,
- inverse-expected-difference function,
- integer-valued weights chosen either by genetic algorithm (GA), or purely randomly;
- unweighted, or 'uniform weighted' (all weights set to 1).

The first two methods calculate the weight of each dimension as a deterministic function of the coefficient values of that dimension, and attempt to minimise inter-session intra-speaker variability. The third method finds weights which achieve a maximal separation between the genuine speaker and the impostor set. For comparison, the results of the unweighted verifier (as used in previous chapters) are also included.

8.1 Initial tests

The initial tests use the full-length utterances (*ABCDEFG*) from the small database described in Chapter 3. For each of the 10 speakers, 5 utterances are used to make Bernasconi-style templates (see Chapter 7) and the remaining 15 utterances are used for testing. In each case, the DTW process involved in making the templates uses unweighted distance measurements. The weights generated are speaker-specific.

8.1.1 Inverse variance

The commonly-used *Mahalanobis distance* (O'Shaughnessy 1987, pp. 425–426) is based on the inverse of the covariance matrix for the individual cepstral coefficients for each speaker. If the cepstral coefficients have been orthogonalised – for instance, by a Karhunen-Loève transformation – the only non-zero elements of the matrix are those of the main diagonal, corresponding to the inverse of the variance of each coefficient. The work required to calculate either the full covariance matrix or the orthogonal cepstral coefficients has led to simpler weighting functions using only the inverse of the variance of each coefficient, originally by Tohkura (1986) but also used by Xu, Oglesby & Mason (1989) and Gaganelis and Frangoulis (1991).

In such a scheme, there is an assumption that features which remain consistent over an entire utterance, or over several utterances by the same speaker, may characterise that speaker. This theory was tested by observing the utterance-wide variance of each of the 12 time-normalised cepstral coefficients for every speaker's test utterance. For each dimension, the resulting variances were averaged (over all 15 utterances) and the reciprocal of the average was used as the weight. This produced a single 12-element weighting vector for each speaker. The ith element of any speaker's weighting vector is given by:

$$\frac{1}{w_i} = \frac{1}{n}\sum_{u,f} t^2_{u,f,i} - \left(\frac{1}{n}\sum_{u,f} t_{u,f,i}\right)^2 \qquad (8.2)$$

where $t_{u,f,i}$ is the time-normalised cepstral coefficient in the ith dimension of frame f of test utterance u. The number of frames in each utterance may be different; n indicates the actual number of coefficients summed. The weights for all 10 speakers are plotted on the same graph (Figure 8.1) to demonstrate the similarity of the function across the speakers. Clearly, the higher dimensions are weighted more heavily than the lower ones.

8.1.2 Inverse expected-difference

This function, as used by Furui (1981), normalises the dimensions by their so-called *expected* differences after time alignment, for the intra-speaker case. This ensures that cepstral coefficients may contribute equally to the overall distance measure, regardless of numerical scale, which is not an indicator of utility. Test

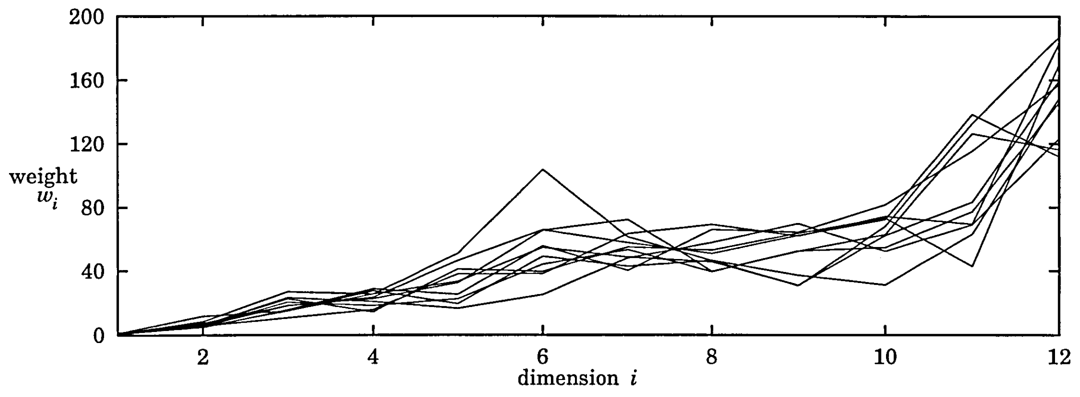

Figure 8.1: Each trace shows one speaker's weights calculated with the inverse-variance function.

utterances from each speaker were time-aligned with their reference template and for each of the 12 dimensions, the average distance between corresponding frames was calculated over the entire utterance. Averaging over all 15 test utterances produced the expected differences for each dimension. The reciprocals of these were used as the weights. The ith weight for any speaker is given by:

$$\frac{1}{w_i} = \frac{1}{n} \sum_{u,f} \left(t_{u,p_u(f),i} - r_{f,i} \right)^2 \tag{8.3}$$

where $t_{u,p_u(f),i}$ is the time-normalised cepstral coefficient in the ith dimension of the frame in the uth test utterance when aligned with r_f, frame f of the speaker's reference template. (DTW is used to define $p_u(f)$, the warp-path relating the f frames of the template to test utterance u.) Although the number of points in the warp-path may vary for each of the test utterances, the actual number of distances in the summation is n. The weights are plotted in Figure 8.2 for all 10 speakers to highlight the similarities with the inverse-variance case. Again, the highest weights are the strongest.

8.1.3 GA-generated weights

Using any of the traditional methods, each individual weight in the vector is calculated by applying the same, deterministic function which, despite having a basis in theory, nonetheless restricts the available solutions. An alternative is to allow the individual weights to be derived independently, yet applying them collectively as a vector. The genetic algorithm, suggested by Holland (1975) and examined in depth by Goldberg (1989) is one such method. Spillman (1993) gives

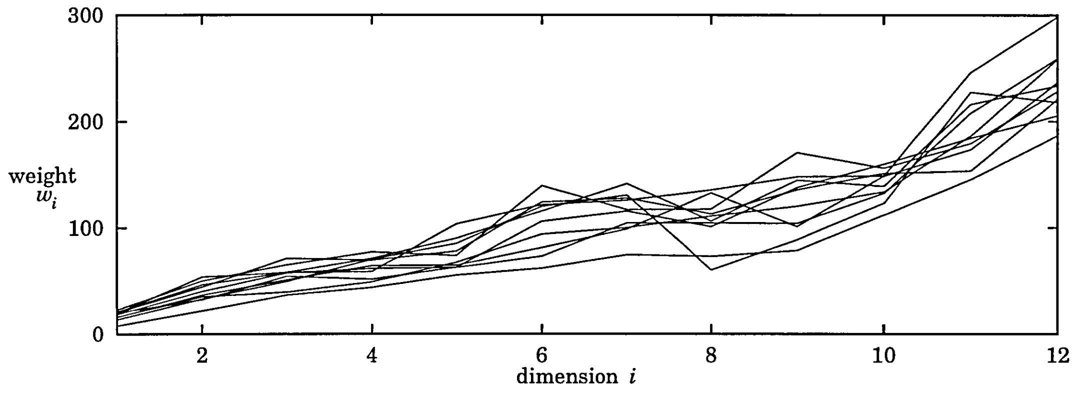

Figure 8.2: The inverse-expected-difference weighting function, plotted for all speakers.

a concise introduction to genetic algorithms; the details are reviewed briefly here.

The GA is a directed search method based on the perceived evolutionary processes of nature. To search a large solution space, a group (*population*) of candidate solutions is generated randomly and the best-performing candidates are selected to generate the next population. The cycle continues for a specified number of generations or until an acceptable solution has been found. Each candidate solution is usually encoded as an integer in the form of a string of bits, representing the presence or absence of some feature, analogous to the chromosome in nature. In its simplest form, the GA consists of only three processes applied to the current group of chromosomes to generate the next population.

- Selection: the suitability (*fitness*) of each chromosome is assessed and used to choose *parent* chromosomes to shape the next generation.

- Mating: pairs of parents are combined to produce two new *child* chromosomes. The commonest approach is known as *crossover*, where each parent is divided at some randomly chosen point into a head and a tail. The child chromosomes are obtained by combining the head of one parent with the tail of the other, and vice versa.

- Mutation: individual bits in each new chromosome are inverted according to some pre-set probability. This can sometimes make the child exist in an entirely different region of solution space than either of the parents, thus allowing the algorithm to escape from regions of local – but perhaps not global – optima.

The GA is not guaranteed to find an optimal solution but during latter generations most members of the population should exhibit properties characteristic of good solutions. In this application, the goal is to separate maximally the genuine and impostor score distributions but the extent of the optimum separation cannot be known in advance. Instead, the optimisation criterion is to find a weighting vector which produces significantly better separation than the unweighted case. The GA is an appropriate method of doing this, as no assumptions are made about which dimensions are important for verification, or about the properties of cepstral coefficients of those dimensions. The GA has no constraints on either the starting point or the extent of the search space of solutions, and should therefore be capable of finding high-performance weighting vectors, should they exist.

Implementation

For the genetic algorithm to generate weighting vectors, each weight was allowed to vary between 0 and 15 so that it could be encoded simply using 4 bits. By concatenating all 12 of the 4-bit weights, the weighting vector could be encoded as a 48-bit chromosome. In the general case, when using n-dimensional cepstral vectors, and allowing each weight to be quantised to m levels, there are $2^{n \log_2 m}$ different weighting vectors from which to choose. In this case, the range of choice is 2^{48}, which is clearly too large to search exhaustively[6] as there is no guarantee (quite the opposite!) that the solutions are ordered (in terms of verifier performance) within this range. Traditional search methods, such as gradient descent, are therefore probably unsuitable but this type of problem is appropriate for solution using the GA.

A vital consideration when using a GA is the specification of the *objective function* which determines the fitness of a candidate solution. As mentioned above, this is assessed in terms of the separation of the score distributions for the genuine speaker and the impostors. Since fitness has to be repeatedly computed during each generation, while the GA is running, weighting vectors were assessed using the F measure (described in Chapter 5) as this requires less computation than does d'. However, d', with its stronger resistance to outliers, was preferred for finally scoring what appeared to be the best weighting vectors.

[6]In fact, there are less than 2^{48} *unique* solutions because some of the vectors are linearly dependent. For instance, vectors $(1,1,1,1,1,1,1,1,1,1,1,1)$ and $(2,2,2,2,2,2,2,2,2,2,2,2)$ both give identical separation statistics/error-rates when used as the weighting vector.

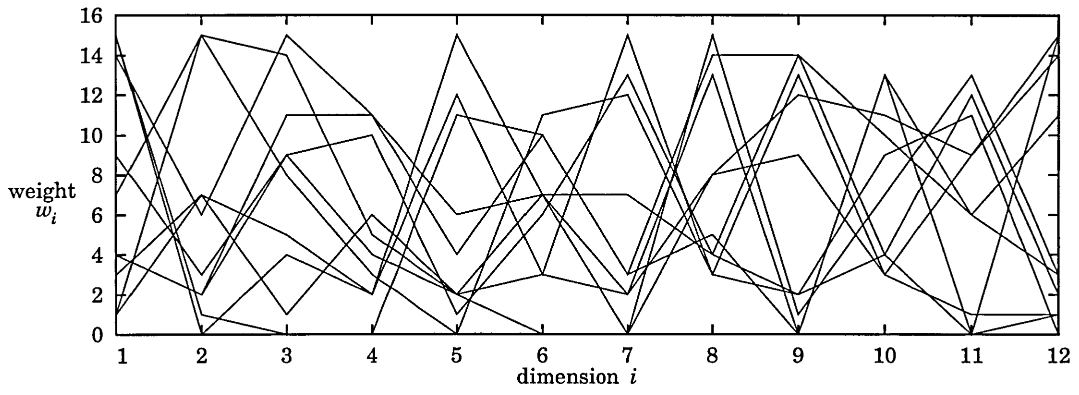

Figure 8.3: The best weighting vectors found by the GA, for every speaker. Unlike the traditional weighting functions, no order or pattern is apparent.

Following the 'Simple Genetic Algorithm' of Goldberg (1989), the initial population of candidate vectors was generated by randomly setting bits in the chromosomes to 1 or 0 with equal probability. At each generation, only the three basic operators were employed: roulette-wheel selection, single-point crossover (with probability P_c) and mutation (with probability P_m). Various configurations were tried before settling on a population size of 32 with $P_c = 0.7$, $P_m = 0.02$ and, unless stated otherwise, the program ran for 50 generations. Other settings did not find better solutions within an acceptable time. The algorithm was coded in C and executed on a standard 80386-based PC with a floating-point co-processor.

For comparison with Figures 8.1 and 8.2, the best weighting vectors found by the GA for all speakers are plotted on a single graph: Figure 8.3. The pattern of weights that was apparent in the previous figures is not repeated. Indeed, the weights appear to be uncorrelated among all the speakers. This may be taken as evidence that a different weighting vector is needed for each individual. It is also interesting to see that, in general, the whole dynamic range of the weights is used, unlike the traditional methods.

8.1.4 Results

For each of the 10 speakers, the speaker separation was assessed using speaker-specific weights generated by the three methods described above. In the case of the GA, the vector used was the best one discovered within the first 50 generations on the basis of the F measure. Table 8.1 shows the d' ratings for each case and the corresponding results for the unweighted verifier are given for comparison.

Speaker	Inverse-Variance	Inverse-Expected Difference	Best GA	Unweighted
AF	7.516	8.308	9.048	8.563
AM	3.920	4.088	4.115	3.927
AS	7.977	6.051	8.935	5.109
MH	8.798	7.674	8.498	8.106
NT	6.045	6.183	8.139	5.454
BK	4.215	4.911	4.847	4.392
FS	6.119	6.235	6.476	6.379
JP	5.322	4.398	5.398	5.687
KM	3.862	3.963	5.179	4.066
TM	5.626	6.237	6.798	6.215
Average	5.940	5.805	6.743	5.790

Table 8.1: The verifier was tested using four different weighting methods. In each case, the d' is shown for each speaker. Only the GA-derived weights offer significant performance improvement.

The d' rating is improved by inverse-variance weighting for only 3 speakers (compared to the unweighted results). Perhaps this is not surprising since cepstral coefficients which vary widely over an utterance may actually vary slowly from frame to frame: the variance would not then be a good indicator of the significance of the dimensions of speaker space. The inverse-expected-difference weighting scheme does slightly better as it improves the separation for 5 speakers; the average d' rating is marginally lower but the differences are not statistically significant.

In contrast, 9 of the 10 speakers benefit from the GA-derived weights; the only speaker who does not is JP, for whom none of the weighting schemes improve the speaker separation. At the other extreme, AS benefits markedly from any kind of weighting. The GA-derived weights significantly out-performed the other three methods (tested using the paired t-test), with an average d' of 6.743, a gain of nearly 14% on the next-best method: the inverse-variance.

8.1.5 GA versus random optimisation

To find out if the reason for the success of the GA was the random influence, a program was written which simulated the GA without any of the standard

Speaker	Best GA	Best random
AF	9.048	9.548
AM	4.115	4.848
AS	8.935	9.129
MH	8.498	10.472
NT	8.139	8.737
BK	4.847	5.440
FS	6.476	7.577
JP	5.398	6.188
KM	5.179	4.354
TM	6.798	7.540
Average	6.743	7.383

Table 8.2: d' of score distributions for each speaker, comparing the GA-produced weights with random weights.

operators; this is equivalent to generating each population completely at random, with no regard for previous generations and fitness levels etc. The program was run for each subject in the database, producing 50 populations of 32 randomly-generated weighting vectors; again a record was kept of the best vectors for each subject, and the average fitness of each generation. Table 8.2 compares the best vectors for each speaker, as found by the GA and the random program.

From Tables 8.1 and 8.2, it is apparent that the randomly-chosen weights actually give greater separation between the average speaker and impostors than weights selected by any other method. This is quite surprising in itself; but also it raises questions about the mechanisms in the GA for converging on optimal solutions. Perhaps the GA found good vectors solely because of the randomness of the search? One way to test this is to examine the evolution with time ('generation') of average population fitness for both the GA and the random program. Figure 8.4 plots these quantities for a single speaker, for a cycle of 50 generations.

Figure 8.4 shows that, for this speaker, average population fitness (measured using the F function) follows a slightly upward trend, whereas the average fitness of each randomly-generated population remains at a fairly steady level. This pattern is typical of all the speakers in the small database, but for this speaker in

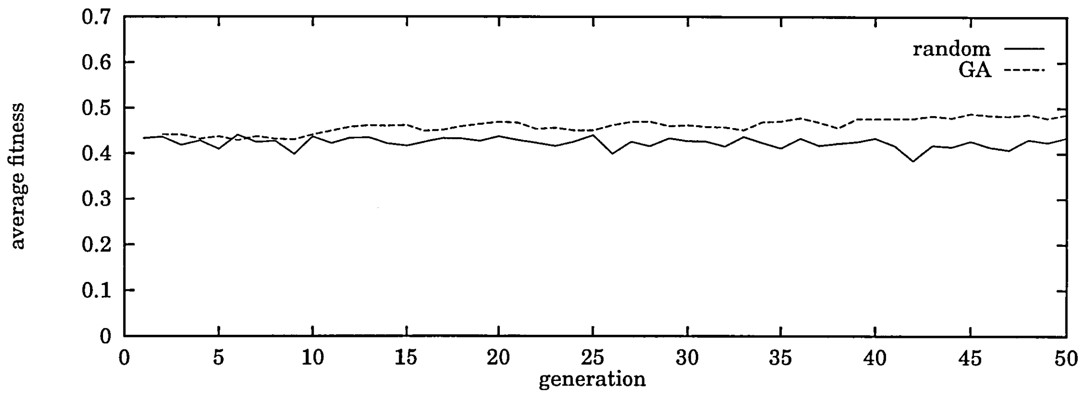

Figure 8.4: Fitness (average F rating of the population at each generation) for both the GA and the random program, for speaker KM.

particular, the upward trend is pronounced[7]. To some extent, this result restores faith in the GA: it finds collections of good vectors at the expense of finding the single best-performing vector. However, for a banking application where fast training procedures would be essential, this rate of convergence may not be acceptable.

However, the slow convergence of the GA may well be due to the deliberately simple configuration. The concatenation of binary-encoded weights makes detection of successful *schemata* (similarity templates) difficult because at certain points in the chromosome, i.e. at weight boundaries, the adjacency of bits is meaningless. (For instance, the least significant bit of one weight is unlikely to act in conjunction with the most significant bit of a neighbouring weight.) Conversely, unless the crossover point is between weight boundaries, there is a risk that the disruption will cause the loss of good schemata rather than lead to better ones. Perhaps the use of position-constrained multi-point crossover would have helped (De Jong and Spears 1990). In any case, there seemed to be little purpose in experimenting further with the GA configuration when the random weights generation performed so well.

8.2 Extensive tests

Although the random vectors gave the best results, obtaining the weights for any given speaker entailed using the other speakers as impostors during training.

[7]An extra experiment was conducted for this speaker where the GA was left to run for 200 generations; the average fitness continued to climb, finishing at about $F = 0.480$.

In the ATM application it would be impossible to obtain voice samples of every impostor but training with some standard fixed-size database would be feasible. Such a database would also be useful for determining each individual's accept/reject threshold. By adopting the ranked-impostor threshold method (described in Chapter 4), a small subset of the database comprising potentially close impostors could be used to develop an optimised weighting vector for a speaker enrolling onto the system.

To test that such weights would not be too specific to the training group i.e. they would not increase the likelihood of accepting 'unseen' impostors, an experiment was conducted using the large database with speaker-specific, randomly-chosen vectors. The experiment is an extension to that carried out in Chapter 7 using the *Sir Winston Churchill* recordings and Bernasconi templates (the best-performing single-template method). Using the ranked impostor listings from that experiment, each speaker's closest five impostors were identified and their utterances were collected to form the impostor test-set for that speaker.

Following the findings of Section 8.1.5, the weighting vectors were chosen using a simple random search procedure, as follows. Each of the 12 weights was generated as a random number between 0 and 15. Incorporating this weighting vector into the DTW stage, the F rating was assessed based on the scores of the genuine speaker's 15 self-tests and the 75 (5×15) scores of their own impostor test-set. The best vector found after 50 repetitions was then used to obtain new scores for the genuine speaker and the *entire* impostor set. (Trial and error suggested that 50 repetitions were sufficient to find good vectors, although obviously there is no guarantee that they are even near-optimal or indeed that any of them would improve on the unweighted verifier's performance.) Based on these scores, new F and d' ratings were calculated for the genuine speaker and a new ranked impostor file was also generated. Thresholds derived from this file were applied to the new test scores and error rates for the weighted verifier were recorded. The entire procedure was performed for each of the 214 speakers in the database.

8.2.1 Results

To demonstrate the effect of incorporating weighting vectors, the results presented here are analogous to those in Section 7.3.2. Figure 8.5 shows the error-rate interaction as the accept/reject threshold is varied. The shape of the traces is similar to those in Figure 7.7 but there are some differences. The EER (at 2%) is substantially lower than the unweighted version (at 4.5%). The FA rate only approaches 0% when $I_t = -40$, in contrast to the unweighted case; conversely, the FR rate quickly falls below 1% as the threshold is increased but the unweighted verifier never reaches this level.

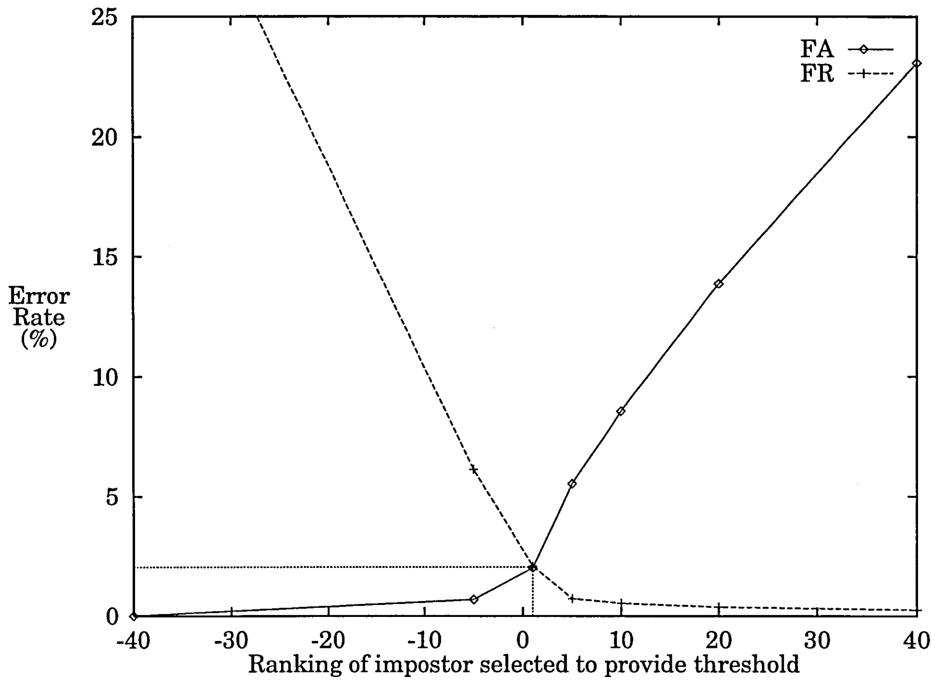

Figure 8.5: When weighting vectors are incorporated into the DTW stage of the verifier, error rates are lower than the corresponding unweighted verifier. Here, the EER is reduced to 2% from 4.4% in the unweighted case.

Over the whole database, the d' ratings are significantly higher ($>$ 99% confidence, using the z-test) with the use of weighting vectors. This is illustrated by Figure 8.6 which shows the distribution of d' ratings for both the weighted and the unweighted verifier. The use of weights has boosted many of the poorer ratings, resulting in a higher mean (6.4 from 5.2) although the higher results are largely unchanged. This is confirmed by an examination of the results for all speakers, ranked by either the F or d' ratings. The top of the weighted and unweighted listings are substantially the same, apart from a slightly different ordering of the 'best' speakers. In contrast, most of the 'worst' speakers (in the unweighted case) have such markedly improved ratings (after weighting) that they no longer occupy the bottom of the list; the speakers who have taken their places also have improved results but to a lesser extent.

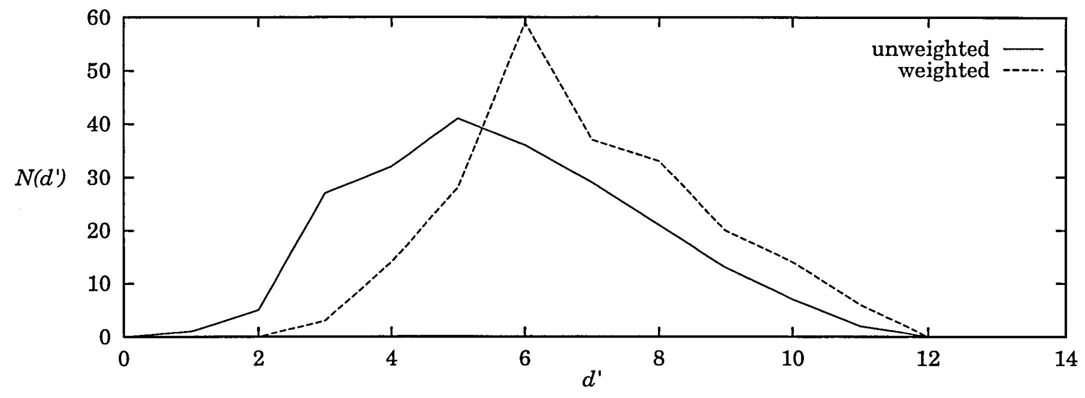

Figure 8.6: The distribution of d' for all speakers in the test set for both the unweighted, and the weighted verifier. The use of weights changes the distribution from skewed to normal, with improved results across the entire test set, especially for the poorest-performing speakers.

This agrees with the error-rate analysis in that the main gains appear to be in reducing the incidence of false rejection; the 'worst' cases without weighting were those where the genuine distribution exhibited one or two extreme outliers but was otherwise compact. Weighting tends to produce tighter clustering of the self-test scores, thus reducing the FR rate and improving d' and F: the majority of speakers now have F ratings exceeding 0.5 which indicates complete separation.

For comparison with Section 7.3.5, the results of applying a 3-tries strategy using both the weighted and the unweighted verifier are shown in Table 8.3. In this scheme, each speaker can have 3 verification attempts before rejection, so only those speakers with 3 or more scores above their own threshold are at risk. The thresholds are generated using the ranked impostor method such that the average speaker has equal FA and FR rates (as shown in Figure 8.5). Incorporating the weights allows 207 speakers to be classified as 'safe', 8 more than the unweighted case. Only 4 speakers have more than a 1% chance of false rejection, i.e. $P_{FR} > 0.01$.

The false rejection rate (averaged across the entire test-set) using the 3-tries scheme can be calculated using the data from the Table 8.3 as 0.29% for the weighted verifier and 0.14% for the unweighted verifier. This somewhat surprising result highlights the fact that for some speakers, the performance diminished with the introduction of the weighting vector. This is apparent from Table 8.3 which shows that two speakers have 10 self-test scores above their threshold. Inspection of the results for these subjects (speakers 250 and 284) reveals that none of

n_r	P_{FR}	Verifier	
		Unweighted	Weighted
0–2	0.000	199	207
3	0.002	8	-
4	0.009	4	3
5	0.022	1	1
6	0.044	1	1
7	0.077	-	-
8	0.123	-	-
9	0.185	1	-
10	0.264	-	2
11	0.363	-	-
12	0.484	-	-
13	0.629	-	-
14	0.800	-	-
15	1.000	-	-

Table 8.3: The risk of false rejection P_{FR} is calculated based on n_r, the number of each speaker's 15 self-test scores which exceed their own threshold. The number of speakers at risk is reduced to 7 when weighting vectors are used.

the 50 randomly-generated vectors performed better than the uniform weighting. (The F and d' ratings using the best vector were lower than the unweighted case.) It is possible that allowing more 'guesses' would cause better vectors to be found but since only two speakers require it, 50 seems an appropriate default: a better strategy would be to use the results of the unweighted verifier as an initial reference against which the results with weights are compared.

Table 8.4 summarises the performance with and without weighting vectors.

	Average		
	F	d'	EER (%)
Unweighted	0.453	5.215	4.4
Weighted	0.539	6.359	2.0

Table 8.4: The verifier performance is superior using weighting vectors, as indicated by the higher average F and d' ratings and lower equal-error rate.

8.3 Further analysis

To investigate the mechanism by which the results are improved, the score distributions for speaker 123 – previously the worst speaker – are shown in Figure 8.7 before and after weighting. In the unweighted case, two of the self-test scores for speaker 123 are extremely high (approximately 400) and cause ratings of $F = 0.05$ and $d' = 1.8$, despite the other 13 acceptable scores. However, the use of the weighting vector apparently corrects the mismatch in these utterances and the genuine distribution becomes unified and completely separate from that of the impostors.

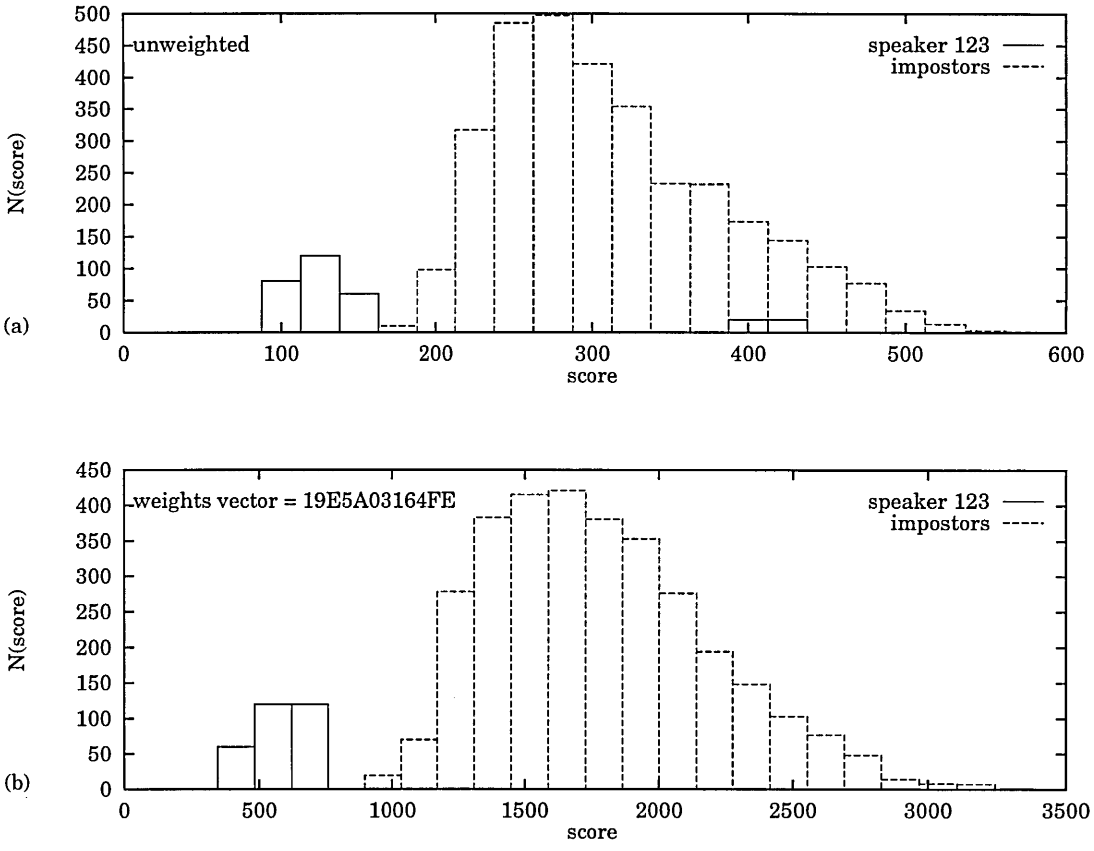

Figure 8.7: (a) Without weighting vectors, the results for speaker 123 are the worst in the test set: the two outlying scores over 400 cause very poor F and d' ratings. (b) The use of weights brings these scores back into the correct range and the genuine and impostor distributions become completely separate. (For clarity, the genuine scores are shown on a ×20 scale.)

Listening to all 20 of speaker 123's recordings revealed that 3 of the 5 which make up the template – including the centroid, or dominant utterance – have incorrectly-determined start points, leaving only *Winston Churchill*. (The reader should recall from Section 4.1.3 that (only) the initial /s/ of *Sir Winston Churchill* was sacrificed for the sake of uniformity from the automatic start-point detection routine.) It therefore seemed likely that the reference template did not include the *ir* but only *Winston Churchill*. Of the 15 self-test utterances, 2 were short and did not include the initial *ir*, but these were not the 2 problem utterances, both of which contained the full *ir Winston Churchill* without any extraneous material.

The DTW program was altered so that frame references and local distances along the warp-path were written to a file, and the self-tests for speaker 123 were repeated. It was found that the local distances for the two problem utterances were consistently higher than those of the other 13 utterances, at every stage of the warp-path: the poor (final) scores were not due to isolated periods of mismatch, such as might be caused by recording artifacts. It seemed likely that the problem utterances were being mismatched straight from the start of the warp, so the first few frames of these warp-paths were compared with those of the other utterances, as illustrated in Table 8.5.

From the table, the warp-paths for utterances 41 and 48 support the hypothesis that the reference template does not contain the *ir* of *Sir* because these utterances are matched to within one frame of the start of the template and still achieve a low score. The DTW procedure can compensate for the other utterances containing the full text by starting the warp 'late', i.e. matching the start of the reference template to frame 6, 7, 8 or 9 of the test utterance, in effect bypassing the *ir*. However, for the two problem utterances (103 and 121), the DTW routine has erroneously matched the start of the template with the start of the test utterance.

Perhaps this mismatch is not surprising. Most of the speakers in the database have some variety of Scottish accent and say *Sir Winston Churchill* as /sɪr wɪnstən ʧʌrʧhɪl/. Since the initial /s/ is removed by the start-point detection routine, in the problem utterances the /ɪ/ of *i̲r* is being matched with the /ɪ/ of *Wi̲nston* in the reference template. After manually resegmenting the problem files to exclude all of the *ir* and then repeating the DTW comparisons with the template, the scores were drastically reduced (both to about 110) and were comparable to the rest of the genuine scores.

Utterance	Text	Start of path	Score
41	*Winston Churchill*	2, 3, 3, 5, 6	119
42	*ir Winston Churchill*	7, 9, 9, 11, 12	92
48	*Winston Churchill*	2, 3, 4, 6, 7	115
66	*ir Winston Churchill*	9, 10, 10, 12, 14	137
68	*ir Winston Churchill*	7, 9, 9, 11, 12	100
73	*ir Winston Churchill*	8, 10, 10, 12, 14	132
81	*ir Winston Churchill*	7, 8, 9, 10, 11	101
103	*ir Winston Churchill*	1, 2, 2, 3, 3	**409**
110	*ir Winston Churchill*	6, 7, 8, 9, 11	105
113	*ir Winston Churchill*	7, 9, 9, 10, 12	99
116	*ir Winston Churchill*	7, 9, 9, 10, 11	113
119	*ir Winston Churchill*	7, 9, 9, 10, 12	114
120	*ir Winston Churchill*	6, 7, 9, 10, 11	135
121	*ir Winston Churchill*	1, 2, 2, 3, 3	**376**
134	*ir Winston Churchill*	7, 8, 9, 10, 12	92

Table 8.5: Speaker 123's self-tests. Utterances 103 and 121 produce scores which lie far outside the distribution of the others. The warp-paths of all the other full-text utterances begin a few frames 'late' – i.e. not at frame 1 or 2 – so that they match the reference template.

To understand how the weighting vector can accomplish this without resegmenting, consider the start of the DTW procedure in both the unweighted and the weighted case for utterance 103 by speaker 123, as illustrated in Table 8.6. (In the following discussion, (x,y) denotes the point in the warp-grid representing the distance between frame x of the reference template and frame y of the test utterance.) In the unweighted case, the first frame of the reference template is matched against the first frame of the test utterance, as indicated by the lowest local distance of the 9 frames searched. Since DTW is a form of dynamic programming, the fact that (1,1) has the lowest local distance does not force the finally-chosen warp-path to start there. However, to a large extent, it does determine which areas of the grid will be searched. In contrast, by multiplying the local differences by the appropriate weighting vector, test frame 7 has a marginally lower distance and is chosen as the start of the first estimate of the warp-path. Further examination of the warp grids revealed that starting at (1,7) offers more scope to find better matches throughout the grid, and therefore achieves a lower final score, more in keeping with the other non-problem utterances.

In effect, starting the warp at (1,7) allows the matching procedure to ignore the extra material (the *ir*) at the start of the test utterance. (Actually, it is the *lack* of material at the start of the reference template which is being ignored.)

Test	Local distance	
frame	unweighted	weighted
9	22.8	180.4
8	12.2	82.0
7	9.1	56.8*
6	14.7	94.7
5	20.4	181.4
4	14.8	137.1
3	8.2	86.9
2	21.3	182.3
1	6.5*	63.7

Table 8.6: The first column of the weighted and unweighted DTW warp grids for utterance 103 by speaker 123. Use of the weights makes reference frame 1 slightly 'closer' to test frame 7 than test frame 1. (The star indicates the lowest local distance in each column.) Starting the warp from (1,7) compensates for the material missing from the start of the reference template.

A similar correction may be obtained by modifying the DTW procedure to be more tolerant of timing mismatches. For each frame of the reference template, a group of frames in the test utterance is searched for the best match. In terms of the warp-grid, the centre of this vertical search range is the test frame which best matched the previous reference frame. By setting the search range to ±8 instead of ±4, the routine manages to quickly find the correct portion of the grid through which to 'travel', as shown in Table 8.7. The centre of the first search range is the point (1,9) and the lowest distance of the first column is found to be (1,1), as before. This dictates that the centre of the search range for the next column is (2,1) but only the points above this are examined, since there are no points below. Because the search range is now 8 instead of 4, the minimum at (2,9) is found. The search range in the next column is therefore centred on (3,9), which is on a discernible path of low distances through the grid.

The grid also illustrates the maximum local search range property of the *local minimum* method of DTW used here. The search range in column 4 is centred on the point (4,10) but frames 14–18 remain with their default (unmeasured)

Test frame	Reference frames 1	2	3	4	5
18	∞	∞	∞	∞	∞
17	28	∞	∞	∞	∞
16	46	∞	∞	∞	∞
15	32	∞	∞	∞	65
14	27	∞	∞	∞	65
13	43	∞	∞	59	46
12	33	∞	∞	55	23
11	30	∞	50	33	23*
10	33	∞	36	21*	36
9	23	13*	16*	47	78
8	12	15	29	96	110
7	9	38	66	86	100
6	15	45	65	73	92
5	20	34	47	55	81
4	15	38	50	66	96
3	8	23	40	71	85
2	21	32	56	∞	∞
1	6*	18	∞	∞	∞

Table 8.7: By revealing more of the warp-grid (calculating more of the local distances), the DTW routine is able to find the correct path without using weights. The minimum distance in each column is denoted by a star; in column 2, the extra search range allows the routine to recover the correct path after the mismatch in column 1.

distances of ∞. It is not worthwhile calculating the local distances at these points because their *accumulated* distances would have to be assessed along paths which include points which still have their (default) infinite values. This is due to the slope constraints imposed upon the warp-path (see Section 4.2) which dictate that a path to point (4,14), for instance, would have to pass through (3,12), (3,13) or (3,14). The accumulated distance at each of these points is also unknown because they, in turn, have infinite-valued predecessors. In such cases, working backward through the grid would add many more calculations, so points such as (4,14) remain un-assessed.

In this section, three methods of fixing the problem utterances for speaker 123 have been found: manually resegmenting the file, using a weighting vector, and extending the DTW search range. In each case, the main effect seems to be a correction of the mismatch with the first frame of the reference template. If this is the *only* effect, using arithmetic alone, it should be possible to calculate a weighting vector which would separate maximally the two contending frames of the test utterance. As an example, consider t_1 and t_7, vectors representing respectively the first and seventh frames of utterance 103 by speaker 123. The differences between the coefficients of these test vectors and the corresponding coefficients in the reference vector are shown in Table 8.8. Summing the columns for each vector $a = (t_1 - r_1)^2$ and $b = (t_7 - r_1)^2$ gives, respectively, the local distances for points (1,1) and (1,7) in Table 8.6.

	dimension											
vector	1	2	3	4	5	6	7	8	9	10	11	12
$a = (t_1 - r_1)^2$	0.3	0.1	0.5	0.1	2.9	0.2	0.0	0.6	0.0	0.1	1.6	0.0
$b = (t_7 - r_1)^2$	0.0	0.0	0.4	1.2	0.3	0.8	0.2	0.1	0.2	4.3	0.8	0.8
$a - b$	0.3	0.1	0.1	-1.1	**2.7**	-0.6	-0.2	0.5	-0.2	**-4.2**	0.8	-0.8

Table 8.8: To force the warp-path to begin at (1,7) instead of (1,1), weights could be derived which maximised the distance between the first test frame (t_1) and the first reference frame (r_1). Also, the distance between t_7 and r_1 should be minimised. Both these aims could be achieved by emphasising dimension 5 and de-emphasising dimension 10.

Vectors a and b differ substantially in only 2 dimensions: in dimension 5, t_1 is further than t_7 from r_1, but the reverse is true in dimension 10. Clearly, to make the DTW procedure favour t_7 over t_1, dimension 5 should be emphasised (weighted heavily) and dimension 10 should be de-emphasised. The differences in the other dimensions are comparatively small, so weighting them would have little effect. Row 3 of Table 8.8 therefore represents a set of ideal weights to distinguish t_7 from t_1. Raising the level of these weights to remove negative values, expanding the range to [0,15] and rounding to integers gives the vector $w_{103} = (9, 9, 9, 6, 14, 7, 8, 10, 8, 0, 11, 7)$. A similar process for the other problem utterance (121) produced the vector $w_{121} = (7, 6, 10, 2, 2, 0, 6, 6, 6, 1, 15, 10)$ and both of these arithmetically derived vectors are plotted with the randomly selected weights, w_{rand} in Figure 8.8.

Neither of the weighting vectors produced by optimising for a single utterance is markedly more similar than the other to the randomly selected weighting vector, which was effectively optimised for all 15 utterances. There are differences and similarities amongst all three vectors but there seems to be a general agreement

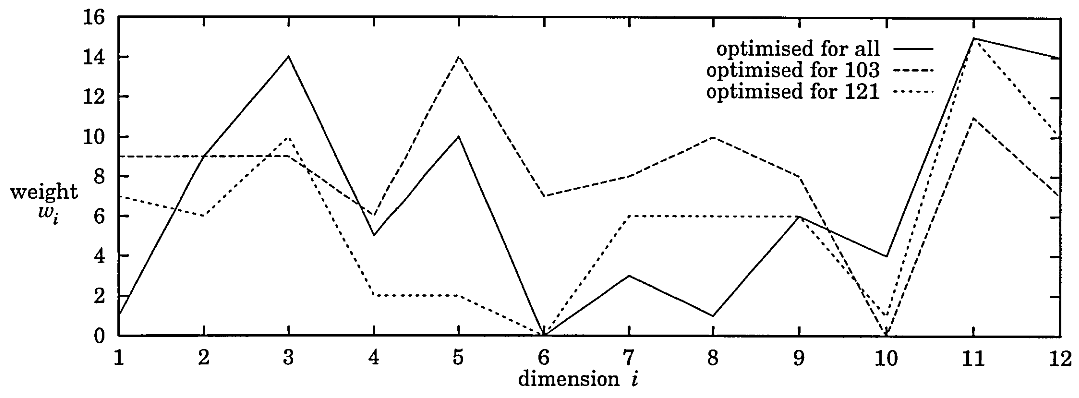

Figure 8.8: Three weighting vectors for speaker 123. Vectors w_{103} and w_{121} have been derived arithmetically, to force the correct starting point of the warp path for the problem utterances, 103 and 121. The directed random search scheme was used to generate w_{rand}.

about dimensions 4, 9, 10 and 11. The actual performance of the verifier using these vectors was assessed, and the results are shown in Table 8.9.

weights	F	d'
w_{rand}	0.611	6.110
w_{103}	0.549	5.695
w_{121}	0.093	2.144
unweighted	0.050	1.806

Table 8.9: Without weights, the verifier performs poorly for speaker 123. Weights optimised for individual problem utterances, w_{103} and w_{121}, offer very different performances. The best discrimination is obtained with w_{rand}, found by the directed random search.

Although the verifier performance using w_{103} is almost as good as using w_{rand}, w_{121} offers little improvement over the unweighted verifier. Examination of individual scores and warp-paths shows that although the starting point for utterance 121 is correct, utterances 73 and 81 start at the wrong point and accumulate very large scores. It seems therefore that choosing weights to fix specific mismatches between the template and individual utterances cannot be relied upon to improve performance generally. The random search routine uses all of the self-test utterances to find weights which fix problems without introducing others, by directing the search using the F measure, which also checks for reduced impostor scores. This then raises the question of how weights generated in this manner would affect utterances (by the genuine speaker) which have not been included in the optimisation.

A similar analysis conducted for the second-worst speaker (098) found that a single utterance (66) was responsible for the very poor F and d' ratings when no weights were used. With randomly selected weights this outlier was restored to the body of the genuine distribution by encouraging the warp-path to start at frame 1 instead of frame 8. With weights derived arithmetically to force this starting point, the score for one of the other utterances (73) was increased, creating another outlier. Although the F and d' ratings were much improved over the unweighted case, complete separation of the genuine and impostor distributions was not achieved, as it had been using randomly selected weights.

Examination of the resulting warp-path showed that it had not been corrupted – it started at the correct point – but that 2 points in the path contained very large local distances. Indeed, these 2 points contributed to more than a third of the score (distances accumulated over the path). These large distances were due mainly to dimensions 6 and 11, both of which were weighted heavily using the arithmetically derived vector; the randomly selected vector also emphasises dimension 11 but is zero in dimension 6 and therefore prevents the distance from being large. This is more evidence that a weighting vector derived for a single utterance in isolation is not guaranteed to work for other utterances.

Interestingly, when the wide search-range version of the DTW procedure was applied (without weights) to speaker 098's utterances, 3 outlying scores were produced. In each case, the starting point for the warp-path was wrong although the weights made the correct point the most attractive. However, as the DTW algorithm effectively searches backwards as well as forwards, it is possible for the path to pass through points which are not the lowest local distances. Apparently, a wider search range offers opportunities for the warp-path not only to correct itself, but also to stray.

Generally, the use of weights offers most improvement (of separation statistics) for 'poor' speakers but there are also marginal improvements for many of the speakers who were already completely (or mostly) separate from the impostors. This is at odds with the hypothesis that weights only allow errors to be fixed. One of the best speakers is 288, whose unweighted d' and F ratings were 9.9 and 0.568 respectively. These both rose by about 7% with the use of randomly selected weights. With or without weights, all of the warp-paths start from the correct point. (The reference template was *Winston Churchill* and all the self-test utterances were *ir Winston Churchill.*)

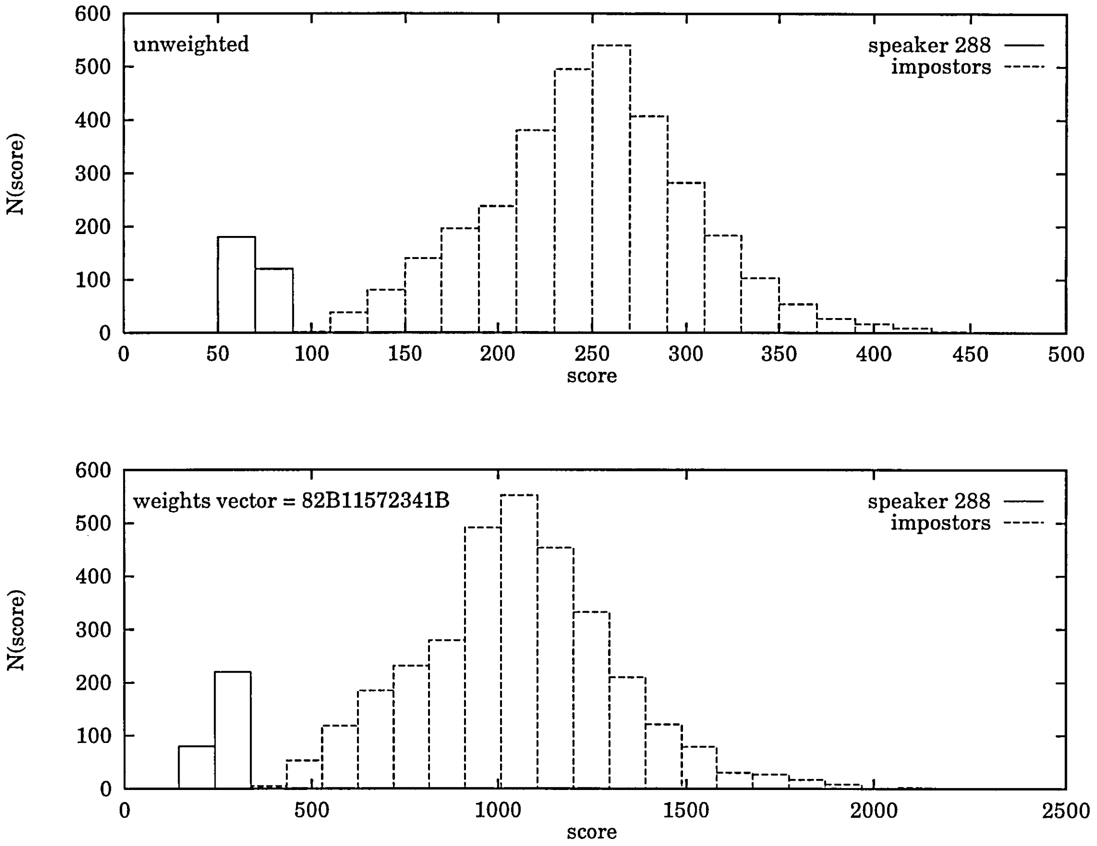

Figure 8.9: Does the use of weights really improve the score distributions for speaker 288?

Examination of the score distributions before and after weighting (Figure 8.9) actually suggests that perhaps the weights have not improved the separation, in terms of assessing the likelihood of error. The genuine speaker's distribution is skewed right and the relative gap between the two distributions has decreased. Perhaps the capability of d' and F to express separation gets saturated at some point, so that minor changes to already distinct distributions are not correctly reflected in the ratings. Further selection of weights directed by these ratings may then be unreliable.

Doubt therefore remains over the usefulness of weighting vectors for speakers who are already distinct from their impostors. However, the cases of speaker 123 and 098 suggest that weighting vectors help the DTW procedure to match correctly similar phonemes at the start of the test and reference patterns, and maybe this also happens all along the warp-path. The effects may be less visible – a lower local distance at one point, perhaps – but equally valid for non-problem utterances.

8.4 Conclusions

Initial experiments with the small database found little performance gain when incorporating traditional weighting functions into the DTW distance measure. In contrast, when the weights were integer-valued and chosen by a genetic algorithm, the gains were significant. A fairer comparison might have included the Fisher linear discriminant which, like the GA, optimises on the training set: both the inverse-variance and the inverse-expected-difference weighting functions treat each dimension in isolation, thus limiting their influence. However, it was found that the GA's success could be at least matched by a simple random search routine and rather than developing the admittedly simple GA configuration, random selection was used to find weighting vectors for each speaker in the large database.

The experiments with the large database were exactly the same as those in Chapter 7 using Bernasconi-style templates, so it was easy to study the effect of using the weighting vectors. Significant improvements were indicated by the generally higher F and d' ratings and the lower equal average-error rate; the main beneficiaries seemed to be those speakers with the poorest unweighted results. Close examination of two cases revealed that the weighting vectors were correcting for inconsistent detection of the starting points of the utterances. This seems to be achieved by forcing the DTW algorithm to make the correct match between a frame in the reference template and one of two confusible frames in the test utterance. This process is easiest to detect at the start of a warp-path but may be active at every step, thus allowing improvement in cases where the match is already good. This would help to explain the results of the small database test because the utterances were edited manually so start-point detection errors should be small.

This simple, practical explanation should not make the use of such weighting vectors less attractive; quite the opposite actually, since their influence can be explained rationally to potential customers (e.g. financial institutions) without reference to the somewhat nebulous realms of multi-dimensional speaker space. There are two main implications for a practical system: storage space and training time. The identity token for each speaker would need to include the weighting vector as well as the reference template and acceptance thresholds but the (likely) implementation on a 'smart card' would allow this. After enrollment, the weights would need to be generated using a small subset of some database but this would be required to determine the speaker-specific thresholds anyway.

The experiments with the large database were conducted with the intention of discovering whether the weights were too specific to the impostors used in the optimisation process. However, the converse was completely overlooked; i.e. since the weights were based on an optimal clustering of all of the self-tests, they may not apply to other utterances by the true speaker. Again, the paucity of data for each speaker makes it hard to test this aspect but it certainly merits future examination.

Notwithstanding, it has been shown to be feasible and worthwhile to generate a weighting vector through a random selection process for every newly-enrolled speaker to the system.

Chapter 9

Optimising the Verifier

Previous chapters have examined isolated modifications to the verifier configuration with a view to improving performance. Chapter 6 showed that the amount of data to work with may be as important as utterance length, and that the summation of scores for individual words (e.g. the digits) could work well. Chapter 7 indicated that performance gains could be registered by keeping the reference utterances as separate tokens and comparing each of them to the test token. However, for less complexity the Bernasconi composite template seems adequate and only marginally less effective. Chapter 8 demonstrated that a random selection procedure could find weighting vectors for each speaker which could compensate for start-point detection errors and make the self-test scores more consistent with no adverse effect on the impostor distribution. This chapter describes an attempt to combine these techniques in a practical implementation for an ATM network.

9.1 Weighting vectors and digits

There is currently much interest in using the digits to form the verification phrase, especially if they are chosen randomly at every access attempt (Rosenberg *et al* 1991, de Veth and Bourlard 1994). Fraudulent access then requires sophisticated recording/replay equipment, the opportunity to record the customer saying all (ten or so) digits; and the customer's bank card.

The experiment in Section 6.3.2 demonstrated that a series of digits can be used in place of a single long utterance as the verification phrase. It was decided to follow (Jacobs and Setlur 1994) and emulate the current ATM security system of personal identification numbers by using 4-digit verification phrases. To discover an optimum level of performance for a practical implementation, an experiment was conducted using the *best* 4 digits, each with its own weighting vector, for each of the 214 speakers in the large database.

9.1.1 Finding the best digits

As a result of the experiments described in Section 6.3.1, file was produced with a line for each speaker showing the 12 digits ranked according to d' rating (best first). A fragment of the file is shown in Figure 9.1.

```
001 SEVEN NOTHING FOUR TWO ZERO NINE FIVE ONE THREE SIX EIGHT OH
002 NOTHING FOUR SIX THREE EIGHT FIVE ZERO TWO SEVEN ONE NINE OH
003 NINE SEVEN THREE FIVE ONE SIX FOUR NOTHING ZERO OH TWO EIGHT
                                 ⋮
289 ZERO TWO FIVE EIGHT NINE SIX NOTHING SEVEN FOUR THREE OH ONE
290 NOTHING TWO NINE SIX ZERO FIVE EIGHT SEVEN THREE ONE OH FOUR
```

Figure 9.1: The first and last few lines from a file containing the relative performance of the digits for each speaker. For example, speaker 001's best and worst digits are *seven* and *oh*, respectively.

For each speaker, the best 4 digits were used to simulate a verification procedure which prompts the user for a sequence of words, spoken individually. For example, speaker 001's best four digits are *seven, nothing, four* and *two.* Table 9.1 shows the 'popularity' of each of the digits for selection as one of each speaker's best 4 digits. The most popular best digits are *zero* and *six* whilst *oh* appears in only seven of the sequences.

9.1.2 Weights for each digit

As well as using the best 4 digits, the scheme was further optimised through the use of weighting vectors, specific to each speaker and each digit. In each case, the vector chosen was the best of 50 random selections, using the process described in Section 8.1.5. As before, each speaker's nearest 5 impostors were used with the F measure to determine the best weighting vectors. Ranked impostor listings were not readily available for the digits so the listings for *Sir Winston Churchill* were used instead, making the assumption that the rankings would not change much for different utterances. This assumption may be unjustified but

digit	Occurrence as	
	best digit	one of 4 best digits
zero	33	105
six	33	101
nine	16	91
nothing	21	82
seven	19	82
five	17	77
two	20	75
eight	22	69
four	16	62
one	9	57
three	8	48
oh	0	7

Table 9.1: Each digit's usefulness gauged by frequency of occurrence in the best 4-digit strings for each speaker. The digit *zero* is joint first (along with *six*) as a speaker's best digit, and is included somewhere amongst the best 4 for almost half of the 214 speakers.

the findings of Section 8.2.1 suggested that the weights mainly affect the genuine scores and the impostor scores merely act as an 'anti-target' for the optimisation process. In other words, the weights produce a tight clustering of self-test scores, whilst preventing *any* impostor scores from approaching the genuine distribution.

9.2 Method

Due to problems with some utterances failing to survive the automatic end-point detection routine, data from only 211 speakers were used. There are 10 utterances for each of the 4 digits in each speaker's best sequence. The first 5 of these are used to form a reference template using the Bernasconi method, since the experiments of Chapter 7 suggested that it performed well and simplified the comparison process. All 10 utterances of each digit are used as test tokens, including the 5 used to make the template. Thus, it may be expected that 5 of the genuine test scores for each digit are very low and the other 5 are somewhat higher. This causes an artificially low d' and so these absolute results are not directly comparable to results elsewhere in this report. (However, since the same process is applied to all speakers, it should be possible to draw conclusions about any trends which may appear.) Jackknifing might have solved this problem but time did not permit the further investigation of the technique needed to address the reservations expressed in Chapter 6.

The d' rating is calculated in two stages. First, the genuine speaker's score distribution is generated, by comparing each test utterance with the template for that digit. The first score for the first digit is added to the first score for the second digit and so on, until there are 10 4-digit score-sums, as shown in Table 9.2. The process is repeated for all (210) impostors, producing $210 \times 10 = 2110$ impostor scores. A record is kept of the d' rating for each length of sequence, i.e. 1, 2, 3 and 4 digits.

utterance	*seven*	*nothing*	*four*	*two*	sum
1	188.1	229.6	148.3	174.5	740.4
2	170.6	209.1	120.2	229.1	729.1
3	171.0	181.1	113.4	164.7	630.1
4	126.1	246.1	123.9	316.0	812.1
5	116.5	517.3	144.0	118.0	895.8
6	232.4	240.8	211.1	147.0	831.4
7	280.1	415.8	151.9	154.8	1002.7
8	256.4	577.2	377.4	180.4	1391.3
9	303.1	512.3	207.7	162.9	1186.1
10	276.8	434.3	818.4	209.1	1738.7
mean	212.1	356.4	241.6	185.7	995.8
std. dev.	66.7	150.0	217.2	55.4	347.3

Table 9.2: The individual scores for speaker 001's best four digits, and the sums for each of the 10 *seven, nothing, four, two* 'sequences'.

For each speaker, a different weighting vector is used for each digit. Since the weighting vector multiplies the distances between frames, the score distributions for each digit are not guaranteed to lie within similar ranges, as shown in Figure 9.2. Normalising each set of scores by the magnitude of the weighting vector would be possible but is unnecessary because the effect is the same for both genuine and impostor scores and it is the scores, not the distributions, which are added together.

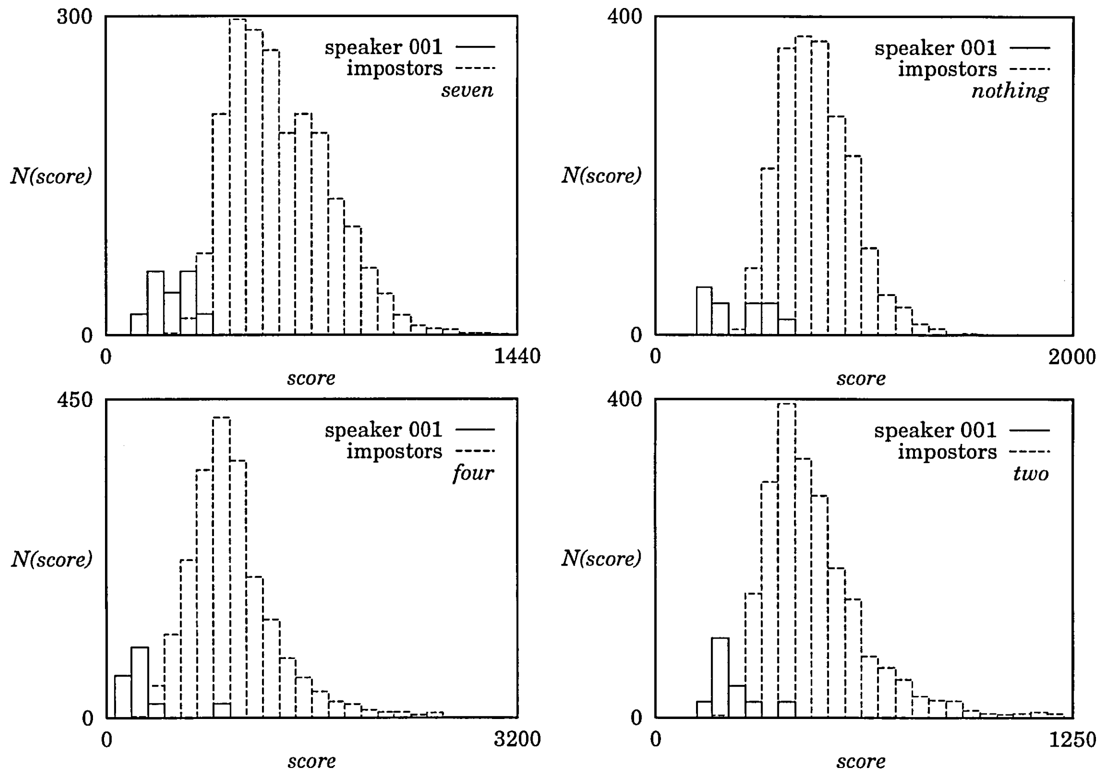

Figure 9.2: The score distributions for each of speaker 001's best 4 digits. A separate weighting vector is used with each digit, causing the absolute range of scores to differ from digit to digit.

9.3 Results

The d' was calculated for each speaker after each digit was added to the series. The distribution of d' ratings across the population at each stage of the series of digits is shown in Figure 9.3 by a vertical arrangement of diamonds. The dark areas are caused by many diamonds overlapping. The means of the distributions for each length of 'utterance' have been connected by lines, and there is a gradual increase as more digits are used.

As in Chapter 6, adding extra digits produces diminishing improvements in speaker separation, approaching a maximum mean d' of approximately 6.5. However, the statistical significance of these improvements also diminishes: performance with 4 digits is certainly better than a single digit but not definitely better than 3 digits.

Inspection of the d' sequences for individual speakers (Figure 9.4 shows six examples) shows a confusing picture: only 73 of the 211 (approximately one in

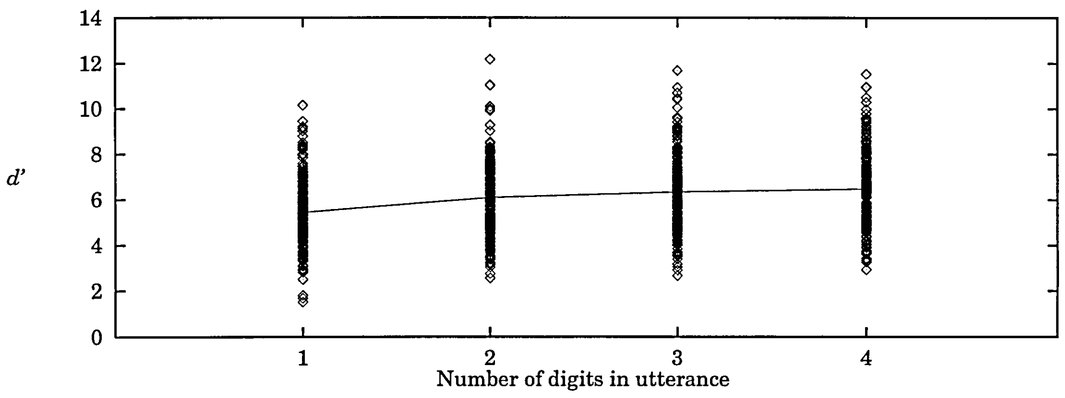

Figure 9.3: The distribution of d' ratings across the population is shown as a vertical pattern at each stage of a speaker-specific sequence of words. Speaker- and word-specific weights were used: lines connecting the population means suggest that using more words improves the verifier performance but with diminishing returns.

three) rises montonically. For some speakers, in fact most, the d' using only a single digit is no worse than using four. This effect was apparent in the results of Section 6.3.2 but was attributed to the speaker-dependent variety of usefulness of the digits. It was hoped that choosing the best four digits would remove it but there are several factors at work here.

- For any given speaker, there is no guarantee that the weights found for each digit will produce the same degree of improvement. This is partly due to the optimisation process being based on random selection but even if every single vector was tried, parity of performance across the digits could not be guaranteed. So, for some speakers, the weights found for the second, third or fourth digits in their sequence cannot boost the performance of those digits to a level comparable to that of their best digit.

- A bad score at any point in the sequence 'spoils' all results later, but not earlier, in the sequence. This is because the new scores at each stage are added to the aggregate scores of the previous stages. Since each speaker's digit sequence was chosen with the highest ranking digit first, the usefulness of each digit should decrease as the sequence lengthens. Thus, it becomes more unlikely that new scores could compensate for earlier poor scores.

- As noted in Section 8.2.1, occasionally the random routine is unable to find a better weighting vector than $(1,1,1,1,1,1,1,1,1,1,1,1)$, even with 50 attempts. It is therefore to be expected that the ordering of the best

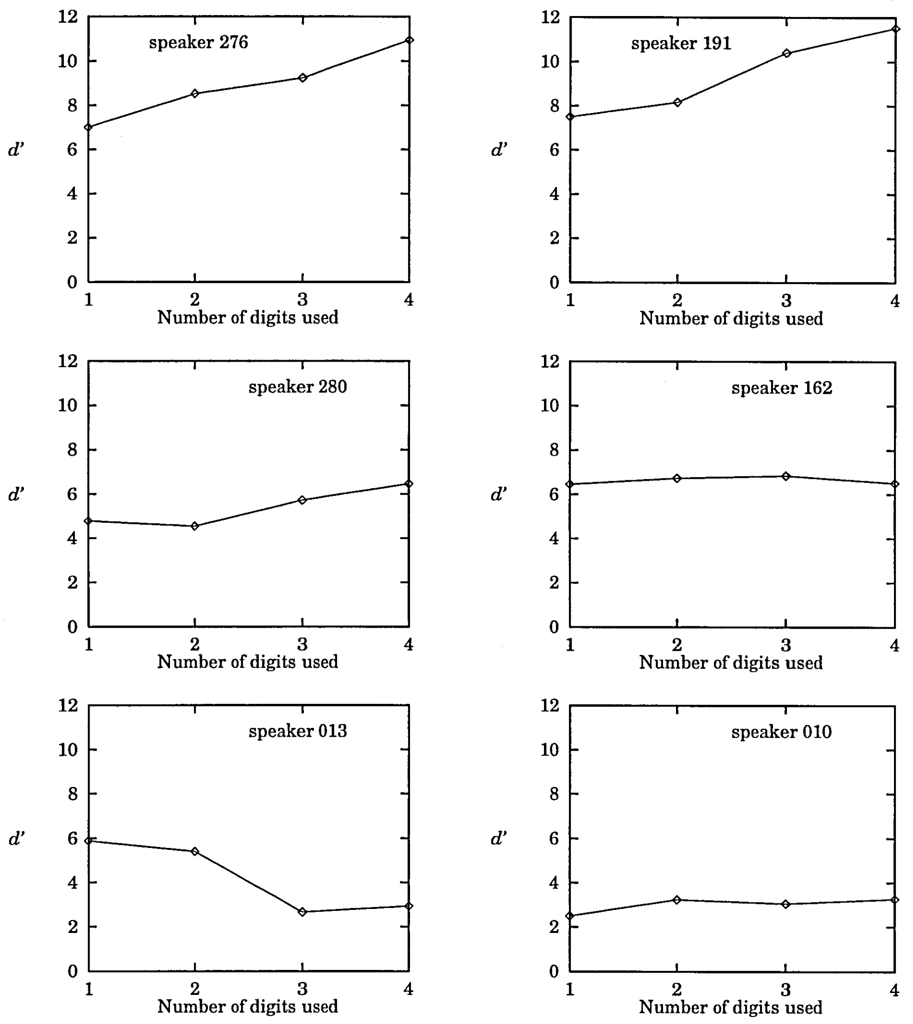

Figure 9.4: Six examples of d' sequences. In terms of final d' (for a 4-digit 'utterance'), speakers 191 and 276 have the highest, speakers 013 and 010 have the lowest, and speakers 280 and 162 are intermediate. Only the top two speakers display a monotonically increasing d'.

digits for a speaker would be altered by the optimisations, and the spoiling effect (described above) would act against monotonicity.

- The scores are added together consistently, so that if the digit sequence is *one six oh nine*, then the score for the first *one* utterance is added to the score for the first *six* utterance, and so on. However, there is no relationship between these two utterances: although they are the first of each digit, their contexts (preceding and following words) in the original recording are probably completely different. Consequently, for a given speaker, the first *one* score may be lower than other *one* scores but when added to the first *six* score (which may be higher than average) the sum is neither high nor low. When two groups of numbers are added together, the ordering of each group cannot affect the mean of the sums but can affect the standard deviation, as shown with mock data in Table 9.3. Since these are the two factors used in calculating the d' rating (see Chapter 5), the ordering of the utterances for each digit – currently arbitrary – could affect the separation statistics of the sums.

set	μ	σ
$A = (1, 8)$	4.5	3.5
$B = (9, 6)$	7.5	1.5
$C = (10, 14)$	12	2
$C' = (7, 17)$	12	5

Table 9.3: Mock data-sets $C = (A_1 + B_1, A_2 + B_2)$ and $C' = (A_1 + B_2, A_2 + B_1)$ have the same mean μ, but different standard deviations, σ.

The basis of this experiment is that a continuous text can be replaced effectively by a sequence of short words. The average d' rating for 4 digits with weighting vectors is 6.5, which is similar to the results using weights with *Sir Winston Churchill*, summarised in Table 8.4. However, as mentioned before, the d' ratings in this experiment are artificially high, so the performance here is probably not as good.

However, another comparison is possible: the previous results suggested that some speakers may be more identifiable than others. This was a possible explanation for the wide variation of performance across the population. If this is true regardless of the text used, then the speaker rankings for this experiment should be similar to previous rankings. Figure 9.5 compares the two rankings by plotting one along each axis. (The ranking for this experiment is based on the final d' rating for a 4-digit sequence.) If they were the same, or even similar, a line at 45° to both axes would be visible. This is not the case, indeed, quite the opposite: there is no correlation at all between the two rankings. Perhaps this is due to a text-dependent component of the scores for individual speakers: for some, certain sounds work well, because they highlight distinctive features of those speakers.

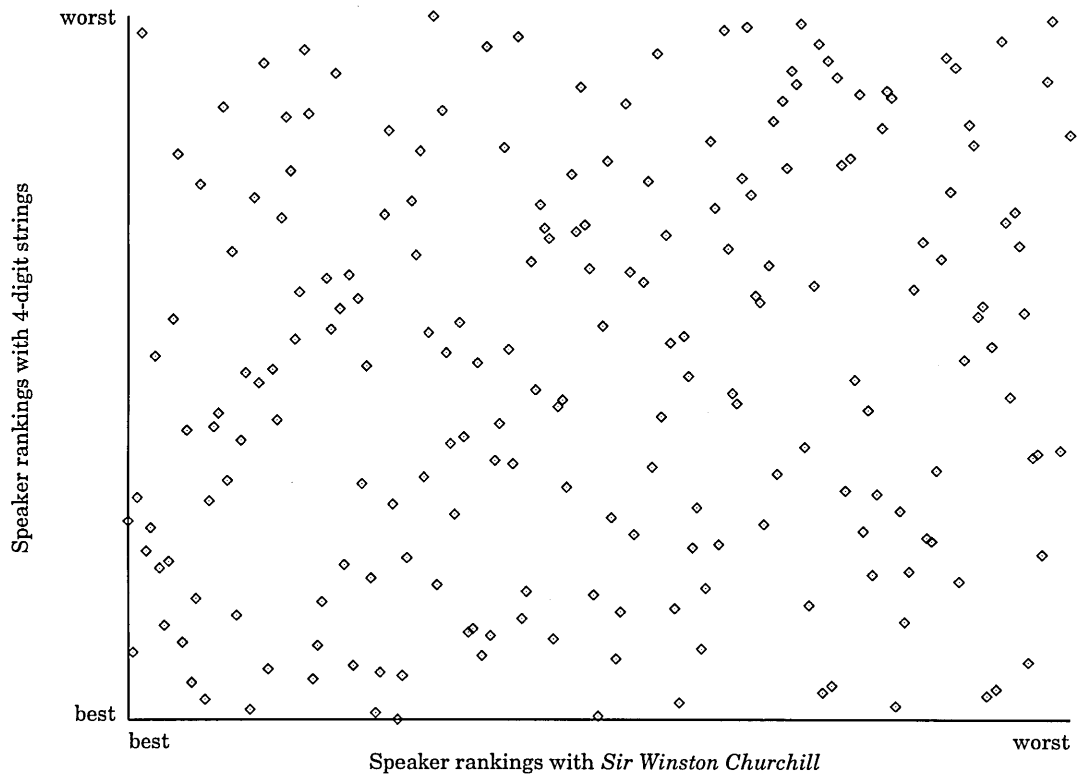

Figure 9.5: Comparing speaker rankings for different texts.

9.4 Conclusion

The average d' ratings of this experiment cannot be compared readily with previous experiments but the upward trend and shape of the curve confirm the earlier findings that individual scores may be combined for improved performance, up to a point. This experiment suggests that, on average, the useful number of words is 3 or 4, when weighting vectors are used. However, the non-monotonicity of the ratings in many cases suggests that a decision made before all 4 words have been spoken may give better performance. In practice, this would mean pre-specifying some threshold of separation (in terms of F or d', perhaps) which, if achieved, would 'prematurely' verify the identity.

Since the major effect of the weighting vectors seems to be finding good starting points for the DTW warp path, application to the digits is suitable: short utterances have a high risk of mistaken start- and end-points. Also, the consequences are probably worse than for longer words because a greater proportion of the word is affected.

In bringing together the insights gained from the previous chapters, this experiment has demonstrated an improved verifier configuration which could be applied to customer-operated banking, but further investigation using more self-test data is necessary.

Chapter 10

Conclusion

The aim of this project was to investigate speaker verification with a view to incorporating the technology into an unattended banking service, such as an automated teller machine. The purpose was not necessarily to develop a viable system but to discover the mechanism by which voices can be matched, and to instate reliable methods of assessment for any system. Some progress has been made toward the former and the latter has been achieved by the collection of two speech databases and the definition of two methods of assessment which convey more useful information than the often-quoted equal-error rate.

10.1 Experimental results

Following a survey of published reports and commercially-available products, a basic text-dependent, template-matching (dynamic time-warping) verifier has been implemented in software. Subsequent to the implementation, a trend appeared in the literature towards using hidden Markov models (Rosenberg *et al* 1991) or artificial neural networks (Oglesby and Mason 1991) but recently the older approaches, such as vector quantisation (Fakotakis 1992) and DTW (Yu, Mason & Oglesby 1995) have been used successfully. Besides, the verifier chosen allowed several different aspects of the configuration to be modified, and the effects noted. A series of experiments have been conducted to try to determine how such a system could best be applied to an ATM. The main findings are as follows.

- Although differences between speakers may be observed and measured at the frame level, the reliability of the discrimination process increases with the addition of more data, up to a point. This does not necessarily mean that a long utterance must be used, but the discrimination power of short utterances may be combined to good effect. Thus, as Chapter 6 concluded, several individual digits may be used as the verification text. Such a scheme is appropriate for the ATM context and has also been adopted in studies of other verification systems.

- The method of building a representative template from a few training utterances can affect the verification performance. However, this may have as much to do with the emphasis given to individual training utterances rather than the method itself. Combining the training utterances into a single composite template is the most attractive scheme in practice because storage capacity in any system is likely to be limited, although less so in a smart-card implementation. Also, the speed of the transaction is important and although keeping the tokens as separate entities allows a more powerful decision-making procedure, it probably wouldn't justify the extra complexity – and time.

- In calculating the difference between feature vectors, the use of weights to emphasise certain dimensions can improve the reliability of the verifier. A random selection process guided by the separation of genuine and impostor characteristics was used to find individual weighting vectors for each speaker and the improvement was almost universal. The reason is that the weighting vectors are optimised to influence the warp paths for the training group so that the self-test scores are tightly clustered. This seems to allow more freedom for the DTW process to find the best match for the initial frames and thereby correct for inconsistently determined start- and end-points. Endpoint detection which worked for a variety of noise conditions and speech strengths was found to be a very difficult problem and the 'safety net' provided by the weights increases reliability.

10.2 Envisaged scenarios

Bringing together all of these findings leads to the following recommended scenarios for applying speaker verification to ATMs.

10.2.1 Enrollment

An enrolling customer attends an initial recording session at the local branch of the bank. A specific room or booth would probably be required to minimise background noise. A continuous recording of randomly-ordered digits would be taken, prompted automatically, such that each digit is spoken 20 times. A computer within the branch would construct a reference template for each of the digits, possibly using some form of jackknifing to detect and discard atypical utterances.

The template for each digit would then be transmitted to a central computer holding similar material for perhaps 200 other speakers. The central computer would determine the closest impostors to the newly-enrolled customer (probably the same impostors could be used for each digit) and with this information, weighting vectors for each digit would be selected using the random routine. The ranked listings would be recalculated using the weights and the threshold would be determined as the average score of the nth impostor, according to the level of security required by the bank.

The collection of templates, thresholds and weighting vectors would then be encrypted and sent back over the network to the bank branch and written onto the customer's own personal smart card which would probably be used for other services also. The whole process would take about 10 minutes.

10.2.2 Verification

On presenting the smart card to the ATM, the user would be visually prompted to speak a digit, selected at random by the machine. The utterance would be recorded and as it is matched to the corresponding template (extracted from the card), another digit would be prompted. The process would continue until 4 digits have been compared. The scores of the individual matches would be summed and compared with the sum of the individual thresholds (stored on the card). If the score exceeds the threshold, the user may be prompted to speaker further digits, depending on the bank's view of an acceptable true-customer rejection rate.

10.3 Future work

This project has raised questions as well as answered them:

- Why do the ranges of scores vary from speaker to speaker? Section 5.1.1 illustrated that for any given speaker, there is no standard range for either the genuine scores or the impostor scores, but that the absolute ranges did not affect verifier performance. Genuine and impostor tests have one thing in common: the reference template. It would be interesting to alter the construction of the template for each speaker to examine the affect on the score ranges. Changing the utterances selected to make the template, or changing the length of the template are two possibilities.

- What role could jackknifing play in similar experiments? Figure 6.8 showed that the jackknife technique applied here produced a large distribution of self-test scores which was itself made from many smaller, individual distributions, which are not always similar. However, this may be due to the scheme using, effectively, single-token templates; an alternative scheme may give different results. For instance, it would be possible to take every unique combination of 5 tokens, make a template and record the self-test scores for each of the remaining 5 tokens.

  Jackknifing might also be used as a tool for selecting tokens to contribute to a template. Tokens with outlying sub-distributions may be discarded to produce a homogeneous template; or, they may be included to produce a comprehensive template covering several aspects of a person's voice.

- How would the results change if Furui's template method was optimised? Section 7.1.1 described how templates made this way are defined, to a large extent, by the initial token used in the construction. Using the small database, it would be possible to designate, in turn, each of the 20 tokens as the initial template, and discover a 'best' template, perhaps defined as the one giving the lowest average self-test score. Using these templates, the Furui method might compare more favourably with the others. The findings could be checked by extending this approach to the *Sir Winston Churchill* experiments.

- Why do the false acceptance and false rejection rates diverge rapidly on either side of the equal error-rate point? This is illustrated in Figure 7.6, for example. Such figures could be replotted using linear numerical scales to find out if this phenomenon is due entirely to the method of generating the thresholds. Sensitivity to poorly-estimated thresholds would be an important consideration in any practical system.

- How much do the speaker rankings vary from one template construction method to the next? Figure 7.5 shows that for some people, the verifier performance can be influenced strongly by the template method. However, Figure 7.10 shows that many of the verification errors are due to a small group of 'likely' impostors for a given individual genuine speaker. These two findings may be at odds with each other: are speaker rankings a function of the template method, or are some people are just more difficult to identify?

- How would performance vary with time? Collection of recordings from a set of subjects at regular intervals over several years could be used to test methods of keeping the reference templates up-to-date, and the rate of updating necessary to maintain performance could be checked.

- What effect do the weighting vectors have on 'unseen' utterances from the genuine speaker? Are they too specific to the training set? Is there a method for updating the weights in the same way as the reference template? Again, more recordings are needed.

- What is the best verification scenario in terms of the interface to the ATM? Using a telephone handset would allow prompting by voice, but would this be acceptable to customers? How many digits should be prompted? A sequential decision strategy might mean shorter verification times for many customers. Also, how many re-tries should be allowed after rejection? These issues would need to be examined in conjunction with public field trials.

More work on endpoint detection is also needed if a template-matching process is to be used, although this itself may indicate the unsuitability for this application. Further speaker verification experiments have already begun in this department using neural networks and testing with speakers who are available for re-recording (Finan, Sapeluk and Damper 1996). Whether the questions raised in this report can be answered in time for the inevitable widespread introduction of smart cards remains to be seen.

Appendix A

Resources

For this project, two speech databases were collected and many analysis and display programs were written. This appendix describes their whereabouts and operation, to aid reproduction of the major experiments.

A.1 Small database

In `C:\BERLIN\` on the machine labelled MC1213, there are sub-directories `A`, `B`, etc. These contain TNC files (20 for each speaker) named `xxyz.TNC` where `xx` are the initials of the speaker, `y` is the last letter in the utterance and `z` is the utterance number.

A.2 Large database

On the machine labelled MC1252, there are two installed hard drives, one designated `C` and another partitioned as `D` and `E`. Many of the analysis programs written for this project also make use of drive `F`, a ramdisk for storage of intermediate results.

In the root directory on drive `D`, there are sub-directories labelled `Cxxx`, where `xxx` is the reference number of a person. The numbers range from `001` to `300`. Within each of these directories, there are IDF files corresponding to each of the 140 words recorded for that speaker. Section A.4.1 describes the IDF format.

A.2.1 Full text recordings

In D:\E_BACKUP\CUT\, there is an archived directory structure for each speaker, containing IDF and TNC files for all 20 utterances (*Sir Winston Churchill*). Also included are the impostor score and ranking files compared against each template (4 methods of construction) and statistics files for randomly-generated weighting vectors.

To perform the experiments, the TNC files for all speakers were copied to D:\E_BACKUP\TNCS\ as 7 archive files. (It was necessary to use several files because DOS allows a maximum of 1024 files in a directory.) Batch files to load them to the ramdisk are in the same directory, and in the directory above, the file BATS.ZIP contains more batch files to run various experiments.

A.2.2 Digits

All of the tests using digits were conducted after making the digits TNC database. There are 4 archive files in E:\DIGITS\ which accomplish these tasks.

BIGTEST.ZIP contains the following files.

- MASTER.BAT calls MULTIGCC.BAT once and MULTIJK.BAT 12 times.
- MULTIGCC.BAT calls GCC.BAT for each speaker. This gets IDF files from D:\Cxxx\ and cuts the leading silence off using EPD3.EXE. IDF2TNC.EXE is then called to convert the shortened speech files to cepstral coefficients.
- MULTIJK.BAT calls JK.BAT for each speaker, which produces from their 10 utterances a jack-knifed score distribution, which is stored as RESULT.DAT in E:\DIGITS\ONE\001\, for example. A summary of RESULT.DAT is stored in DATSTAT.OUT, in the same directory.

In MAKEMEAN.ZIP, MULTIGM.BAT calls GM.BAT for every speaker. GM.BAT calls GMP.BAT for every digit, for the given speaker. GMP.BAT calls GET_MEAN.EXE which reads RESULT.DAT for the specified digit and speaker, and finds out which of the 1260 jackknife templates produced scores closest to the distribution mean. The template is recreated and stored in E:\DIGITS\ONE\001\, for example.

DIGITZPS.ZIP contains batch files to create 5 archives called 1.ZIP,. . .,5.ZIP in each of the digit sub-directories in E:\DIGITS\. TESTDIGS.ZIP contains the following batch files, which can be used to perform unweighted tests for all speakers saying all the digits (Chapter 6). The results of these tests are stored in E:\DIGITS\STA\ as ALL.DP (a list of d' ratings for each speaker and each digit) and ONE.ZIP, etc. which contains distribution statistics for each speaker and digit.

- STARDIGS.BAT makes 6 directories on F, labelled REFS, TNC1–TNC5. Copies *.ZIP from E:\DIGITS\TEMPLATE\ to F:\REFS.
- ALLLOOP.BAT calls MULTILP one, MULTILP two, etc.
- MULTILP.BAT unzips all the test TNCs for the given digit, e.g. 1.ZIP in E:\DIGITS\ONE\ to F:\TNC1, before calling LOOP.BAT for each speaker. The resultant statistics files are archived as ONE.ZIP etc., in E:\DIGITS\STA\.
- LOOP.BAT extracts ONE.TNC from F:\REFS\001.ZIP and renames it to 001R.TNC. Copies DATSTAT.OUT from E:\DIGITS\ONE\001\ to 001SELF.DAT in the root directory of F, then executes FTEST 001. The results are archived as 001ZDAT.ZIP in D:\DIGITS\IMPDATS\ONE\ and temporary statistics are saved as F:\001.STA etc.

In C:\TCPP\MALC\CONCAT\, MISCDIG.EXE is used to perform tests using concatenated digit scores. There are too many combinations of digit strings, speakers and bands of the jackknifed distributions to test exhaustively, so these are selected randomly. The program takes one argument, an integer which specifies how many thousands of iterations to perform. The program should be run from the ramdisk, and uses the RESULT.DAT files from E:\DIGITS\ONE\001\, for example. Also, impostor score archives, such as 001ZDAT.ZIP from D:\DIGITS\IMPDATS\ONE\, are used. Results are appended to the text file C:\MISCDIG.RES. Copies of this file can be found in E:\DIGITS\RES\.

Weighted digits

C9.EXE in C:\TCPP\MALC\C9\ reads LISTINGS.DPS in the same directory, to produce BESTDIGS.TXT, which contains the list of 12 digits, ranked by decreasing performance, for each speaker. This file may then be read by RANDOM2.EXE (in C:\TCPP\MALC\RANDOM2\) which extracts the first (best) 4 digits per speaker, and generates 50 random weighting vectors for each. Each vector for each digit is tested using the 5 impostors closest to the given speaker, taken from

BERNASCO/IMPOSTOR.AVG which may be found in each speaker's archive file (such as 001.ZIP) in D:\E_BACKUP\CUT\. Digit reference templates are archived in C:\TCPP\MALC\DIGREFS\DIGREFS.ZIP and test utterances are taken from E:\DIGITS\ONE\001\, for instance.

The results are stored as BESTDIGS.STA, which is a text file containing 4 lines for each speaker. Each line has the speaker's code, a digit name, the best weighting vector (expressed a string of 12 hexadecimal digits) and F and d' ratings (assessed using the subset of impostors). WDIGITS.EXE in C:\TCPP\MALC\WDIGITS\ is used to assess the performance with *all* the impostors, using the concatenated scores scheme. The results are appended to C:\WDIGITS.RES.

A.3 Utilities

The following executable files are stored in D:\EXES\.

- IDF2TNC converts IDF to TNC based on linear predictive coding.
- CATIDF concatenates 2 IDF files.
- TIMEDIT allows graphical display and editing of large TIM (Hypersignal) files. Sections are written as IDF files.
- FLIPDAT reverses a TIM or IDF file.
- DAT2TXT converts either TIM or IDF to text files.
- TNC2TXT converts TNC to a text file.
- REFCON2 prompts for IDF files and makes a Furui reference template.
- REFCON3 makes a Bernasconi template from 5 IDF files.
- REFCON4 makes a Bernasconi template from 5 TNC files.
- WARP6 compares 2 TNC files using dynamic time-warping and displays the result graphically. Requires EGAVGA.BGI in the same directory.
- COMPARE2 compares 2 TNC files but appends the result to a file called RESULT.DAT.

A.4 In-house file formats

Two file formats commonly used in this project have been established in this department. The IDF format is due to Turnbull (1991); the TNC format is original.

A.4.1 IDF format

Each integer data file consists of a 16-byte header followed by n 2-byte signed integers, where n is the number of samples in the recording. The header format is shown in Table A.1. The first item is the number of 16-byte paragraphs that the file occupies, including the header itself. The next item represents the sampling frequency, f_s, encoded as floor($(1 - 5 \times 10^6)/f_s$), where floor(x) returns the largest integer $\leq x$.

type	bytes	description
unsigned int	1, 2	paragraphs
int	3, 4	frequency
int	5, 6	scale
–	7–16	not used

Table A.1: Header format of integer data file (IDF)

A.4.2 TNC format

Time-normalised cepstral coefficients are stored in TNC files, consisting of a 10-byte header followed by data. The first two bytes represent an integer n, which is the number of frames into which the original utterance was split for analysis. Next, two (4-byte) floating point numbers represent possible lower and upper thresholds which may be embedded with a reference template, thus allowing 3 verification decisions: accept, reject or try again. (Currently, these thresholds are not used, and all TNC files have both thresholds set to zero.) Immediately following the header are the 12 coefficients (48 bytes) for each of the n frames.

Appendix B

DTW equations

Let $\mathbf{r}_i$ and $\mathbf{t}_j$ be p-dimensional vectors representing the ith reference frame and jth test frame, respectively. (In this project, $p = 12$.) The distance δ_{ij} between them is defined as:

$$\delta_{ij} = \sum_{k=1}^{p} (\mathbf{r}_{i_k} - \mathbf{t}_{j_k})^2. \tag{B.1}$$

This is equivalent to the square of the Euclidean distance; the square root operation is omitted to reduce computation.

Dynamic time-warping is usually described in terms of an M by N grid (M reference frames, N test frames) of points (i, j), each representing a distance $d_{i,j}$, defined as:

$$d_{i,j} = \begin{cases} \delta_{ij} + \min(d_{i-1,j},\ d_{i-1,j-1},\ d_{i-1,j-2}) & H_{i-1} = 0 \\ \delta_{ij} + \min(d_{i-1,j-1},\ d_{i-1,j-2}) & H_{i-1} = 1 \end{cases} \tag{B.2}$$

where H_i is a check to ensure that a candidate path through the grid cannot make two consecutive horizontal transitions.

$$H_i = \begin{cases} 1 & \min(d_{i-1,j},\ d_{i-1,j-1},\ d_{i-1,j-2}) = d_{i-1,j} \\ 0 & \text{otherwise} \end{cases} \tag{B.3}$$

The first step is to find the best match to r_1 from the first $2\epsilon+1$ frames of the test token. This search range is defined as $c \pm \epsilon$, where c is the centre point. In this project, $\epsilon = 4$ and c_1, the centre of the first search range, is 5. At each iteration, the centre of the next search range, c_{i+1} is chosen such that:

$$d_{i,c_{i+1}} = \min_{(c_i-\epsilon)\leq j\leq(c_i+\epsilon)} d_{i,j} \tag{B.4}$$

where $d_{i,j}$ is calculated according to equation B.2. Implemented as a computer program, the initial conditions:

$$d_{i,j} = \begin{cases} \text{MAXFLOAT} & i \neq 0 \\ 0 & i = 0 \end{cases} \tag{B.5}$$

and $H_0 = 0$ ensure that the routine can start to search the grid. (MAXFLOAT is the compiler's maximum floating-point number.) Each column is searched (incrementing i) until either $i = M$ or $c = N$. The overall distance, D, between the reference and test tokens is the distance accumulated at the final point in the path, normalised by dividing by the number of contributing local distances:

$$D = d_{i_{\text{final}},c}/i_{\text{final}}. \tag{B.6}$$

References

Acero, A., Crespo, C., de la Torre, C. and Torrecilla, J.C. (1993) "Robust HMM-based endpoint detector", *Proc. ESCA Eurospeech '93*, Berlin, Germany, **3**, 1551–1554.

Anderson, T. and Patterson, R. (1994) "Speaker recognition with the auditory image model and self-organizing feature maps: a comparison with traditional techniques", *Proc. ESCA Workshop on Automatic Speaker Recognition, Identification and Verification*, Martigny, Switzerland, April 5–7, 153–156.

Angus, J.A.S. and Whitaker, M.T. (1987) "An algorithm for increasing speed in dynamic time warping", *Proc. European Conference on Speech Technology*, Edinburgh, Scotland, September 2–4, **2**, 284–287.

Atal, B.S. and Hanauer, S. (1971) "Speech analysis and synthesis by linear prediction of the speech wave", *Journal of the Acoustical Society of America*, **50**, 637–655.

Atal, B.S. (1974) "Effectiveness of linear prediction characteristics of the speech wave for automatic speaker identification and verification", *Journal of the Acoustical Society of America*, **55**, 1304–1312.

Bennani, Y. and Gallinari, P. (1991) "On the use of TDNN-extracted features information in talker identification", *Proc. IEEE ICASSP '91*, Toronto, Canada, **1**, 385–388.

Bernasconi, C. (1990) "On instantaneous and transitional information for text-dependent speaker verification", *Speech Communication*, **9**, 129–139.

Bielby, G., Lennig, M. and Mermelstein, P. (1987) "Speaker verification with sequential decision on a speaker specific vocabulary", (abstract only) *Proc. European Conference on Speech Technology*, Edinburgh, Scotland, September 2–4, **2**, 459.

BIMBOT, F., CHOLLET, G. AND PAOLONI, A. (1994) "Assessment methodology for speaker identification and verification systems. An overview of SAM-A Esprit project 6819 – task 2500", *Proc. ESCA Workshop on Automatic Speaker Recognition, Identification and Verification*, Martigny, Switzerland, April 5–7, 75–82.

BOOTH, I., BARLOW, M. AND WATSON, B. (1993), "Enhancements to DTW and VQ decision algorithms for speaker recognition", *Speech Communication*, no. 13, 427–433.

BUCK, J.T., BURTON, D.K. AND SHORE, J.E. (1985), "Text-dependent speaker recognition using vector quantisation", *Proc. IEEE ICASSP '85*, Tampa, USA, 391–394.

CAREY, M., PARRIS, E. AND BRIDLE, J. (1991) "A speaker verification system using alpha-nets" *Proc. IEEE ICASSP '91*, Toronto, Canada, **1**, 397–400.

COLOMBI, J., ANDERSON, T., ROGERS, S., RUCK, D. AND WARHOLA, G. (1993) "Auditory model representation for speaker recognition", *Proc. ICASSP '93*, Minneapolis, MN, USA, 700–703.

DE JONG, K.A. AND SPEARS, W.M. (1990) "An analysis of the interacting roles of population size and crossover in genetic algorithms", *Proc. 1st Workshop on Parallel Problem-solving from Nature*, October 1–3, 38–47.

DERMATAS, E.S., FAKOTAKIS, N.D. AND KOKKINAKIS, G.K. (1991) "Fast endpoint detection algorithm for isolated word recognition in office environment", *Proc. IEEE ICASSP '91*, Toronto, Canada, **1**, 733–736.

DE VETH, J. AND BOURLARD, H. (1994) "Comparison of hidden Markov model techniques for automatic speaker verification", *Proc. ESCA Workshop on Automatic Speaker Recognition, Identification and Verification*, Martigny, Switzerland, April 5–7, 11–14.

DODDINGTON, G. (1985) "Speaker recognition—identifying people by their voices", *Proc. IEEE*, **73**, 1651–1664.

FAKOTAKIS, N., TSOPANOGLOU, A. AND KOKKINAKIS, G. (1992) "Speaker verification over telephone lines based on digit strings", *Proc. Eusipco '92*, **1**, 399–402.

FINAN, R.A., SAPELUK, A.T. AND DAMPER, R.I. (1996) "Comparison of multilayer and radial basis function neural networks for text-dependent speaker recognition", *Proc. IEEE Int. Conf. on Neural Networks*, Washington, DC, USA, June 3–6, 1992–1997.

FORSYTH, M.E., BAGSHAW, P.C. AND JACK, M.A. (1994) "Incorporating discriminating observation probabilities (DOP) into semi-continous HMM for speaker verification", *Proc. ESCA Workshop on Automatic Speaker Recognition, Identification and Verification*, Martigny, Switzerland, April 5–7, 19–22.

FURUI, S. (1974) "An analysis of long-term variation of feature parameters of speech and its application to talker recognition", *Electron. Commun.* , **57-A**, 34–42.

FURUI, S. (1981) "Cepstral analysis techniques for automatic speaker verification", *IEEE Trans. Acoustics, Speech and Signal Processing*, **ASSP-29**, no. 2, 254–272.

FURUI, S. (1994) "An overview of speaker recognition technology", *Proc. ESCA Workshop on Automatic Speaker Recognition, Identification and Verification*, Martigny, Switzerland, April 5–7, 1–9.

GAGANELIS, D.A. AND FRANGOULIS, E.D. (1991) "A novel approach to speaker verification", *Proc. IEEE ICASSP '91*, Toronto, Canada, **1**, 373–376.

GAN, C.K. AND DONALDSON, R.W. (1988) "Adaptive silence deletion for speech storage and voice mail applications", *IEEE Trans. Acoustics, Speech and Signal Processing*, **36**, no. 6, 924–927.

GODFREY, J., GRAFF, D. AND MARTIN, A. (1994) "Public databases for speaker recognition and verification", *Proc. ESCA Workshop on Automatic*

Speaker Recognition, Identification and Verification, Martigny, Switzerland, April 5–7, 39–42.

GOLDBERG, D.E. (1989) *Genetic Algorithms in Search, Optimization and Machine Learning*, Addison-Wesley, Reading, MA.

GOLDSMITH, M.J. (1989) *Speech Databases for UK Speech Technology Research: A Survey of Resources and Future Needs*, Report RSA(EXT)010, National Physical Laboratory, Teddington, England.

GRANT, P.M. (1991) "Speech recognition techniques", *Electronics & Communication Engineering Journal*, February, 37–48.

GREEN, D.M. AND SWETS, J.A. (1966) *Signal Detection Theory and Psychophysics*, John Wiley, New York, NY.

HAIGH, J.A. AND MASON, J.S. (1993) "A voice activity detector based on cepstral analysis", *Proc. ESCA Eurospeech '93*, Berlin, Germany, **2**, 1103–1106.

HANNAH, M.I. (1992) *Automatic speaker verification in ATMs*, MPhil-to-PhD Transfer document, University of Abertay Dundee.

HANNAH, M.I., SAPELUK, A.T., DAMPER, R.I. AND ROGER, I.M. (1993a) "The effect of utterance length and content on speaker-verifier performance", *Proc. ESCA Eurospeech '93*, Berlin, Germany, **3**, 2299–2302.

HANNAH, M.I., SAPELUK, A.T., DAMPER, R.I. AND ROGER, I.M. (1993b) "Using genetic algorithms to improve speaker-verifier performance", *Proc. IEE/IEEE Workshop on Natural Algorithms in Signal Processing*, 24/1–24/9, Chelmsford, England.

HIGGINS, A.L. AND WOHLFORD, R.E. (1986) "A new method of text-independent speaker recognition", *Proc. IEEE ICASSP '86*, Tokyo, Japan, 869–872.

HOLLAND, J.H. (1975) *Adaptation in Natural and Artificial Systems*, University of Michigan Press, Ann Arbor, MI.

HOLLIEN, H. AND MAJEWSKI, W. (1977) "Speaker identification by long-term spectra under normal and distorted conditions", *J. Acoustical Society of America*, **62**, 975–980.

HOROWITZ, P. AND HILL, W.(1989) *The Art of Electronics*, 2nd ed., Cambridge University Press: Cambridge, England.

HUGHES, A., LONGAIR, I., ASHLEY, R AND KIRBY, K. (1995) "Increased accuracy in large scale sewer flow mean velocity measurements using an array of ultrasonic velocity transducers", *Proc. IEE Colloquium on Intelligent Measuring Systems for Control Applications*, London, England, 3/1–3/3.

IRVINE, D.A. AND OWENS, F.J. (1993) "A comparison of speaker recognition techniques for telephone speech", *Proc. ESCA Eurospeech '93*, Berlin, Germany, **3**, 2275–2278.

ITAKURA, F. (1975) "Minimum prediction residual principle applied to speech recognition", *IEEE Trans. Acoustics, Speech and Signal Processing*, **ASSP-23**, 67–72.

JACOBS, T. AND SETLUR, A. (1994) "A field study of performance improvements in HMM-based speaker verification", *Proc. 2nd IEEE Workshop on Interactive Voice Technology for TeleCommunications Applications*, Kyoto, Japan, 121–124.

LAMEL, L.F., RABINER, L.R., ROSENBERG, A.E. AND WILPON, J.G. (1981) "An improved endpoint detector for isolated word recognition", *IEEE Trans. Acoustics, Speech and Signal Processing*, **ASSP-29**, 777–785.

LATHI, B.P. (1989) *Modern Digital and Analog Communication Systems*, 2nd ed., Holt, Rinehart & Winston: Philadelphia, PA.

LI, K. AND WRENCH, E. (1983) "An approach to text-independent speaker recognition with short utterances", *Proc. IEEE ICASSP '83*, Boston, USA, **2**, 555–558.

MAKHOUL, J. (1975) "Linear prediction: a tutorial review", *Proc. IEEE*, **63**, 561–580.

MAKHOUL, J. (1977) "Stable and efficient lattice methods for linear prediction", *IEEE Trans. Acoustics, Speech and Signal Processing,* **ASSP-25**, 423–428.

MAUUARY, L. AND MONNÉ, J. (1993) "Speech/non-speech detection for voice response systems", *Proc. ESCA Eurospeech '93*, Berlin, Germany, **2**, 1097–1100.

MILLER, R.G. (1974) "The jackknife – a review", *Biometrika*, **61**, 1–15.

MYERS, C.S., RABINER, L.R. AND ROSENBERG, A.E. (1981) "On the use of dynamic time warping for word spotting and connected word recognition", *Bell Systems Technical Journal*, **60**, 303–325.

NAIK, D., ASSALEH, K. AND MAMMONE, R. (1994) "Robust speaker identification using pole filtering", *Proc. ESCA Workshop on Automatic Speaker Recognition, Identification and Verification*, Martigny, Switzerland, April 5–7, 225–230.

NETSCH, L. AND DODDINGTON, G. (1992) "Speaker verification using temporal decorrelation post-processing", *Proc. IEEE ICASSP '92*, San Francisco, USA, II-181–184.

NEY, H. (1981) "An optimization algorithm for determining the endpoints of isolated utterances", *Proc. IEEE ICASSP '81*, Atlanta, GA, USA, 720–723.

OGLESBY, J. AND MASON, J. (1988) "Speaker identification using neural nets", *Proc. IOA Speech '88*, Edinburgh, Scotland, 1357–1363.

OGLESBY, J. AND MASON, J. (1989) "Speaker recognition with a neural classifier", *Proc. IEE First Int. Conf. on Artificial Neural Networks*, 306–309.

OGLESBY, J. AND MASON, J. (1990) "Optimisation of neural models for speaker identification", *Proc. IEEE ICASSP '90*, Albuquerque, NM, USA, 261–264.

OGLESBY, J. AND MASON, J. (1991) "Radial basis function networks for speaker recognition", *Proc. IEEE ICASSP '91*, Toronto, Canada, 393–396.

OGLESBY, J. (1994) "What's in a number?: Moving beyond the equal error rate" *Proc. ESCA Workshop on Automatic Speaker Recognition, Identification and Verification*, Martigny, Switzerland, April 5–7, 87–90.

OWENS, F.J. (1993) *Signal Processing of Speech*, Macmillan Press: Basingstoke, England.

O'SHAUGHNESSY, D. (1987) *Speech Communication: Human and Machine*, Addison-Wesley: Reading, MA.

PIPKIN, F.B. (1984) *Medical Statistics Made Easy*, Churchill Livingstone: Edinburgh, Scotland.

PRESS, W.H., TEUKOLSKY, S.A., VETTERLING, W.T. AND FLANNERY, B.P. (1988) *Numerical Recipes in C: The Art of Scientific Computing*, Cambridge University Press: Cambridge, England.

PRUZANSKY, S. (1963) "Pattern matching procedure for automatic talker recognition", *J. Acoustical Society of America*, **35**, 354–358.

RABINER, L. AND SAMBUR, M. (1975) "An algorithm for determining the endpoints of isolated utterances", *Bell Systems Techology Journal*, **54**, 297–315.

RABINER, L.R., ROSENBERG, A.E. AND LEVINSON, S.E. (1978) "Considerations in dynamic time warping algorithms for discrete word recognition", *IEEE Trans. Acoustics, Speech and Signal Processing*, **ASSP-26**, 575–582.

ROSE, R., FITZMAURICE, J., HOFSTETTER, E. AND REYNOLDS, D. (1991) "Robust speaker identification in noisy environments using noise adaptive speaker models", *Proc. ICASSP '91*, Toronto, Canada, **1**, 401–404.

ROSENBERG, A.E., LEE, C-H, AND GOKCEN, S. (1991) "Connected word talker verification using whole word hidden Markov models", *Proc. IEEE ICASSP '91*, Toronto, Canada, **1**, 381–384.

ROSENBERG, A.E. AND SOONG, F.K. (1991) "Evaluation of a vector quantization talker recognition system in text independent and text dependent modes", *Proc. IEEE ICASSP '86*, Tokyo, Japan, 873–876.

RUDASI, L. AND ZAHORIAN, S. (1991) "Text-independent talker identification with neural networks", *Proc. IEEE ICASSP '91*, Toronto, Canada, **1**, 389–392.

SAKOE, H. AND CHIBA, S. (1978) "Dynamic programming algorithm optimization for spoken word recognition", *IEEE Trans. ASSP*, **ASSP-26**, 43–49.

SOLZHENITSYN, A. (1969) *The First Circle*, William Collins: Glasgow, Scotland.

SOONG, F.K., ROSENBERG, A.E., RABINER, L.R., AND JUANG, B.H. (1985) "A vector quantization approach to speaker recognition", *Proc. IEEE ICASSP '85*, Tampa, FL, 387–390.

SOONG, F.K., ROSENBERG, A.E. (1985) "On the use of instantaneous and transitional spectral information in speaker recognition", *Proc. IEEE ICASSP '86*, Tokyo, Japan, 877–880.

SPILLMAN, R. (1993) "Genetic Algorithms: Nature's way to search for the best", *Dr. Dobb's Journal*, February, 26–30.

SU, L., LI, K. AND FU, K. (1974) "Identification of speakers by use of nasal coarticulation", *Journal of the Acoustical Society of America*, **56**, 1876–1882.

THOMPSON, J. AND MASON, J.S. (1994) "The pre-detection of error-prone class members at the enrollment stage of speaker recognition systems", *Proc. ESCA Workshop on Automatic Speaker Recognition, Identification and Verification*, Martigny, Switzerland, April 5–7, 127–130.

TOHKURA, Y. (1994) "A weighted cepstral distance measure for speech recognition", *Proc. IEEE ICASSP '86*, Tokyo, Japan, **1**, 761–764.

TURNBULL, J.M. (1991) *A New computer-based speech therapy tutor for vowel production*, Ph.D. Thesis, University of Abertay Dundee.

VELIUS, G. (1988) "Variants of cepstrum based speaker identity verification", *Proc. IEEE ICASSP '88*, New York, USA, **5**, 583–586.

XU, L., OGLESBY, J. AND MASON, J. (1989) "The optimization of perceptually-based features for speaker identification", *Proc. IEEE ICASSP '89*, Glasgow, Scotland, **1**, 520–523.

Yu, K., Mason, J. and Oglesby, J. (1995) "Speaker recognition using hidden Markov models, dynamic time warping and vector quantisation", IEE Proc.-Vis. Image Signal Process., **142**, no. 5, 313–318.

Zheng, Y-C. and Yuan, B-Z. (1988) "Text-dependent speaker identification using circular hidden Markov models", *Proc. IEEE ICASSP '88*, New York, USA, **5**, 580–582.

THE EFFECT OF UTTERANCE LENGTH AND CONTENT ON SPEAKER-VERIFIER PERFORMANCE

M.I. Hannah*, A.T. Sapeluk*, R.I. Damper† and I.M. Roger‡
*Department of Electronic and Electrical Engineering, Dundee Institute of Technology, Dundee, Scotland
†Department of Electronics and Computer Science, University of Southampton, Southampton, England
‡Technology Development Group, NCR, Dundee, Scotland

ABSTRACT

Utterance length is known to be an important factor in text-independent speaker verification. Here, we examine the effect of length on speaker separation for text-dependent verification. When both test data and templates are continuous-word strings, separation increases with length with apparent saturation at about 2 seconds. Simulating a possible practical scenario using continuous-word test data and templates formed by concatenating discrete words did not achieve useful separation for the longest tokens, apparently because of word-boundary effects. When test data and templates are discrete-word with matching scores accumulated over the maximum length, however, separation was good.

Keywords: Speaker verification, text-dependent verification, utterance length.

1 INTRODUCTION

Attempts to verify a person's identity using features extracted from samples of their speech implicitly assume that there is little or no overlap of statistical distributions of such features between the individual and the rest of the population [1]. Many studies of text-independent verification (e.g. [2]) have reported that utterance length is an important factor in determining the degree of separation of the two distributions. We explore here the effect that utterance length has on performance in the case of text-dependent verification. Following [1] and [3], our verifier employs dynamic time-warping matching of test and reference utterances represented as sequences of cepstral coefficients obtained from linear prediction analysis.

This paper is structured as follows. The details of the speech data used and of the verifier employed are described in Sections 2 and 3 respectively. Section 4 discusses the means of assessing verification performance while section 5 describes our 4 experiments and their results. Finally, Section 6 concludes.

2 SPEECH DATA

Speech data for experiments 1 and 2 (see below) were recorded from 10 young adults, 5 male and 5 female, in a quiet laboratory. An anti-alias filter with 4 kHz cut-off was used prior to sampling at a rate of 10 kHz and resolution of 16 bits. Each subject spoke the phrase ('text') *ABCDEFG* 20 times; each utterance took no more than 3 seconds. Using the same text for all subjects allowed us to designate any subject as a *genuine* speaker and the other nine as *impostors*, although all were speaking naturally. Utterances were then segmented (see Section 5) to produce tokens of varying length. Additional recordings were made by a subset of the original 10 speakers to provide the data for experiments 3 and 4.

Each token was analysed in 25 ms non-overlapping frames using the lattice method of 12th-order linear prediction analysis. Each frame is then described by a vector of 12 cepstral coefficients, obtained using the recursion described in [4]. Each cepstral vector is then normalised by subtracting from each vector the average vector for the whole utterance. The time sequence of normalised vectors forms the input to the verifier.

3 VERIFIER OPERATION

The verifier is an un-optimised version based on the principles detailed in [1] and [3]. It relies on matching a test utterance to reference patterns ('templates') using dynamic time-warping (DTW) to produce a score, where 0 indicates a perfect match and higher scores indicate progressive degrees of mis-match. Except in experiment 4, a reference pattern for a given speaker and utterance is actually based on 5 repetitions of that utterance by that speaker. Each repetition is DTW-matched to every other, and the utterance with the lowest accumulated score is adjudged to be the centroid of the group. The other 4 utterances are then time-aligned to the centroid and all 5 averaged to form the template. None of the utterances used in the construction of the template are ever used in testing.

4 ASSESSING PERFORMANCE

The performance metric traditionally used in speaker verification is the error rate. However, there are two possible types of error: false acceptance and false rejection. A general statement of the accuracy of a system must cover both cases because it is possible to bias the decision towards minimisation of one error at the expense of the other. Hence, the *equal-error rate* is often quoted, where the probability of each type of error is the same, but this implies the existence of some threshold value of speaker scores. That is, an identity claim from an

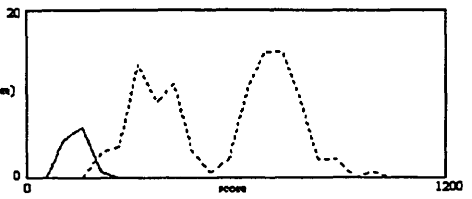

Figure 1: Typical distributions of *genuine* and *impostor* scores (solid and broken lines, respectively).

individual is accepted (or rejected) if the verification score is less than (or more than) the threshold.

Due to the inherent variability of individual voices, a single, global threshold cannot be guaranteed optimal for every speaker; instead, individual thresholds are needed. However, the thresholds should themselves depend on the distribution of scores produced by the matching procedure – such as those shown in Figure 1 – and so cannot be set in advance. Also, if total separation (non-overlap) of score distributions occurs, the *degree* of separation cannot be conveyed by simply quoting a zero error rate – yet two systems which avoid errors with a limited test set may well not produce equal error rates in their eventual application. For these reasons, we feel that setting of thresholds and estimation of error rates is ill-advised, except during the final stage of development of a verifier. Instead, we focus on the separation and/or the overlap of score distributions for the genuine speaker and the impostors. To help in this assessment, we employ two figures of merit: a version of the classical d' and our own 'overlap' function, F.

4.1 Using d'

The d' measure from classical signal detection theory [5] assumes the existence of two response distributions (signal and signal-plus-noise, corresponding here to genuine and impostor) which are Gaussian and equal variance. Then:

$$d' = \frac{|\mu_s - \mu_{sn}|}{\sigma}$$

where μ_s and μ_{sn} are the means of the signal and signal-plus-noise score distributions respectively, and σ^2 is the common variance. In our case, the assumptions are invalid but d' seems nonetheless to provide a reasonably consistent measure of separation. We have modified the original to compensate for the inequality of variances thus:

$$d' = \frac{\mu_{imp} - \mu_{gen}}{S}$$

where μ_{imp} and μ_{gen} are the means of the impostor and genuine distributions respectively, and S is the geometric mean of the standard deviations of the two distributions. Note that (unlike the situation in signal detection theory) d' can be negative in the (highly undesirable!) case that $\mu_{gen} > \mu_{imp}$.

4.2 Overlap Function (F)

We have developed an alternative measure of verifier performance based on the so-called 'overlap' function, F. This is an easily-computed, *ad hoc* figure which has a valid range of $[0, 1]$, although this is split into two equal sub-ranges to cover the distinct cases of overlap and non-overlap, or (total) separation. F is computed differently for each case (labelled $F_{overlap}$ and $F_{non-overlap}$) but increases linearly with verifier performance over the entire range. $F_{overlap}$ is inversely related to the probability of an impostor's utterance producing a score that is within the range of the genuine speaker's scores:

$$F_{overlap} = 0.5 - \frac{f}{2N_i} \qquad (1)$$

where f is the number of 'failures' in terms of an impostor score being within the range of genuine scores while N_i is the number of impostor trials. In cases of non-overlapping distributions, F expresses the separation between them as:

$$F_{non-overlap} = 1 - \frac{GEN_{max}}{2 \cdot IMP_{min}} \qquad (2)$$

where $GEN_{max} (\geq 0)$ is the maximum genuine score and IMP_{min} is the minimum impostor score. The factor of 2 in equations 1 and 2 sets F to be 0.5 at the overlap/non-overlap boundary.

5 EXPERIMENTS

To provide flexibility for testing our text-dependent verifier, the same text was spoken by all the subjects in the first two experiments. The phrase *ABCDEFG* was chosen for the following reasons. The final verifier system would (ideally) not use input any longer than this (about 3 seconds); the text is familiar enough for the subjects to speak naturally; and it could be segmented easily to produce tokens of varying length. (It should be noted that an exactly linear measure of length was sacrificed in order to preserve the continuity of natural speech, by segmenting at word boundaries.)

5.1 Expt. 1: Forward Segmentation

To obtain progressively longer tokens, the first n letters were extracted from the phrase, i.e. A, AB and so on, until the full-length utterance was used. For each designated genuine speaker and for each length of utterance, there were 15 self-tests and 135 (15 x 9) impostor tests. The scores from these tests were plotted as histograms for analysis of the genuine and impostor distributions, as depicted in Figure 1.

Inspection of the histograms revealed a common pattern: the genuine speaker's distribution is narrower and has a lower mean

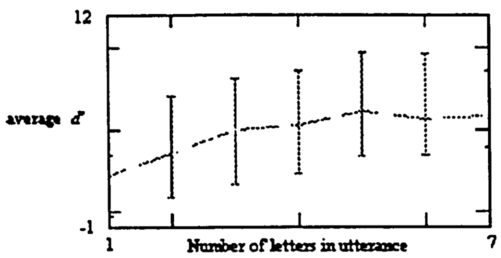

Figure 2: Speaker separation (as measured by average d') as a function of the number of letters in the verification token. Error bars show the range of d' values.

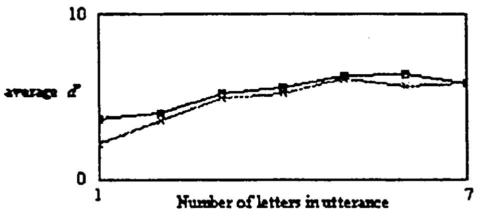

Figure 3: Speaker separation (as measured by average d') as a function of the number of words in the verification token, for both forward (crosses) and reverse (boxes) segmented utterances.

than that of the impostor set. This is to be expected if the verifier works correctly and the subjects in the database represent a broad population. Also, judging by the area of overlap of the the distributions, the performance of the verifier appeared to improve as longer tokens were used. This observation is confirmed by examination of both of the performance measures defined in Section 4 above. Figure 2 shows a graph of d' (averaged over all subjects) against the number of letters in the verification token. The error bars in the graph indicate the range of values over all the subjects. It was found that values of d' above (approximately) 4 corresponded with complete separation. It can be seen that the overall trend is for an increase in d' with length. There is some suggestion of saturation at $n = 5$ letters (about 2 seconds of speech). Although the data are not shown here, for some individuals only 2 letters were needed to give complete separation.

5.2 Expt. 2: Reverse Segmentation

To test that the above result was not an artifact of the position of specific words within the phrase (i.e. depended on *content* rather than length), segmentation was also done starting from the phrase end and producing progressively longer utterances by including only the last n words. The results of this experiment are similar to those for the forward segmentation: *viz.* an overall trend of improved performance with increased utterance length. Figure 3 shows the average d' values for both forward- and reverse-segmented utterances. For comparison, Figure 4 shows the corresponding average values of the overlap function F. The close agreement of results obtained with the two different measures is evidence of the improving effect on verifier performance of longer utterances.

The main difference between the results of this and experiment 1 is that reverse segmentation produced a slightly better one-word performance. One explanation might be that G is a phonetically richer word (containing more speaker-specific information) than A. Another possibility is that the first word in a connected-word utterance is not good for verification, perhaps

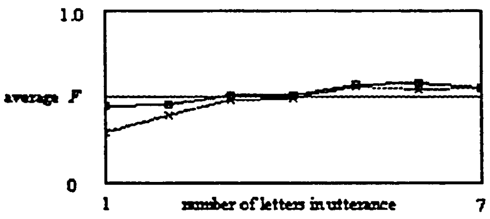

Figure 4: Speaker separation (as measured by average F) as a function of the number of words in the verification token, for both forward (crosses) and reverse (boxes) segmentation. The horizontal line at $F = 0.5$ corresponds to zero error rate.

as an effect of a high initial outflow of breath. In any event, the average performance using single-word tokens is poor.

5.3 Expt. 3: Concatenated Templates

We envisage a practical application of speaker verification working with the user prompted to enter short sequences of connected words in random order. Matching in this case would have to be against reference models for the *individual* words in the phrase. Thus, the test data could consist of a single, connected utterance but the reference data might consist of concatenated templates for the individual words prompted. Given the findings of experiments 1 and 2, it might be expected that the verifier would perform poorly because it is merely combining several single-word results, each of which is itself poor.

To test this, a subset of 5 of the 10 speakers in the original database provided additional recordings of 5 repetitions of the isolated words A, B, etc., up to G. After manual endpoint detection, the utterances were concatenated and combined as described in section 3 above to form another reference template of *ABCDEFG* for each subject. The new templates were tested with the same data as in experiments 1 and 2. The results were

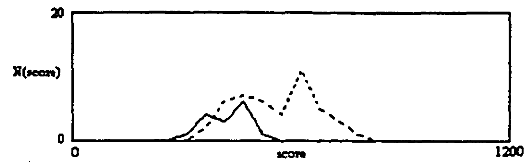

Figure 5: Typical genuine and corresponding impostor distributions produced in Experiment 3.

indeed very poor, none of them exhibiting any worthwhile degree of separation of genuine speaker and impostors. A typical pair of distributions is shown in Figure 5.

The distribution for the impostors was largely unaffected by the use of the new, concatenated templates but the genuine distribution shifted considerably in the direction of much increased average score. In effect, the genuine speaker was no longer distinguishable from the impostors. As mentioned before, high performance was not expected but the average values of d' and F were in fact lower than the single-word values in the previous experiments, suggesting other factors were at work. For example, a time-lapse between the recordings of the reference and test material is known to reduce performance [6, p. 498]. Also, from a separate experiment, we have discovered that the verifier is sensitive to the presence (or absence) of inter-word silence in the reference template if it is not duplicated in the test token. In an attempt to exclude these effects, we devised a fourth experiment.

5.4 Expt. 4: Discrete Utterances

For the previous experiment, the subjects had been recorded 5 times saying each of the discrete words A to G, and these recordings formed (by concatenation) the reference data for verification. The test data were the full-length utterances as in experiments 1 and 2. In this experiment, the discrete-word recordings provided *both* the test and reference tokens.

After manual end-point detection, the first of the 5 tokens of each discrete word was taken as reference for that speaker. Thus, for each word and each speaker, there were 4 genuine and 20 impostor tests. The scores for these comparisons were totalled for the 7 discrete words A to G, and d' and F values calculated. Table 1 shows the resulting d' values (the results for F were essentially the same) and compares them with those found in experiment 1.

The results in Table 1 show that separation attainable by summing scores for discrete-word matches can at least equal that obtained with continuous-word templates and test data. This tends to confirm the interpretation of experiment 3 given above, i.e. that word-boundary differences between test and reference data can cause severe problems for verification. Overall, the result is encouraging since it indicates that a practical system which operates by prompting the user for a number of discrete utterances in random order could work well.

subject	Expt. 4	Expt. 1
MH	12.05	8.11
JP	7.16	5.69
FS	5.64	6.38
AS	6.15	5.11
AF	5.95	8.56
average	7.39	6.77

Table 1: d' resulting from summing scores for discrete-word matches (Expt. 4) and by matching continuous utterances (Expt. 1).

6 CONCLUSIONS

Utterance length has been shown to have an important effect on speaker separation in text-dependent verification. Using continuous-word test and reference data, we find an increase in separation with length; there is some suggestion of saturation at about 2 seconds duration. In a practical system, it would be attractive to prompt the user for random-order utterances from a small vocabulary. We were unsuccessful in comparing continuous utterances with concatenated templates in this setting. This is apparently because of the problem that word-boundary effects pose to our verification technique. Simply totalling scores for matches between discrete-word reference and test data can, however, work well.

REFERENCES

[1] FURUI, S. (1981) "Cepstral analysis techniques for automatic speaker verification", *IEEE Trans. Acoustics, Speech and Signal Processing*, **ASSP-29**, 254–272.

[2] SOONG, F.K., ROSENBERG, A.E., RABINER, L.R., JUANG, B.H. (1985) "A vector quantization approach to speaker recognition", *Proc. IEEE ICASSP '85*, Tampa, FL, 387–390.

[3] BERNASCONI, C. (1990) "On instantaneous and transitional information for text-dependent speaker verification", *Speech Communication*, **9**, 129–139.

[4] ATAL, B.S. (1974) "Effectiveness of linear prediction characteristics of the speech wave for automatic speaker identification and verification", *Journal of the Acoustical Society of America*, **55**, 1304–1312.

[5] GREEN, D.M. AND SWETS, J.A. (1966) *Signal Detection Theory and Psychophysics*, Wiley, New York, NY.

[6] O'SHAUGHNESSY, D. (1987) *Speech Communication: Human and Machine*, Addison-Wesley, Reading, MA.

Using Genetic Algorithms to Improve Speaker-Verifier Performance

M.I. Hannah*, A.T. Sapeluk*, R.I. Damper† and I.M. Roger‡

*Department of Electronic and Electrical Engineering,
Dundee Institute of Technology
†Department of Electronics and Computer Science,
University of Southampton
‡Technology Development Group, NCR Dundee

Abstract

Previous studies have indicated that speaker-verification errors may be minimised by emphasising, or weighting, some speech features with respect to others. Traditionally, cepstral coefficients are used as the features and each weight is calculated using some deterministic function based on the variance of the features. Here, we consider derivation of the weights as an optimisation problem, where the goal is to separate maximally the feature distributions of the genuine speaker and the impostors. We use a genetic algorithm (GA) to derive these weights, and find the resulting verifier performance exceeds that obtained using traditional weighting functions by a significant margin. However, the success of the current implementation of the GA seems largely attributable to the random influence on the candidate vectors.

1 Introduction

Speaker-verification systems attempt to distinguish between the genuine individual and an impostor by comparing features extracted from samples of their speech. Atal (1974) demonstrated that accurate verification can be achieved by using cepstral coefficients to represent the speech features and by emphasising or *weighting* some of these features with respect to others. These findings have been confirmed by other researchers such as Velius (1988), who compared several different methods of deriving the cepstral weights. Most of these methods calculate the weight of each dimension as a deterministic function of the coefficient values of that dimension, and attempt to minimise inter-session intra-speaker variability. However, determination of the weight values can be considered as an optimisation problem, where the goal is to achieve a maximal separation of speech features between the genuine speaker and the impostor set. In this paper, we explore the use of a genetic algorithm (GA) to derive these weights, and compare the resulting verifier performance with that obtained using traditional weighting functions.

The paper is structured as follows. The details of the speech data used and of the verifier employed are described in Sections 2 and 3 respectively. Section 4 discusses the means of assessing verification performance while section 5 describes the methods of deriving the weights. The results of each method are presented in Section 6 and are discussed further in Section 7. Finally, Section 8 concludes.

2 Speech Data

Speech data for the experiments described in this paper were recorded from 10 young adults, 5 male and 5 female, in a quiet laboratory. An anti-alias filter with 4 kHz cut-off was used prior to sampling at a rate of 10 kHz and resolution of 16 bits. Each subject spoke the phrase ('text') *ABCDEFG* 20 times; 5 of these repetitions were used to form a reference template as described below, with the remaining 15 used for testing. Using the same text for all subjects allowed us to designate any subject as a *genuine* speaker and the other nine as *impostors*, although all were speaking naturally. Consequently, for each subject in the database, there

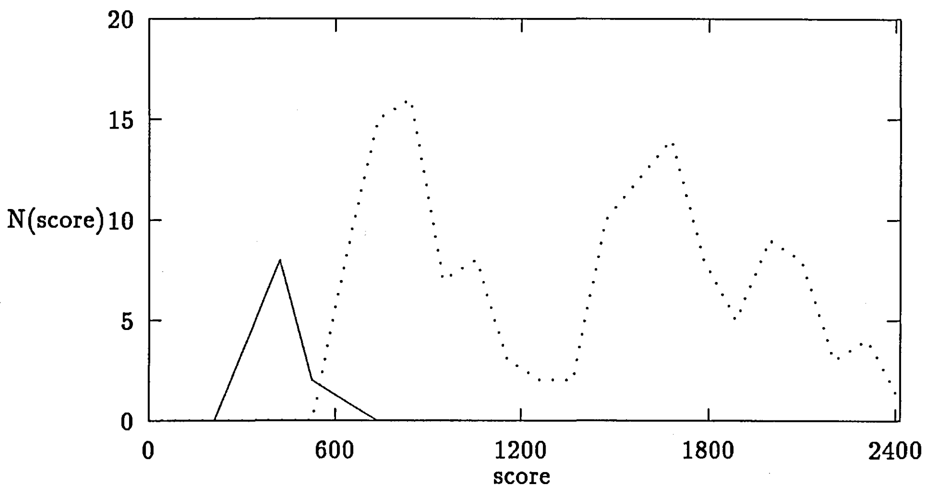

Figure 1: Typical distributions of *genuine* (solid line) and *impostor* (broken line) scores. The distributions are actually histograms (with a bin width of 25) with the mid-ordinates of each bin value joined into a line for clarity. The bimodal impostor distribution is typical when the genuine speaker is female.

were 15 genuine verification attempts and 135 (15 x 9) 'impostor' attempts. Although this is by no means an exhaustive test set, we consider it sufficient to demonstrate unequivocal changes in the performance of the verifier.

Each utterance was analysed in 25 ms non-overlapping frames using the lattice method of 12th-order linear predictive coding. Each frame was then described by a vector of 12 cepstral coefficients, obtained using the recursion described by Atal (1974). The average vector for the whole utterance was then subtracted from each individual vector to produce a time sequence of normalised cepstral vectors, which formed the input to the verifier.

3 Verifier Operation

The verifier is an un-optimised version based on the principles detailed in Furui (1981) and Bernasconi (1990). It relies on matching a test utterance to reference patterns ('templates') using dynamic time-warping (DTW) to produce a score, where 0 indicates a perfect match and higher scores indicate progressive degrees of mis-match. As stated above, the reference template for each speaker was formed using a subset of 5 of their utterances; each was DTW-matched to every other, and the utterance with the lowest accumulated score was adjudged to be the centroid of the group. The other 4 utterances were then time-aligned to the centroid and all 5 averaged to form the template. None of the utterances used in the construction of the template were ever used in testing.

4 Assessing Performance

The performance metric traditionally used in speaker verification is the error rate. However, there are two possible types of error: false acceptance and false rejection. A general statement of the accuracy of a system must cover both cases because it is possible to bias the decision towards minimisation of one error at the expense of the other. Hence, the *equal-error rate* is often quoted, where the probability of each type of error is equalised by the setting of an appropriate decision threshold. That is, an identity claim from an individual is accepted (rejected) if the verification score is less than (more than) the threshold.

Due to the inherent variability of individual voices, a single, global threshold cannot be guaranteed optimal for every speaker; instead, individual thresholds are needed. However, the thresholds should themselves depend on the distribution of scores produced by the matching procedure – such as those shown in Figure 1 – and so cannot be set in advance. Also, if total separation (non-overlap) of score distributions occurs, the *degree* of separation cannot be conveyed by simply quoting a error rate of zero – yet with a limited test set a zero error rate is not uncommon. For these reasons, when considering verifier performance, we focus on the separation and/or overlap of score distributions for the genuine speaker and the impostors. To help in this assessment, we employ two figures of merit: a version of the classical d' and our own 'overlap' function, F. Most of the results in Section 6 quote values of d' because it includes a measure of dispersion of the distributions. Where speed is important, such as in determining the fitness of a candidate weighting vector, F is used since it is simpler to compute.

4.1 Using d'

The d' measure from classical signal detection theory (Green and Swets 1966) assumes the existence of two response distributions (signal and signal-plus-noise, corresponding here to genuine and impostor) which are Gaussian and equal variance. Then:

$$d' = \frac{|\mu_s - \mu_{sn}|}{\sigma}$$

where μ_s and μ_{sn} are the means of the signal and signal-plus-noise score distributions respectively, and σ^2 is the common variance. In our case, the assumptions are invalid but, as our results show, d' seems nonetheless to provide a reasonably useful measure of separation. We have modified the original to compensate for the inequality of variances thus:

$$d' = \frac{\mu_{imp} - \mu_{gen}}{S}$$

where μ_{imp} and μ_{gen} are the means of the impostor and genuine distributions respectively, and S is the geometric mean of the standard deviations of the two distributions. Note that (unlike the situation in signal detection theory) d' can be negative in the (highly undesirable!) case that $\mu_{gen} > \mu_{imp}$.

4.2 Overlap Function (F)

We have developed an alternative measure of verifier performance based on the so-called 'overlap' function, F. This is an easily-computed, *ad hoc* figure which has a valid range of [0, 1], although this is split into two equal sub-ranges to cover the distinct cases of overlap and non-overlap, or (total) separation. F is computed differently for each case (labelled $F_{overlap}$ and $F_{non-overlap}$) but increases linearly with verifier performance over the entire range. $F_{overlap}$ is inversely related to the probability of an impostor's utterance producing a score that is within the range of the genuine speaker's scores:

$$F_{overlap} = 0.5 - \frac{f}{2N_i} \tag{1}$$

where f is the number of 'failures' in terms of an impostor score being within the range of genuine scores while N_i is the number of impostor trials. In cases of non-overlapping distributions, F expresses the separation between them as:

$$F_{non-overlap} = 1 - \frac{GEN_{max}}{2 \cdot IMP_{min}} \tag{2}$$

where $GEN_{max} (\geq 0)$ is the maximum genuine score and IMP_{min} is the minimum impostor score. The factor of 2 in equations 1 and 2 adjusts F to be 0.5 at the overlap/non-overlap boundary.

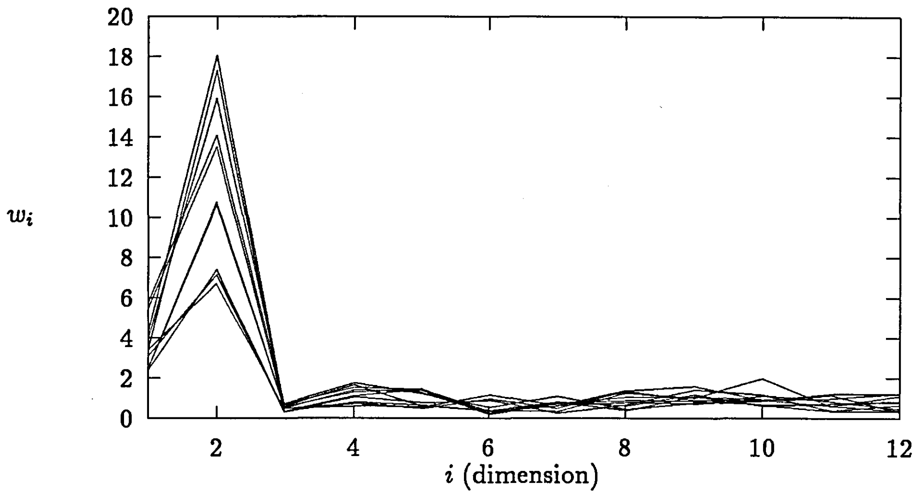

Figure 2: Each trace shows one speaker's weights calculated with the inverse-variance function.

5 Deriving the Weights

The DTW process performs a frame-by-frame comparison of a test token and a reference template. At the frame level, the comparison takes the form of a distance measurement, where the squared differences between corresponding dimensions are summed for all n (=12) dimensions. When weights are included, the general expression for the local distance d between a test frame t and a reference frame r is:

$$d = \sum_{i=1}^{n} w_i (t_i - r_i)^2$$

Previous speaker-verification studies using weighted cepstral distance measures (such as those summarised by Velius in 1988) have been successful in reducing error rates. However, the derivation of the weights has been based on theoretical assumptions rather than using an optimisation procedure to yield the best practical results. In this paper, we will compare the GA-derived weights with two of these traditional measures: the inverse-variance function and the inverse-expected-difference function. We will also include results for the unweighted, or 'uniform weighted' case (all weights set to 1). Before discussing the results we define (except for the unweighted case) the various weighting schemes to be examined.

5.1 Inverse Variance

One common weighting function is the reciprocal of the intra-speaker variance of each dimension of the cepstral vector. This is an acknowledgement that feature differences in certain dimensions may be more significant than in other dimensions. There is also an assumption that features which remain consistent over an entire utterance, or over several utterances by the same speaker, may characterise that speaker. For each of the speakers in the database, the reciprocal of the variance of each normalised cepstral coefficient was calculated over each of 15 utterances and averaged to produce a set of inverse-variance weights for that speaker. The weights for all 10 speakers are plotted on the same graph (Figure 2) to demonstrate the similarity of the function across the speakers. Clearly, the first two dimensions are weighted much more heavily than the others.

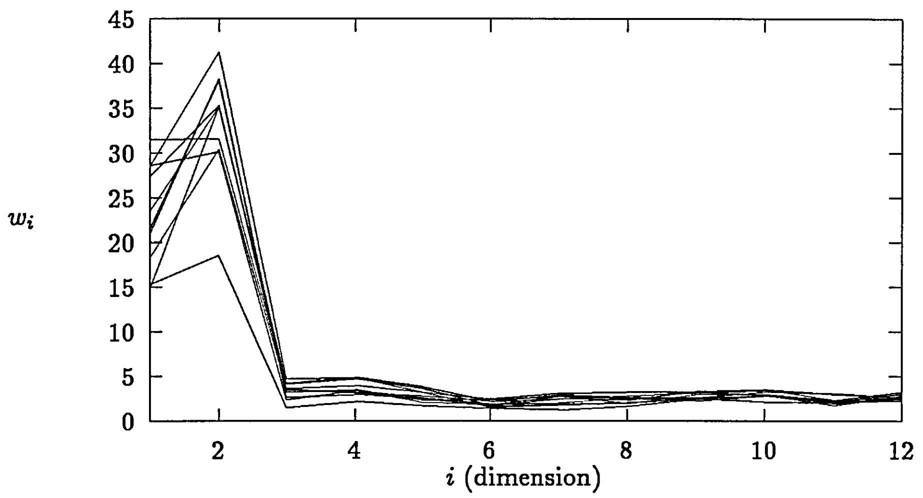

Figure 3: The inverse-expected-difference weighting function, plotted for all speakers.

5.2 Inverse Expected-Difference

This function, as used by Furui (1981), normalises the dimensions by their so-called *expected* differences after time-alignment, for the intra-speaker case. This ensures that cepstral coefficients may contribute equally to the overall distance measure, regardless of numerical scale, which is not an indicator of utility. Test utterances from each speaker were time-aligned with their reference template and for each of the 12 dimensions, the average difference between corresponding frames was calculated over the entire utterance. Averaging over all 15 test utterances produced the expected differences for each dimension. The reciprocals of these were used as the weights in this case, and are plotted in Figure 3 for all speakers to highlight the similarities. As in the inverse-variance case, weights w_1 and w_2 are stronger than the rest.

5.3 GA-generated Weights

Using the above methods, each individual weight is calculated by applying the same, deterministic function which, despite having a basis in theory, nonetheless restricts the available solutions. An alternative is to allow the individual weights to be derived independently, yet considering them collectively as a vector. The genetic algorithm is an appropriate method of doing this, as no assumptions are made about which dimensions are important for verification, or about the properties of cepstral coefficients of those dimensions. The GA has no constraints on either the starting point or the extent of the search space of solutions, and should therefore be capable of finding high-performance weighting vectors, should they exist.

5.4 Implementation

For the genetic algorithm to generate weighting vectors, an appropriate form of coding must be found. We decided to allow each weight to vary between 0 and 15 so that it could be encoded simply using 4 bits. By concatenating all 12 of the 4-bit weights, the weighting vector could be encoded as a 48-bit string, or 'chromosome' in GA terminology. In the general case, when using n-dimensional cepstral vectors, and allowing each weight to be quantised to m levels, there are $2^{n \log_2 m}$ different weighting vectors from which to choose. In our case, the range of choice is 2^{48}, which is clearly too large to search exhaustively[1]. Also, there is no guarantee (quite the opposite!) that the solutions are ordered (in terms of verifier performance) within

[1]In fact, there are less than 2^{48} *unique* solutions because some of the vectors are linearly dependent. For instance, vectors $\vec{x} = (1,1,1,1,1,1,1,1,1,1,1,1)$ and $\vec{y} = (2,2,2,2,2,2,2,2,2,2,2,2)$ both give identical separation statistics/error-rates when used as the weighting vector.

Speaker	Inverse-Variance	Inverse Expected Difference	Best GA	Unweighted
AF	7.516	8.308	9.048	8.563
AM	3.920	4.088	4.115	3.927
AS	7.977	6.051	8.935	5.109
MH	8.798	7.674	8.498	8.106
NT	6.045	6.183	8.139	5.454
BK	4.215	4.911	4.847	4.392
FS	6.119	6.235	6.476	6.379
JP	5.322	4.398	5.398	5.687
KM	3.862	3.963	5.179	4.066
TM	5.626	6.237	6.798	6.215
Average	5.940	5.805	6.743	5.790

Table 1: d' of score distributions for each speaker, with different weighting functions.

this range. Traditional search methods, such as gradient descent, are therefore probably unsuitable but this type of problem is appropriate for solution using the GA.

Following the 'Simple Genetic Algorithm' of Goldberg (1989), our implementation was as follows. The initial population of candidate vectors was generated by randomly setting bits in the chromosomes to 1 or 0 with equal probability. At each generation, only the three basic operators were employed: roulette-wheel selection, single-point crossover (with probability P_{cross}) and mutation (with probability P_m). The results presented in Section 6 were obtained with a GA configured with a population size of 32, $P_{cross} = 0.7$, $P_m = 0.02$ and, unless stated otherwise, the program ran for 50 generations. The algorithm was coded in C and executed on a standard PC with maths co-processor.

A vital consideration when using a GA is the specification of the objective function which determines the *fitness* of a candidate solution. Clearly, in the present work, this should be some appropriate measure of verification performance. For the reasons given in Section 4, our objective function does not assess the 'accuracy' of the system, but instead focuses on the separation of the score distributions for the genuine speaker and the impostor. Since fitness has to be repeatedly computed for each generation, while the GA was running weighting vectors were assessed using the F value for the corresponding score distributions. Finally, however, d' was preferred for scoring what appeared to be the 'best' weighting vectors.

6 Results

For each of the 10 speakers, the speaker separation was assessed using 4 differently-derived weight vectors: inverse-variance, inverse-expected-difference, genetic algorithm and unweighted (all weights set to 1). In the case of the GA, the vector used was the best one discovered within the first 50 generations on the basis of the F measure.

6.1 GA versus Traditional Methods

Table 1 shows the d' values for the best vectors that the GA found for each speaker and compares these with the corresponding values when using the traditional methods. We have found that d' tends to range from approximately 3.0 (very poor performance) to about 10.5 (considerable separation of genuine and impostors). With this in mind, it can be seen that the GA-derived weights out-performed the three traditional methods, for each speaker, with an average d' of 6.743, a gain of nearly 14% on the next-best method: the inverse-variance. It is also interesting to note that the unweighted verifier's performance was only marginally worse than the traditionally-weighted verifiers.

For comparison with Figures 2 and 3, the best weighting vectors found by the GA for all speakers are plotted on a single graph: Figure 4. As can be seen, the pattern of weights that was apparent in the

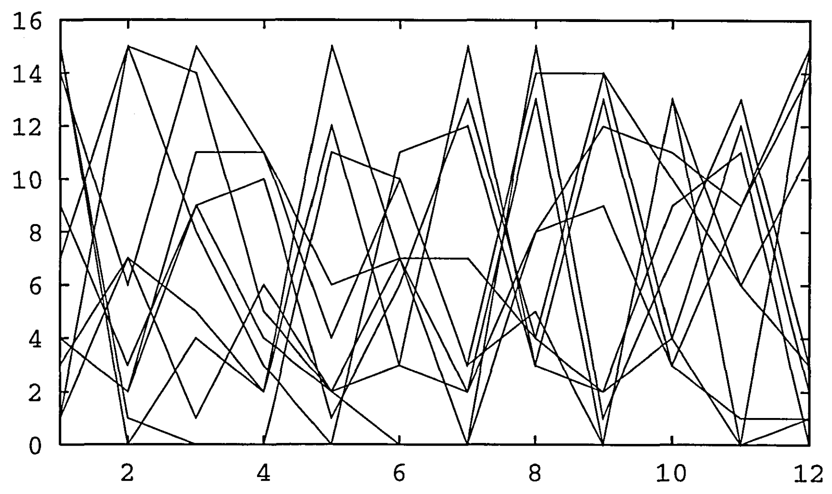

Figure 4: The best weighting vectors found by the GA, for every speaker. Unlike the traditional weighting functions, no order or pattern is apparent.

inverse-variance and inverse-expected-difference methods is not repeated. Indeed, the weights appear to be uncorrelated among all the speakers. This may be taken as evidence that different weighting vectors are needed for each individual. It is also interesting to see that, in general, the whole dynamic range of the weights is used, unlike the traditional methods.

6.2 GA versus Random Optimisation

To find out if the reason for the success of the GA was the random influence, we ran a program which simulated the GA without any of the standard operators; this is equivalent to generating each population completely at random, with no regard for previous generations and fitness levels etc. The program was run for each subject in the database, producing 50 populations of 32 randomly-generated weighting vectors; again a record was kept of the best vectors for each subject, and the average fitness of each generation. Table 2 compares the best vectors for each speaker, as found by the GA and the random program.

From Tables 1 and 2, it is apparent that the randomly-chosen weights actually give greater separation between the average speaker and impostors than weights selected by any other method. This is quite surprising in itself; but also it raises questions about the mechanisms in the GA for converging on optimal solutions. Perhaps the GA found good vectors solely because of the randomness of the search? One way to test this is to examine the evolution with time ('generation') of average population fitness for both the GA and the random program. Figure 5 plots these quantities for a single speaker, for a cycle of 50 generations.

Figure 5 shows that, for this speaker, average population fitness (measured using the F function) follows a slightly upward trend, whereas the average fitness of each randomly-generated population remains at a fairly steady level. This pattern is typical of all the speakers in the database, but for this speaker in particular, the upward trend is pronounced[2]. To some extent, this result restores faith in the GA: it finds collections of good vectors at the expense of finding the single best-performing vector.

6.3 Summary

Table 3 shows the best results averaged across all speakers, for each method. The results are expressed as both d' and equal-error rate (EER), since the latter is more familiar.

[2]An extra experiment was conducted for this speaker where the GA was left to run for 200 generations; the average fitness continued to climb, finishing at about $F = 0.480$.

Speaker	Best GA	Best random
AF	9.048	9.548
AM	4.115	4.848
AS	8.935	9.129
MH	8.498	10.472
NT	8.139	8.737
BK	4.847	5.440
FS	6.476	7.577
JP	5.398	6.188
KM	5.179	4.354
TM	6.798	7.540
Average	6.743	7.383

Table 2: d' of score distributions for each speaker, comparing the GA-produced weights with random weights.

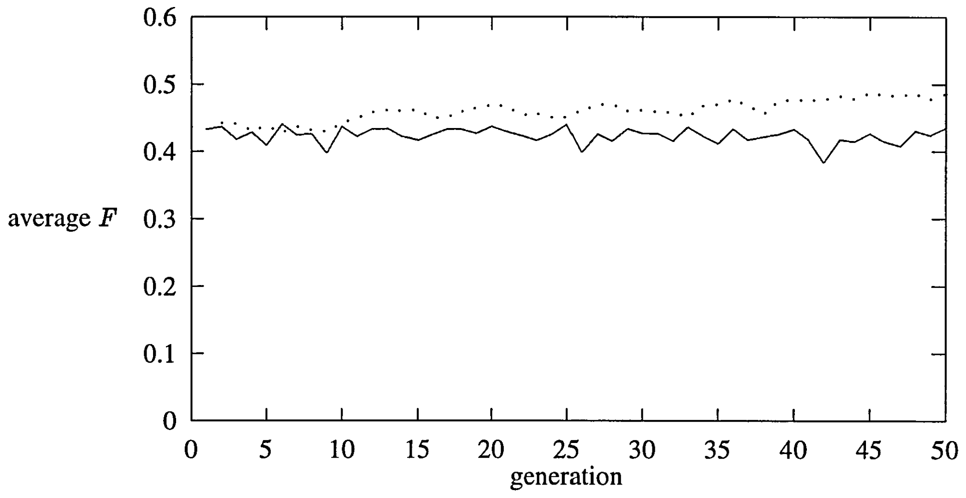

Figure 5: Fitness, averaged over the population at each generation, for both the GA (broken line) and the random program (solid line).

Weight derivation method	average d'	average EER (%)
Random	7.383	0.67
GA	6.743	0.67
Inverse-variance	5.940	2.67
Inverse-expected difference	5.805	2.67
Unweighted	5.79	2.67

Table 3: A summary of the verifier performance with various weighting methods. Figures are averages over all speakers.

7 Discussion

Why do random (or GA-produced) vectors do well? We can postulate that there exist a few simple rules relating a given pair of speakers to characteristics of cepstral coefficients. For instance, 'to distinguish between speaker A and speaker B, maximise the ratio of the weights for coefficients 1 and 3 but minimise the ratio for coefficients 9 and 10'. These rules would themselves be based on the behaviour of the coefficients for several utterances by each individual speaker. Now, since there are 10 speakers in our database, it is possible that some of these speaker-specific rules actually overlap, resulting in conflicting speaker-pair rules. The GA (and random) vectors may be able to find some near-optimal compromise of all of these individual rules, but the traditional methods, which treat each dimension in isolation, cannot. One way to test this theory is to use the best vector for each speaker but compare with a different impostor set, because we may expect that a new set of impostors would bring a new set of compromised rules.

The apparently slow convergence of the GA may be attributable to the simplicity of the algorithm used. In the near future we intend to repeat this series of experiments but using more sophisticated genetic operators, such as elitism and multi-point crossover. Another enhancement would be to allow the chromosome to represent real-valued weights, although the current system does not seem to lack precision, so this will be given a low priority.

In a practical verification system, it is unlikely that a GA could be used on-line due to the required computer time and the unavailability of appropriate impostor data. However, we have observed that – in cases of good speaker separation – the genuine speaker distribution tends to be quite narrow. This means it might be possible merely to use the standard deviation of this distribution as the principal input to the objective function. We have already begun to work on this topic.

8 Conclusions

We have compared the performance of a speaker verifier using traditional weighting functions with one using GA-derived weights. In terms of speaker separation, and for our limited test set, we find the latter to be superior. However, we have also discovered that simply generating the weights at random can yield better performance than the GA-derived weighting vectors. This may reflect the simplicity of random optimisation: perhaps a different chromosome design or the use of different genetic operators would enable the GA to out-perform the random approach. The relative failure of the traditional methods may be the result of considering each dimension in isolation, which the random and GA program do not do.

References

ATAL, B.S. (1974) "Effectiveness of linear prediction characteristics of the speech wave for automatic speaker identification and verification", *Journal of the Acoustical Society of America*, **55**, 1304–1312.

BERNASCONI, C. (1990) "On instantaneous and transitional information for text-dependent speaker verification", *Speech Communication*, **9**, 129–139.

FURUI, S. (1981) "Cepstral analysis techniques for automatic speaker verification", *IEEE Transactions on Acoustics, Speech and Signal Processing*, **ASSP-29**, 254–272.

GOLDBERG, D.E. (1989) *Genetic Algorithms in Search, Optimization and Machine Learning*, Addison-Wesley, Reading, MA.

GREEN, D.M. AND SWETS, J.A. (1966) *Signal Detection Theory and Psychophysics*, John Wiley, New York, NY.

HOLLAND, J.H. (1975) *Adaptation in Natural and Artificial Systems*, University of Michigan Press, Ann Arbor, MI.

VELIUS, G. (1988) "Variants of cepstrum based speaker identity verification", *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, New York, NY, Vol. 5, 583–586.

The Rôle of the Reference Template in Speaker Verification

M.I. Hannah, A.T. Sapeluk and R.I. Damper

Abstract— **This paper compares 4 different ways of using initial ('training') recordings as reference data ('templates') to be compared with a test token for speaker verification. We show that merging the original training data into a single template – as in 2 of the 4 methods – incurs a loss in speaker discrimination relative to the remaining 2 methods which retain all the data. However, these techniques do have the advantage of reduced computational complexity in terms of both storage requirements and processing time for verification.**

The best-performing of our methods used a decision rule in which 60% of the individual matches to each retained template had to satisfy a threshold criterion. However, statistical significance of the superiority of this implementation relative to the next-best, which took a mean of the 5 individual scores as a basis for its accept/reject decision, has not yet been established. This awaits further work with a larger database.

Keywords— **Speaker verification, text-dependent verification, reference template.**

1. Introduction

Most published studies of speaker-verification systems have tended to concentrate on optimising the performance of the matching process, in which a test utterance is compared with reference data assumed to be representative of the given speaker. In text-dependent verification, these reference data are most usually held in the form of *templates*. However, for a verifier to be consistently successful it is vital that the templates are constructed properly from the training data. In this paper, we examine 4 methods of building reference templates and assess the corresponding verifier performance in each case. Following Furui [1] and Bernasconi [2], the verifier used in this study employs dynamic time-warping (DTW) matching of test and reference utterances represented as sequences of cepstral coefficients obtained from linear prediction analysis.

The paper is structured as follows. The details of the speech data used and of the verifier employed are described in Section 2. Section 3 discusses the means of assessing verifier performance while Section 4 describes the four different methods of constructing the template. The results of each method are presented in Section 5 and are discussed further in Section 6.

M.I. Hannah and A.T. Sapeluk are with the Department of Electronic and Electrical Engineering at Dundee Institute of Technology, Bell Street, Dundee DD1 1HG, Scotland. Email: eermih@uk.ac.dct and eetats@uk.ac.dct respectively.

R.I. Damper is with the Department of Electronics and Computer Science at the University of Southampton, Southampton SO9 5NH, England. Email: rid@uk.ac.soton.ecs.

2. Speech Data and Verifier Operation

Speech data for the experiments described in this paper were recorded from 10 young adults, 5 male and 5 female, in a quiet laboratory. An anti-alias filter with 4 kHz cut-off was used prior to sampling at a rate of 10 kHz and resolution of 16 bits. Each subject spoke the phrase ('text') *ABCDEFG* 20 times; 5 of these repetitions were used to form a reference template as described below, with the remaining 15 used for testing. Using the same text for all subjects allowed us to designate any subject as a *genuine* speaker and the other nine as *impostors*, although all were speaking naturally. Consequently, for each subject in the database, there were 15 genuine verification attempts and 135 (15×9) 'impostor' attempts. Although this is by no means a large test set, it should be sufficient to demonstrate unequivocal changes in the performance of the verifier.

Each utterance was analysed in 25 ms non-overlapping frames using the lattice method of 12th-order linear predictive coding. Each frame was then described by a vector of 12 cepstral coefficients, obtained using the recursion described by Atal [3]. The average vector for the whole utterance was then subtracted from each individual vector to produce a time sequence of normalised cepstral vectors, which formed the input to the verifier.

The verifier is an un-optimised implementation based on the principles detailed in [1] and [2]. Sequences of cepstral vectors for the test input are matched to similar sequences for the reference data (templates) using dynamic time-warping to produce a distance measure, or score, where 0 indicates a perfect match and higher scores indicate progressive degrees of mis-match.

3. Assessing Performance

Our previous investigations [4], [5] have led us to develop methods other than the equal-error rate (EER), which is less than ideal for preliminary assessment of verifier performance. This is because we are, at this stage, dealing with relatively small data sets. The basis of the problem with EER is that it is all too easy to obtain 100% correct classification (0% error rate) with different implementations, yet these implementations are not all equally good. We prefer to use some measure of the *separation* of genuine speaker and impostor score distributions. Our chosen figure of merit is a version of d' from classical signal detection theory [6]. We use:

$$d' = \frac{\mu_{\text{imp}} - \mu_{\text{gen}}}{S} \qquad (1)$$

where μ_{imp} and μ_{gen} are the means of the impostor and genuine distributions respectively, and S is the geometric

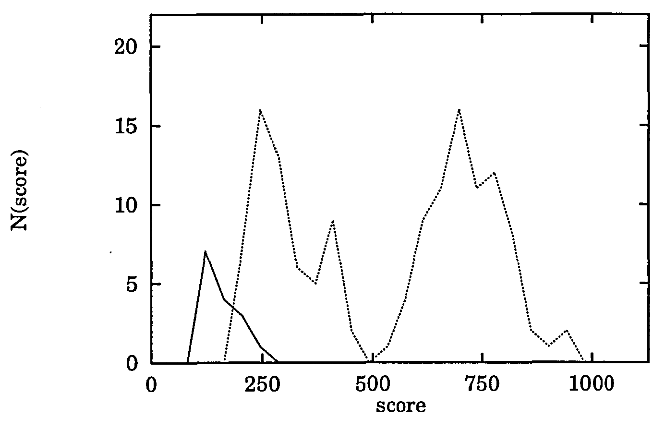

Fig. 1. Distributions of *genuine* (solid line) and *impostor* (broken line) distance scores for female speaker KM. The distributions are actually histograms (with a bin width of 25) with the mid-ordinates of each bin value joined by a straight line for clarity. We refer to the region of overlap between the two distributions as the *ambiguous zone*. The bimodal impostor distribution is typical when the genuine speaker is female.

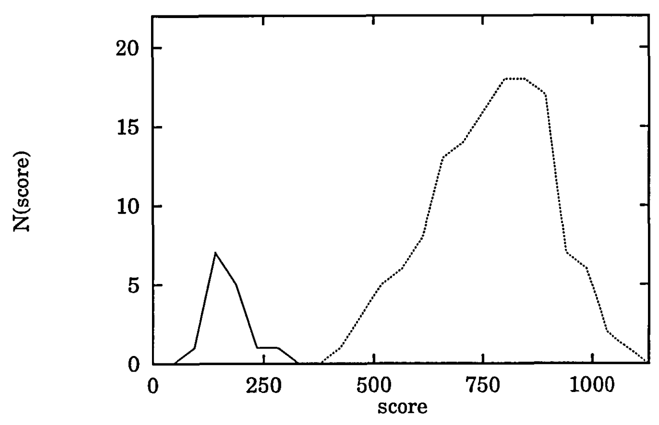

Fig. 2. Distributions of *genuine* (solid line) and *impostor* (broken line) distance scores for male speaker MH. Note that the impostor distribution is unimodal in this case, and there is no overlap of genuine and impostor scores. This latter feature makes use of an error rate as a figure of performance problematic (see text).

mean of the standard deviations of the two distributions. Typical *genuine* and *impostor* distributions are shown in Figures 1 and Figures 2 for a female and male speaker respectively.

Note that d' not only conveys information about the degree of separation (or overlap) of the two distributions. Significantly, it also includes a measure of their dispersions in terms of S. Further discussion of the use of d' can be found in [4], [5].

4. Constructing the Reference Templates

In this work, we have used speech data recorded in an initial pre-testing ('training') session to construct a template data set. These are then used as the reference against which all test tokens are compared. The template data set may consist of a single token or a collection of several tokens, although the initial recording session always involves 5 repetitions of the same phrase, in this case *ABCDEFG*. Here, we assess 4 different ways of constructing the template data set, namely:

- Furui method
- Bernasconi method
- Mean Separate Score (MSS) method
- 60% rule

The first two of these combine the original 5 utterances to form a *single* reference token; the latter two retain all 5 utterances as distinct tokens which are used at verification time. The obvious disadvantage of the extra time needed to do 5 verifications instead of one may be offset by greater flexibility in both the decision mechanism and any scheme which may be introduced for updating the templates to account for long-term variations in the customer's voice.

4.1 Method 1: Furui

Furui [1] developed and tested techniques for implementing each of the processes inherent in text-dependent speaker verification. His method of constructing a reference template is straight-forward, and is as follows. The first of the 5 reference utterances is designated the initial template, to which the second utterance is then time-aligned by DTW. An average of the two patterns is then taken to produce a new template, to which the third utterance is time-aligned. Again, an average is taken, and the process repeated until all 5 utterances have been combined into a single template.

Note that the length of the final template is equal to that of the initially-chosen utterance, irrespective of whether this utterance is typical in respect of its length.

4.2 Method 2: Bernasconi

The verifier used by Bernasconi [2] is very similar to that developed by Furui, although the means of template construction is different. Bernasconi's method calculates DTW distances between all pairs of the original utterances so that, in the present case, each of the 5 utterances is compared to the remaining 4. The utterance with the lowest accumulated distance over the remaining 4 is taken to be the centroid of the group. The remaining 4 utterances are then time-aligned to the centroid, before averaging all 5 to produce the template.

4.3 Method 3: MSS

In this case, a test token is separately compared with each of the 5 initial reference utterances. The resulting scores are then averaged to give an overall score.

4.4 Method 4: 60% Rule

This method is similar to the previous one in that a test token is separately matched against each of the 5 reference tokens. However, each of these verification scores is then compared to a pre-determined threshold. If 60% of the scores (i.e. for 3 or more of the 5 matches) are less than this threshold, an "accept" decision is signified; otherwise, the speaker is rejected.

The threshold for speaker j, T_j, is calculated using the formula suggested by Furui [1]:

$$T_j = a(\mu_{\text{imp},j} - \sigma_{\text{imp},j}) + b$$

where $\mu_{\text{imp},j}$ is the mean of the distribution of impostor scores for speaker j, and $\sigma_{\text{imp},j}$ is the standard deviation of the same distribution. For each speaker, values for a and b are found such that the resulting threshold produces equal rates of false-acceptance and false-rejection errors. These *local* constants are then averaged over all speakers to give *global* constants, with which each speaker's threshold is re-calculated.

In order to compare this approach with Methods 1–3 on the basis of d', it is necessary to derive some overall score measure for a test utterance so that genuine and impostor score histograms can be constructed. This is done as follows. If the decision according to the 60% rule is "accept", then the overall score is taken as the average of those (3 or more) individual verification scores which fall below the threshold. On the other hand, if the decision is "reject", then those (3 or more) scores greater then the threshold are averaged to give the overall score.

5. Results

The bar chart in Figure 3 shows d' computed for the score distributions for each speaker, using all 4 methods of template construction. In interpreting these figures, the reader should recall from equation 1 that a d' value of 5, for example, indicates that the separation of genuine and impostor distribution means is 5 times the common standard deviation (as assessed from the geometric mean of the individual deviations).

The first 5 speakers (BK–TM) are female while the remaining 5 are male. Although there is an apparent tendency for the males to perform better than the females, the best performing males AS and MH are, in fact, two of the authors of this paper and are experienced in the use of speaker verification systems. Further, the best performing female (FS) is a speech therapist. None of the remaining subjects have any significant speech science or speech technology expertise. Thus, it does not seem possible to draw any firm conclusions about gender differences from Figure 3.

It can be seen that for some speakers (notably, JP, AF and AS), the method of template construction can greatly influence the verifier performance associated with those speakers. With the exception of AS and NT, Method 1 gives the poorest results and this is reflected in the low mean d', as shown in Table I. This table also shows an *ambiguity rate*, which measures the percentage of obtained verification scores in the ambiguous zone, i.e. the range of score values bounded by the maximum genuine score and the minimum impostor score (see Fig. 1). Although this measure can be distorted by atypical outliers, it does indicate any overlap of the distributions – information which

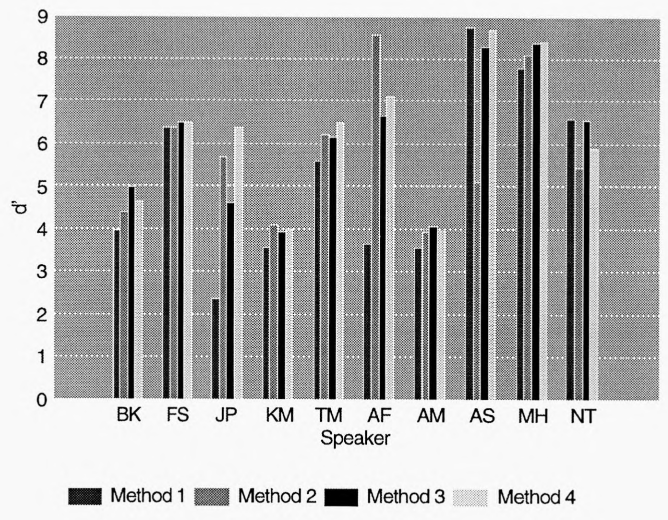

Fig. 3. Bar charts showing d' for score distributions for each speaker using different methods of template construction. (See text for description of these methods. The first 5 speakers (BK–TM) are female; the remaining 5 (AF–NT) are male.)

TABLE I
Comparing methods of template construction in terms of d' and ambiguity rate.

Method	Mean d'	Ambiguity Rate (%)
1	5.2	10.8
2	5.8	3.47
3	6.0	3.07
4	6.2	3.20

d' does not convey. The ambiguity rate is markedly higher for Method 1 than for any of the other approaches.

The reason that Method 1 performs poorly relative to Method 2 is almost certainly a result of the fact that the first utterance is always chosen as the initial reference, regardless of its suitability. By taking the centroid of the training patterns as the basis for time-alignment, Method 2 overcomes this problem. Methods 2 and 3 give roughly comparable results, indicating that choosing the centroid of the group of 5 utterances (Method 2) is similar in its effect to averaging scores (Method 3). Method 3 (MSS) produces slightly better results, but (given the small size of our data sets) the difference is unlikely to be statistically significant. Our expectation was that Method 2 would achieve a good compromise between accuracy and the computational convenience of a single-token template, and this is borne out by the obtained results.

Method 4 produces the best speaker-separation, as indicated by the highest d' averaged over all speakers in Table I. Again, however, this superiority is marginal and has not been subjected to a test of statistical significance. Further, ambiguity rate is slightly higher than for Method 3.

Method 4 (using the 60% rule) is based on deriving thresholds for each individual speaker. Given these thresh-

TABLE II
FALSE ACCEPTANCE AND REJECTION RATES, FAR:FRR (%), FOR ALL FIVE REFERENCE TOKENS 1–5, AND FOR THE 60% RULE (METHOD 4).

Speaker	1	2	3	4	5	60%
BK	5:13	0:20	1:7	0:7	0:47	0:13
FS	0:0	0:0	0:0	0:0	0:0	0:0
JP	1:0	1:0	1:0	6:0	2:20	1:0
KM	0:27	10:0	1:47	9:27	0:27	0:13
TM	1:0	1:0	0:0	4:0	0:0	1:0
AF	0:0	2:0	1:0	0:0	1:0	1:0
AM	4:20	8:7	2:7	8:13	15:13	4:13
AS	0:7	0:0	0:0	0:0	0:0	0:0
MH	1:0	1:0	0:0	0:0	0:0	0:0
NT	27:6	0:7	0:0	0:0	0:27	0:1

olds, we have a basis to estimate false acceptance and false rejection rates (FAR and FRR respectively). Table II shows such FAR and FRR figures both for the individual matches (to each of the 5 templates) and for the overall score based on the 60% rule. This table illustrates, as one might expect, the disparity of performance of the individual training tokens. Also, it shows the gain to be made using the 60% rule – in that this rule generally does only a little worse (if at all) than the best-performing reference, but significantly better than the average across reference tokens. For instance, for speaker NT the average FAR across the 5 reference tokens is 5.4%, but using the 60% rule it is 0%. For the same speaker, the FRR averaged across the 5 tokens is 8% while the 60% rule reduces this to 1%. The sole exception to this generalisation is the FRR for speaker AM which averages 12% across the 5 individual reference tokens, but is 13% using the 60% rule.

Overall, it is apparent from Table I that the approaches to constructing a template data set that average the reference data to form a single pattern (Methods 1 and 2, with an average d' of 5.5) do less well than those that retain the training data in their entirety (Methods 3 and 4, with an average d' of 6.1). However, the latter methods entail greater computational complexity in terms of both storage space and processing time at verification.

6. Conclusions and Discussion

We have compared 4 different methods of template data set construction and found superior overall performance when retaining the original reference tokens separate at verification time (Methods 3 and 4), rather than averaging training data to yield a single reference pattern (Methods 1 and 2). The best performing technique (Method 4) applied a decision rule (in this case, the 60% rule) to the individual distance scores rather than simply averaging them as in Method 3 (MSS). However, because of the paucity of data in our study so far, statistical significance of the difference between Methods 4 and 3 has not been demonstrated.

An analysis was conducted, however, in which false acceptance and false rejection rates were estimated both for the 5 individual templates matched separately and using the 60% rule. This analysis clearly revealed the gains which can be achieved by use of the 60% rule.

Further work is planned to study the sensitivity of these gains to the setting of the individual (single) thresholds. It may be that it is possible to do rather better in deriving a speaker threshold than using the method described in Section 4.4 above. We also intend to repeat the work described here on a larger database of speakers and utterances, so enabling us to assess the statistical significance of the differences between methods of template data set construction.

Finally, we have initiated some work (not yet in reportable form) on the impact of long-term changes in the customer's voice on verifier performance. We envisage a system in which poorly-performing templates in a pool of such reference tokens are identified and replaced by tokens derived from recent input to the system. The assumption is that the poor performance is a result of degradation with time, i.e. the current reference data are no longer representative of the user's current voice. The principal finding of this paper is that retaining reference tokens in such a pool is beneficial for performance, relative to averaging these data (although there is a price to be paid in terms of increased computational complexity). We take this as encouraging for the prospects of our envisaged system.

References

[1] S. Furui (1981), "Cepstral analysis techniques for automatic speaker verification", *IEEE Trans. Acoustics, Speech and Signal Processing*, **ASSP-29**, 254–272.

[2] C. Bernasconi (1990), "On instantaneous and transitional information for text-dependent speaker verification", *Speech Communication*, **9**, 129–139.

[3] B.S. Atal (1974), "Effectiveness of linear prediction characteristics of the speech wave for automatic speaker identification and verification", *Journal of the Acoustical Society of America*, **55**, 1304–1312.

[4] M.I. Hannah, A.T. Sapeluk, R.I. Damper and I.M Roger (1993a), "The effect of utterance length and content on speaker-verifier performance", *Proc. Eurospeech '93, Vol. 3*, 2299–2302, Berlin, Germany.

[5] M.I. Hannah, A.T. Sapeluk, R.I. Damper and I.M Roger (1993b), "Using genetic algorithms to improve speaker-verifier performance", *Proc. IEE/IEEE Workshop on Natural Algorithms in Signal Processing*, 24/1–24/9, Chelmsford, UK.

[6] D.M. Green and J.A. Swets (1966) *Signal Detection Theory and Psychophysics*, John Wiley, New York, NY.