UNIVERSITY OF SOUTHAMPTON

Theoretical and Practical Study of Audio-Visual Person Identification

by

Haoji Hu

A thesis submitted in partial fulfillment for the degree of Doctor of Philosophy

in the
Faculty of Engineering and Applied Science
Department of Electronics and Computer Science

November 2007

UNIVERSITY OF SOUTHAMPTON

ABSTRACT

FACULTY OF ENGINEERING AND APPLIED SCIENCE
DEPARTMENT OF ELECTRONICS AND COMPUTER SCIENCE

Doctor of Philosophy

THEORETICAL AND PRACTICAL STUDY OF AUDIO-VISUAL PERSON IDENTIFICATION

by Haoji Hu

There is an increasing interest in biometric identification but, unfortunately, identification based on only one classifier is unlikely to achieve acceptable performance for practical deployment. A potential way to overcome this is to combine results from more than one classifier.

This thesis is concerned with combining the audio biometric (voice) and the visual biometric (face) for person identification. To achieve this goal, a speaker identification classifier and a face identification classifier are built and tested on the XM2VTS database. In this thesis, we provide both theoretical and practical research work on combining these two classifiers for the purpose of achieving better identification results. Experiments indicate that our approach achieves very high identification rate on the XM2VTS database.

The main contributions of this thesis lie in three parts: first, we have proposed a new algorithm to adjust weighting parameter(s) for combining independent audio and visual signals; second, we have theoretically proved that there is no 'perfect' fusion algorithm suitable for all situations (the 'no panacea' theorem); third, we have built an audio-visual person identification system and achieved good performance on the XM2VTS database.

There are several directions for our future research work, which includes: (1) developing combination algorithms which are robust to noise and unpredictable situations; (2) combining visual features with the audio-visual classifier; (3) research work on face recognition; (4) generalising the method of finding optimal weighting parameter to the person verification cases; (5) theoretical study on multiple classifier combination; (6) building a real-time audio-visual person recognition system.

Contents

List of Figures

List of Tables

Acknowledgements

First and for most, I wish to thank my supervisors, R. I. Damper and Christine Shadle. Without their help and encouragement, this research work can not come into reality. I especially thank R. I. Damper for his comments and suggestions during our weekly meetings. His knowledge and experiences have always guided our research work in fruitful directions. I am also grateful to Christine Shadle, who was another supervisor during the first year of my PhD. Her time and patience has changed me to a researcher.

Over the past years I have also enjoyed working with many talented people in the Information: Signals, Images and Systems (ISIS) group, in particular Mark S. Nixon and John N. Carter, with whom I learned basic knowledge of computer vision, and A. Prügel-Bennett, whose suggestions are very helpful for my research. Present and former colleagues and friends in ISIS have made this group a wonderful research environment, including Steve R. Gunn, John S. Shawe-Taylor, Craig J. Saunders, Amanda L. Goodacre, Tracey Cantlie, Wenming Bian, Nagendra Mandaloju, Zakria Hussain, Jeremy D. Rogers, Andriy Kharechko, Zhonglun Cai, Jianqiang Yang, Xin Liu, Charanpal S. Dhanjal, Hidayah B. Rahmalan, Tasanawan Soonklang, Ziheng Zhou and M. S. Avila Garcia.

A very big 'Thank you' goes to my good friends in Southampton. My life would not be colourful if you were not here. Among them are Chris, Rebecca, Adam, Carol, Thomas, Yizhou Wang, Leilei Lee, Jian Luo, Xiang Liu, Cheng Gu, Cheng Lu, Chuan Bai, Tao Xin, Jiyu Chen and Shujuan Wang.

Finally, deep thanks to my parents and sister. Their consistent support is always crucial in my whole life.

Chapter 1

Introduction

Identifying a person is a straightforward task for human beings. We do it all the time in business and social encounters (Miller, 1994). We say 'Hello' to our friends in the morning; a bank clerk can compare a customer's signature to identify if he or she is the claimed person; we can easily know their identity when we talk with our friends through telephone. We are so accustomed to use the ability of identifying people that we do not pay much attention to the reasons why such an ability exists.

With the rapid progress of computer technology during the 70's of last century, researchers are more and more interested in building a computer system that can identify people automatically. When researchers began to delve into this area, they found there are many challenges and difficulties. Although the performance of biometric person identification systems has been improved significantly compared with some early systems, there are still many challenges that remain to be overcame in designing a completely automatic and reliable biometric system. The area of biometric person recognition has been receiving much attention in the previous several years. Especially after the events of September 11, 2001, the task of building a reliable biometric recognition system to prevent terrorism becomes an urgent endeavour for the governments of many countries, which has revitalised the research activities in this area.

A wide variety of biometrics have been marshalled in support of this challenging task, among which are fingerprints, handwritten signature, hand shape, ear shape, iris, face and voice (Miller, 1994). Some of these biometrics are shown in Figure 1.1. Each has its own advantages and disadvantages. For instance, fingerprint and iris are the most stable biometrics. They do not change dramatically with age, are not easy to alter or imitate, and are highly distinctive to an individual. But many people feel uncomfortable, even annoyed, when they are required to provide their fingerprints or let their irises be scanned. Other biometrics also have some kinds of advantages and disadvantages. Handwritten signatures are easy to imitate. Hand shape, ear shape and voice are not distinct biometrics because it is possible that different people have similar hand shape, ear shape or voice. Face recognition is a topic of active research during the last twenty years, but until now, there still remain some obstacles, such as pose and illumination

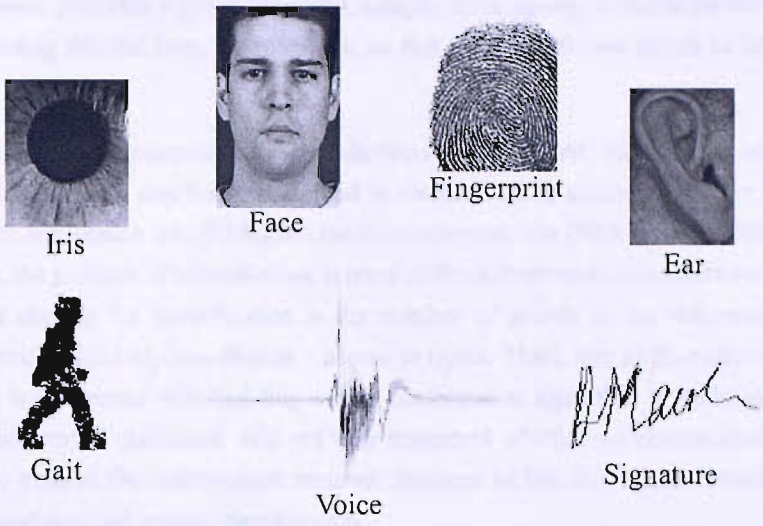


FIGURE 1.1: Some biometrics for person recognition, including iris, face, fingerprint, ear shape, gait, voice and signature etc.

variations, which hinder its further application (Chellappa *et al.*, 1995; Zhao *et al.*, 2003). Gait recognition provides a potential solution for identifying a person from a long distance or in noisy environments, but until now, its performance still needs to be improved.

Since recognition based on only one biometric is unlikely to achieve acceptable performance for practical deployment, researchers come to the idea of building a system by integrating different biometrics together in order to achieve better performance than each single biometric. An investigation of how people solve the recognition problem might be helpful to clarify the idea of integrating different biometrics. When we want to recognise a person, we use the information of his or her face, voice, and other biometrics, then we make decisions based on all these biometrics together. For example, if we look at a person from a long distance, the face is not clear, but we can obtain the recognition result by the voice; in situations when the voice is not clear, we can identify the person by the face. Because of this, it is also advantageous if the recognition system can obtain information of different biometrics, and integrate these biometrics by some methods to obtain better performance. This idea has the potential to provide a better solution than only using a single biometric. A particular situation in integrating biometrics for person recognition is the problem where basically two sources of information exist: audio signal (voice) and video signal (face). This problem is called audio-visual person recognition, which receives much attention lately because it provides a prospective solution to biometric recognition problems. Face and voice are very easy to obtain, and combination of these two makes the recognition system more robust to noise and complex environments.

Normally, tasks requiring biometric recognition can be divided into **verification** and **identification**. Verification involves making a decision to accept or reject the identity claim of a person, based on how well the biometric sample the person produced matches the reference sample which is produced by the claimed individual. Identification entails deciding which

registered person provides a given biometric sample from among a reference set of 'known' people, by finding the one from the reference set that gives the closest match to this biometric sample.

In this thesis, we only concentrate on identification because, first, all methods which can be used in identification can also be implemented in verification, by setting up a score threshold to adjust the false acceptance rate (FAR) and the false rejection rate (FRR) (Webb, 2002). Second, in most cases, the problem of identification is more difficult than verification because the number of alternative choices for identification is the number of people in the reference set, while verification process has only two choices – accept or reject. Third, one of the main contributions of this thesis is concerned with building a new combination algorithm to optimise the scores of the audio and visual classifiers. It is not very important whether we choose identification or verification to explore the combination method. Because of the above three reasons, we limit our scope to audio-visual person identification.

In order to build an audio-visual person identification system, a face identification system and a speaker identification system need to be built individually. Some combination algorithms also need to be specified to combine scores obtained from both systems into one identification result.

1.1 Contributions of this Thesis

The main contributions of this thesis concentrate on the area of combination algorithms for the audio-visual person identification system, which include:

1. Building an audio-visual person identification system. Implementing benchmark algorithms for face and speaker identification, which include Gaussian mixture models (GMM) for speaker identification (Chapter 2), principal component analysis (PCA) and dynamic link architecture (DLA) for face identification (Chapter 3), dynamic programming (DP) for face tracking in videos (Chapter 6).

2. Proposing a new method to estimate the optimal weighting parameter for fusion of scores in audio-visual person identification. This method first estimates the correct identification rate from score distributions, then directly maximises the estimated correct identification rate. Experiments indicate that the proposed method is superior to three other methods tested in reducing the bias and variance of the estimation (Chapter 4).

3. Theoretically proving the 'No Panacea Theorem', which states that if the combination function is continuous and diverse, there exists a situation in which the combination algorithm will give very bad performance. Thus, there is no optimal combination algorithm which is suitable in all situations (Chapter 5).

A minor contribution includes:

1. Implementing the text-independent speaker identification system for use in forensic voice recognition (Chapter 2).

1.2 Thesis Outline

Chapter 2: In this Chapter, we will introduce how to build audio biometric classifiers, which are also called speaker recognition classifiers. We will shortly review methods for speaker recognition, then discuss how to build a text-independent speaker identification system based on the Gaussian mixture model (GMM). Finally, we discuss the issue of applying the speaker identification system to forensic voice recognition.

Chapter 3: In this chapter, we will discuss methods of building visual biometric classifiers, which are also called face recognition classifier. We will review methods which are commonly used in automatic face detection and identification, respectively. We will also implement algorithms for face detection and face identification, and then describe their performance on the XM2VTS database.

Chapter 4: This chapter will outline our methods to combine the audio and visual classifiers. The problem of late integration of classifiers is firstly introduced. Then several frequently-used combination schemes are discussed. Finally, a new method for accurately choosing the optimal weighting parameter(s) for classifier combination is proposed. The proposed method is compared with three other well-established approaches. Using bootstrapping, we conclude that the proposed one can both reduce the bias and variance, thus achieving a better estimation for the optimal weighting parameter.

Chapter 5: We will prove the 'No Panacea Theorem' (NPT), which states that if the combination function is continuous and diverse, there exists a situation in which the combination algorithm gives very bad performance. Thus, there is no optimal combination algorithm which is suitable in all situations.

Chapter 6: In this chapter, we firstly develop an algorithm for tracking face images in video files, then use these face images to build a visual classifier. We also build an audio classifier based on techniques which are discussed in Chapter 2. The proposed method in Chapter 4 is used to combine the audio and visual classifiers. We apply this algorithm to the whole XM2VTS database, which consists of 295 subjects. Experiments indicate that the proposed audio-visual combination scheme can achieve good identification results on this database.

Chapter 7: This chapter concludes the thesis and outlines several directions for future research work, which include (1) developing combination algorithms which are robust to noise and unpredictable situations; (2) combining visual features with the audio-visual classifier; (3) research work on face recognition; (4) generalising the method of finding optimal weighting parameter to the person verification cases; (5) theoretical study on multiple classifier combination; (6) building a real-time audio-visual person recognition system.

Chapter 2

Speaker Identification System

This chapter will address the problem of building a person identification classifier based on audio signals. This research area is also called speaker identification. Speaker identification is the process of automatically deciding the identity of a person by extracting speaker specific information from his or her speech signals. The methods of speaker identification can be divided into two categories – **text-dependent** and **text-independent**. The former requires the speaker to provide utterances of key words or sentences that are the same text for both training and identification, whereas the latter does not rely on a specific text being spoken. The text-dependent methods are usually based on template matching techniques in which the time duration of an input speech is aligned with the template speech, and the similarity between them is accumulated from the beginning to the end of the speech (Furui *et al.*, 1972; Zheng and Yuan, 1988; Naik *et al.*, 1989; Rosenberg *et al.*, 1991). However, the text-independent methods are concerned with extracting speaker-dependent acoustic properties of speech signals, and use these properties for identification (Poritz, 1982; Tishby, 1991; Savic and Gupta, 1990; Reynolds and Rose, 1995).

This chapter will shortly review methods for both text-dependent and independent speaker recognition, then discuss on how to build a text-independent speaker identification system based on the Gaussian mixture model (GMM). Refer to Furui *et al.* (1972), Doddington (1985), O'Shaugnessy (1986) and Campbell (1997) for detailed reviews on speaker recognition. Finally, we discuss the issue of implementing the speaker identification system to forensic voice recognition.

2.1 Feature Extraction

The first step of speaker identification is to convert the input speech signals to a sequence of acoustic feature vectors. Ideally, the method of such a conversion should preserve all the acoustic characteristics that are distinctive for each speaker, while not being sensitive to

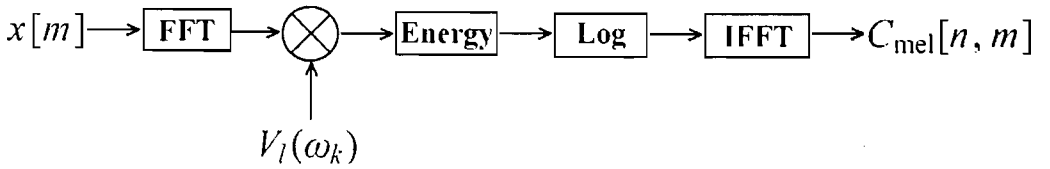


FIGURE 2.1: The process of computing MFCCs. The speech signal is firstly Fourier transformed into the frequency domain, then multiplied by a series of mel-scale filters. The obtained signal is then processed by taking its energy, then taking the log operation, finally inverse Fourier transform is used to obtain the MFCCs. This figure is regenerated from Quatieri (2001).

acoustic variations that are irrelevant to the identification process. Voiers (1964) divides speaker-related features into **high-level features** and **low-level features**. High level features include 'clarity', 'roughness', 'magnitude', and 'animation'. Other high-level features are prosody, pitch intonation, articulation rate, and dialect, etc. He found that such high-level characteristics are perceptual cues in determining speaker identifiability. On the other hand, these features are difficult to extract by computers. In contrast, low-level features, being of an acoustic nature, are more measurable. They can be extracted and measured with comparatively simple methods, and good performance is obtained by using low-level features both in speech and speaker recognition (Davis and Mermelstein, 1980; Furui, 1986; Reynolds and Rose, 1995).

One of the most frequently used set of low-level features is the mel-frequency cepstral coefficients (MFCC). MFCC features, which were introduced by Davis and Mermelstein (1980), exploit auditory principles as well as the decorrelating property of the cepstrum. In addition, They are amenable to compensation for convolutional channel distortion. As such, MFCC has proven to be one of the most successful feature representations in speech-related recognition tasks (Quatieri, 2001, Chap.14).

The procedure of MFCC computation is shown in Figure 2.1. The speech waveform is firstly windowed with the analysis window $\omega[n]$. Normally a Hanning window is used in this process. Refer to Oppenheim *et al.* (1999) for details of how to design analysis windows. Then the windowed speech signals are transformed into frequency domain by fast Fourier transform (FFT). These two steps can be written as the following:

$$X(n, \omega_k) = \sum_{m=-\frac{N}{2}}^{\frac{N}{2}} x[m]\omega[n-m]\exp^{-j\omega_k m}$$

where $\omega_k = \frac{2\pi}{N}k$ with N the FFT length. The magnitude of $X(n, \omega_k)$ is then weighted by a series of filters whose central frequencies and bandwidths roughly match those of the auditory critical band filters (Zwicker *et al.*, 1957). These filters follow the mel-scale whereby band edges and centre frequencies of the filters are linear for low frequency ($<$ 1000 Hz) and logarithmically increase with increasing frequency. We thus call the collection of these filters a mel-scale filter bank. An example of a mel-scale filter bank used by Davis and Mermelstein (1980) is illustrated

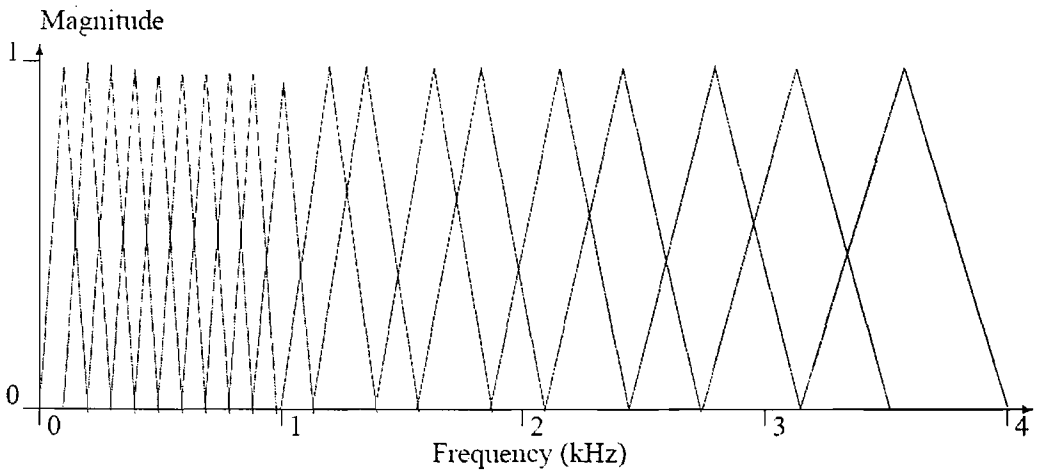


FIGURE 2.2: Triangular mel-scale filter bank used by Davis and Mermelstein (1980) in determining spectral log-energy features for speech recognition.

in Figure 2.2. This filter bank is a rough approximation to actual auditory critical-band filters covering a 4000 Hz range.

The next step in determining the mel-frequency cepstral coefficients is to compute the energy in the FFT coefficients weighted by each mel-scale filter. Denote the frequency response of the lth mel-scale filter as $V_l(\omega)$. The resulting energies are given for each speech frame at time n and for the lth mel-scale filter as:

$$E_{\text{mel}}(n, l) = \frac{1}{A_l} \sum_{k=L_l}^{U_l} |V_l(\omega_k) X(n, \omega_k)|^2$$

where L_l and U_l denote the lower and upper frequency indices over which each filter is nonzero and where

$$A_l = \sum_{k=L_l}^{U_l} |V_l(\omega_k)|^2$$

which normalises the filter according to their varying bandwidths so as to give equal energy for a flat input spectrum.

The mel-frequency cepstral coefficients are the inverse Fourier transformation (IFFT) of $\log\{E_{\text{mel}}(n, l)\}$, which is computed as:

$$C_{\text{mel}}[n, m] = \frac{1}{R} \sum_{l=0}^{R-1} \log\{E_{\text{mel}}(n, l)\} e^{j \frac{2\pi}{R} lm} \tag{2.1}$$

where R is the number of filters. The reason for using the log function is that the production of speech signals can be regarded as an excitation signal passing through a vocal tract filter. It corresponds to a convolution process in the time domain, which is equivalent to multiplying the spectral magnitudes of the source and filter components. When the spectrum is represented

logarithmically, these components are additive, because the logarithm of a product is equal to the sum of the logarithms ($\log(A \times B) = \log(A) + \log(B)$). Thus, by using a log function followed by IFFT, the coefficients corresponding to the excitation signal and the vocal tract filter can become decorrelated. Decorrelated coefficients are often more amenable to probabilistic modelling than are correlated coefficients. Because of the even property of $\log\{E_{\text{mel}}(n, l)\}$, equation (2.1) can finally be written as

$$C_{\text{mel}}[n, m] = \frac{1}{R} \sum_{l=0}^{R-1} \log\{E_{\text{mel}}(n, l)\} \cos\left(\frac{2\pi}{R} lm\right)$$

We will use MFCC for speaker identification in Section 2.4.

2.2 Methods for Text-dependent Speaker Identification

This section will provide a short review on methods for text-dependent speaker recognition, and the next section will review methods for text-independent speaker recognition. For text-dependent speaker identification, the first step is to train reference templates for the required text. When there is an input speech utterance, it is first time-aligned with these reference templates and then a similarity measure is accumulated for the duration of the utterance. Because the text is known in advance, theoretically we can eliminate the information associated with the text in the speech signals, and only retain the information associated with speakers. Because of this, text-dependent methods can achieve good results in a relatively short speech utterance compared with text-independent methods.

A typical approach to text-dependent speaker identification is the dynamic time warping (DTW) algorithm (Furui, 1981; Rabiner and Juang, 1993). The key idea of DTW is to use some form of dynamic programming to align temporal patterns to account for differences in speaking rates across speakers as well as across repetitions of the word by the same speaker. In this approach, each utterance is represented by a sequence of feature vectors (such as the MFCCs discussed in Section 2.1), and the trial-to-trial timing variation of utterances of the same text is normalised by aligning the analysed feature vector sequence of a test utterance to the template feature vector sequence using a DTW algorithm. The overall distance between the test utterance and the template is used for identification decision. The methodology of DTW is well developed and provides good performance for speaker identification.

Another frequently used speaker identification algorithm is hidden Markov model (HMM). The basic theory of hidden Markov models was published in a series of classic papers by Baum and his colleagues (Baum and Petrie, 1966; Baum and Egon, 1967; Baum and Sell, 1968; Baum *et al.*, 1970; Baum, 1972). A HMM can efficiently model the statistical variation in spectral features. Therefore, HMM-based methods have achieved significantly better results than DTW-based methods (Naik *et al.*, 1989; Rosenberg *et al.*, 1991).

2.3 Methods for Text-independent Speaker Identification

Because the text-independent methods do not require a person to say the same words or sentences during training and testing, text-independent speaker recognition is a more difficult task than text-dependent recognition. Algorithms for text-independent speaker identification can be categorised as HMM-based and probability-estimation-based. We will review these two methods respectively.

In text-independent speaker identification, we can not generally predict the words and sentences used in identification. Therefore, the states of the HMM have to be built on the phoneme level. Poritz (1982) proposed using a five-state ergodic HMM (i.e., all possible transitions between states are allowed) to classify speech segments into one of the broad phonetic categories corresponding to the HMM states. Each of the five states represents strong voicing, silence, nasal/liquid, stop burst/post silence, and frication respectively, and is trained with the corresponding phonemes uttered by speakers. Tishby (1991) extended Poritz's work to the richer class of mixture autoregressive (AR) HMMs. In these models, the states are described as a linear combination (mixture) of AR sources. Another approach using HMMs for text-independent speaker identification is speech-recognition-based. Phonemes or phoneme-classes are explicitly recognised, and then each phoneme (-class) segment in the input speech is compared with HMMs corresponding to that phoneme (-class) (Savic and Gupta, 1990).

The probability-estimation-based method assumes that acoustic features produced by each speaker follow some kind of probability distribution. These probability distributions can be estimated from the training data, provided that the time duration of the training data is long enough to decrease variations and thus obtain a stable probability model for each speaker. The time duration of the testing data also needs to be long enough. The first and earliest approach of probability-estimation method is to estimate distributions of simple acoustic features, such as spectrum representations or pitch (Furui *et al.*, 1972; Markel *et al.*, 1977). Reynolds and Rose (1995) extended this method by using Gaussian mixture models (GMMs) to estimate the probability distribution of mel-frequency cepstral features and obtained good results. The GMM algorithm has become a standard for text-independent speaker recognition and many new algorithms are based on it (Reynolds *et al.*, 2000; Chao *et al.*, 2005; Aronowitz *et al.*, 2005).

The vector quantisation (VQ) method can be classified as another kind of probability estimation if we regard the distance from the input feature vectors to those in a VQ-codebook as the reciprocal of a probability measure. In this method, VQ codebooks, consisting of a small number of representative feature vectors, are used as an efficient means of characterising speaker-specific features (Li and Wrench, 1983; Matsui and Furui, 1991; Rosenberg and Soong, 1987).

Recently, methods in machine learning has also been introduced into the area of text-independent speaker recognition. The method of using support vector machines (SVM) for text-independent speaker recognition were firstly proposed by Fine *et al.* (2001) and Kharroubi *et al.* (2001), and received much attention in this area. A series of publications have discussed

the SVM method for text-independent speaker recognition (Campbell, 2002; Wan and Renals, 2005; Campbell *et al.*, 2006, 2007).

2.4 GMM-based Speaker Identification System

The use of GMM for text-independent speaker recognition was proposed by Reynolds and Rose (1995), and became a standard in the area of speaker recognition. In this section, a speaker identification system based on this method is built and tested by the XM2VTS database (Messer *et al.*, 1999).

2.4.1 Gaussian Mixtures

Following Reynolds and Rose (1995), we use cepstral coefficients derived from a mel-frequency filter bank to represent the features for identification. The speaker model is based on the Gaussian mixture model. A Gaussian mixture density is a weighted sum of M component densities:

$$p(\vec{x}|\lambda) = \sum_{i=1}^{M} p_i b_i(\vec{x}) \tag{2.2}$$

where $\vec{x}$ is a D-dimensional random vector, $b_i(\vec{x})$ is the component density of the ith mixture and p_i is the weight of the ith mixture. Each component density is a D-variate Gaussian function of the form:

$$b_i(\vec{x}) = \frac{1}{(2\pi)^{D/2}|\Sigma_i|^{1/2}} \exp\left\{-\frac{1}{2}(\vec{x} - \vec{\mu}_i)^T \Sigma_i^{-1}(\vec{x} - \vec{\mu}_i)\right\}$$

with mean vector $\vec{\mu}_i$ and covariance matrix Σ_i. The mixture weights satisfy the constraint that $P_i \geq 0$ and $\sum_{i=1}^{M} p_i = 1$. The complete Gaussian mixture density is parameterised by the mean vectors, covariance matrices and mixture weights from all component densities. These parameters are collectively represented as the 3-tuple:

$$\lambda = \{p_i, \vec{\mu}_i, \Sigma_i\} \quad i = 1, 2, \ldots, M$$

For speaker identification, each speaker is represented by a GMM and is referred to by his/her model λ.

The GMM can have several different forms depending on the choice of covariance matrices. The model can have one covariance matrix per Gaussian component (nodal covariance), one covariance matrix for all Gaussian components in a speaker model (grand covariance), or a

single covariance matrix shared by all speaker models (global covariance). Here, we use nodal and diagonal covariance matrices for speaker models. Thus, we can write the speaker model as the following:

$$\lambda = \{p_i, \vec{\mu}_i, \sigma_i\} \quad i = 1, 2, \ldots, M$$

where σ_i represents the variance of the ith component.

2.4.2 Maximum Likelihood Parameter Estimation

Given training speech from a speaker, the goal of speaker model training is to estimate the parameters of the GMM, λ, which in some sense best matches the distribution of the training feature vectors. There are several techniques available for estimating the parameters of a GMM (McLachlan, 1988). By far the most popular and well-established method is maximum likelihood (ML) estimation.

The aim of ML estimation is to find the model parameters which maximise the likelihood of the GMM, given the training data. For a training data set X which contains F frames, $X = \{\vec{x}_1, \vec{x}_2, ..., \vec{x}_F\}$, the GMM likelihood can be written as:

$$p(X|\lambda) = \prod_{f=1}^{F} p(\vec{x}_f|\lambda) \tag{2.3}$$

Unfortunately, this expression is a nonlinear function of the parameters λ and direct maximisation is not possible. However, ML parameter estimations can be obtained iteratively using a special case of the EM (expectation-maximisation) algorithm (Dempster *et al.*, 1977).

The basic idea of the EM algorithm is, beginning with an initial model λ, to estimate a new model $\overline{\lambda}$, such that $p(X|\lambda) \geq p(X|\overline{\lambda})$. The new model then becomes the initial model for the next iteration and the process is repeated until some convergence threshold is reached.

On each EM iteration, the following reestimation formulas are used which guarantee a monotonic increase in the model's likelihood value:

Mixture Weights:

$$\overline{p}_i = \frac{1}{F} \sum_{f=1}^{F} p(i|\vec{x}_f, \lambda)$$

Means:

$$\overline{\vec{\mu}}_i = \frac{\sum_{f=1}^{F} p(i|\vec{x}_f, \lambda)\vec{x}_f}{\sum_{f=1}^{F} p(i|\vec{x}_f, \lambda)}$$

Variances:

$$\overline{\sigma}_i^2 = \frac{\sum_{f=1}^{F} p(i|\vec{x}_f, \lambda) \left|\vec{x}_f\right|^2}{\sum_{f=1}^{F} p(i|\vec{x}_f, \lambda)} - \left|\vec{\overline{\mu}}_i\right|^2$$

The *a posteriori* probability for the ith mixture is given by

$$p(i|\vec{x}_f, \lambda) = \frac{p_i b_i(\vec{x}_t)}{\sum_{k=1}^{M} p_k b_k(\vec{x}_t)}$$

Thus, the new model $\overline{\lambda} = \{\overline{p}_i, \vec{\overline{\mu}}_i, \overline{\sigma}_i\}$ is obtained from the old model $\lambda = \{p_i, \vec{\mu}_i, \sigma_i\}$ by the above formulas. After several iterations, the model parameters will converge to a local maximum of equation (2.3).

2.4.3 Decision Rule

Suppose there are K speakers to be identified. Then λ_k, $k = 1, 2, \ldots, K$ is the model corresponding to the kth enrolled speaker. The goal of speaker identification is to find the one among these K models that best matches the test data represented by a sequence of F frames, $X = \{\vec{x_1}, \vec{x_2}, \ldots, \vec{x_F}\}$. In making the decision, we use the following frame-base weighted likelihood distance measure, d_k, which refers to the distance from the test data to the kth speaker model:

$$d_k = \frac{1}{F} \sum_{f=1}^{F} \log p(\vec{x_f}|\lambda_k) \tag{2.4}$$

in which $p(\vec{x_f}|\lambda_k)$ is given in equation (2.2). The normalisation by F is necessary as each token will, in general, have a different length and, therefore, a different number of frames.

Suppose the task of a classifier is to assign an input sequence X to one of K classes $\omega_1, \omega_2, \ldots, \omega_K$. We can then identify speaker s according to the rule:

$$\text{decide } X \in \omega_s \text{ if } s = \arg\max_i d_i$$

2.4.4 Silence Removal

One step of our experiments is to remove the inter- and intra-word silence of speech both in training and testing. The idea of removing inter- and intra-word silence was based on the insight (obvious in retrospect) that there is unlikely to be much if any speaker-specific information contained within non-speech intervals of the signal.

A simple silence-removal technique based on combining information on sound intensity and zero crossing rate is used. This is a version of the Rabiner-Sambur algorithm (Rabiner and

Sambur, 1975), originally designed for detection of the endpoints of isolated words but modified here for the removal of word-internal silence. This algorithm sets two sound intensity thresholds: an upper threshold I_1 and a lower threshold I_2 ($I_1 > I_2$). It also sets a zero-crossing rate threshold, Z. First, the algorithm marks the data points whose intensity is higher than the upper threshold I_1 as 'speech points'. Then it extends the boundary of the speech points to points which have higher intensity than the lower intensity threshold I_2. After this, the algorithm further extends the boundary of the speech points to those whose zero crossing rates exceed Z. The rationale is that low-intensity parts of the signal might be legitimate speech, but such low-intensity regions will correspond to noise-like excitation in the vocal tract spectrally shaped by a high-pass characteristic to have a relatively high zero-crossing rate (e.g., the fricative sounds, /f/, /v/, /s/ and /z/). All the other data points, not marked as speech points, are removed as 'silence'. By appropriately setting these three thresholds, the algorithm can successfully remove silence most of the time.

In Table 2.2, we have compared the correct identification rate in situations when silence was removed and not removed. From our experiments, it is indicated that our silence-removal algorithm can always achieve better identification rate.

2.4.5 Introduction to the XM2VTS database

In this thesis, we will use the XM2VTS database to test our algorithms on audio-visual person identification. This database, intended for research into multi-modal person recognition, is obtainable from the Centre for Vision, Speech and Signal Processing at the University of Surrey, UK. It contains high-quality colour images, 32 kHz 16-bit sound files and video files of 295 subjects. These files are recorded in 4 sessions, over a period of 4 months. For each session, the subjects speak 6 sentences, which are numbered as Sentence 1 to 6. The content of each sentence is as follows.

1. Sentence 1 and 4: Counting from zero to ten. 'Zero, one, two, three, four, five, six, seven, eight, nine, ten.'
2. Sentence 2 and 5: Counting with a different order. 'Five, zero, six, nine, two, eight, one, three, seven, four.'
3. Sentence 3 and 6: Speaking A sentence. 'John took father's green shoe bench out.'

Each file has a unique name, with the form of '(Subject number)_(Session number)_(Sentence number)'. Each subject is assigned to a unique subject number. For example, the first subject was assigned a number '000', and the last one assigned '371'. The other subjects were assigned numbers between '000' and '371'. The session number is from 1 to 4, corresponding to the 4 recording sessions, and the sentence number is as defined above. For example, the audio file which contains Sentence 1 spoken by the first subject, during the first session is labelled as

Disk Number	Content
1	Speech files of 75 subjects. There are 24 files for each person, which is recorded in 4 sessions (6 files for each session).
2	Speech files of another 73 subjects, also 24 files for each person.
3	Speech files of another 74 subjects, 24 files for each person.
4	Speech files of another 73 subjects, 24 files for each person.
5	Frontal face images for the 295 subjects. There are 8 images for each person, which is taken in 4 sessions (2 images for each session).
6	Video files which have both audio and visual information. One video file for each subject, which is recorded during the first session, when the subject speaks Sentence 3.
7	Video files which have both audio and visual information. One video file for each subject, which is recorded during the second session, when the subject speaks Sentence 3.
8	Video files which have both audio and visual information. One video file for each subject, which is recorded during the third session, when the subject speaks Sentence 3.
9	Video files which have both audio and visual information. One video file for each subject, which is recorded during the fourth session, when the subject speaks Sentence 3.

TABLE 2.1: The contents of the XM2VTS database.

'000_1_1.wav'. Correspondingly, '000_1_1.ppm' and '000_1_1.avi' refer to the image and video files which were taken when this sentence was spoken. In this thesis, video files refer to files which contain synchronised sound and moving pictures. Please refer to Appendix A for detailed introduction of the image and video files in the XM2VTS database.

We obtained an incomplete portion of the database, which consists of 9 disks. The content of each disk is listed in Table 2.1. The first four disks contain audio files; the fifth disk contains static images; and the last four disks contain video files which provide both audio and visual information. In this thesis, the XM2VTS database is used to test all the algorithms which we proposed for audio-visual person identification.

2.4.6 Experimental Results

We use the sound files contained on the first disk of the XM2VTS database to test the speaker identification system. It contains speech files of 75 subjects. Each speaker provides 24 speech files which were recorded during 4 sessions (6 files for each session).

For the silence removing algorithm, we set the upper sound intensity threshold I_1 to be 0.5 times the average sound intensity of the speech file, and the lower intensity threshold I_2 to be 0.2 times this average intensity value. The zero-crossing rate threshold Z was set to the average zero crossing rate of the speech file. Our experimental results suggest that, in most cases, these settings are reasonable and correctly remove word-internal silence while retaining the speech itself.

After the silence is removed, we use mel-frequency cepstral coefficients as features. The magnitude spectrum from a 20 ms short-time segment of speech is pre-emphasised and processed by a mel-scale filter bank, then the log-energy filter outputs are cosine transformed to produce the cepstral coefficients. We use the first 20 coefficients excluding the zeroth coefficient, plus the first 20 delta coefficients as the feature set. This process occurs every 10 ms, producing 100 features per second. Gaussian mixtures consisting of 64 component densities are used in these experiments. These experimental settings are proposed by Reynolds and Rose (1995). In their paper, they compared the system's performance by varying the number of Gaussian mixtures from 8 to 64. They find that when enough training speech data are provided, the performance increases dramatically with the number of mixtures when it is below 30, and the performance becomes stable after that.

We iteratively change the training sessions and testing sessions. First, we use one session for training, and the other three for testing; then we use two sessions for training and the other two for testing; finally we use three sessions for training and the other for testing. The identification results are shown in Table 2.2. In order to illustrate the effect of the silence-removing algorithm in Section 2.4.4, we have listed the identification rates with and without silence removed, respectively.

We can see that the identification rates with silence removed are always better than those without silence removed, which indicates our silence-removal algorithm can always improve the speaker identification performance. Since the silence-removing process achieves better identification results, we will always use this technique in the remaining experiments and not mention it explicitly.

For the results with silence removed, the identification rate is around 84% if only one session is used for training (the first 4 lines of Table 2.2). When two sessions are used for training, the identification rate increases to around 96% (the next 6 lines). When three sessions are used for training, the identification rate further increases to around 98% (the last 4 lines). It seems that the identification rate is very high, which gives us an illusion that voice is a nearly perfect

Training Session(s)	Testing Session(s)	Identification Rate Silence not Removed (%)	Identification Rate Silence Removed (%)
1	2,3,4	75.11	83.19
2	1,3,4	74.81	83.85
3	1,2,4	71.33	84.74
4	1,2,3	67.48	87.33
1,2	3,4	89.11	95.56
1,3	2,4	89.00	96.44
1,4	2,3	89.78	97.22
2,3	1,4	89.67	96.44
2,4	1,3	85.44	94.89
3,4	1,2	84.00	95.33
1,2,3	4	96.22	98.00
1,2,4	3	92.22	98.22
1,3,4	2	93.33	98.89
2,3,4	1	87.78	96.67

TABLE 2.2: Speaker identification rate of the GMM-based method. We have listed results both for silence removed and not removed. For the results with silence removed, the identification rate is around 84% if only one session is used for training. When two sessions are used for training, the identification rate increases to around 96%). When three sessions are used for training, the identification rate further increases to around 98%.

biometric for person identification. However, the identification rate declines dramatically in noisy environment, as can be seen in Section 2.5.

The performance of the classifier can be further illustrated in the ranking curve (Figure 2.3). In Figure 2.3, the X axis refers to the ranks of the correct classes, and the Y axis illustrates the cumulative identification rates at the corresponding ranks. For example, when $X = 1$, the corresponding Y is the identification rate which was listed in Table 2.2; when $X = 2$, the Y axis shows the fraction of the testing files which their correct classes generate the highest scores or second highest scores. We show 3 ranking curves out of the 14 classifiers in Table 2.2, which are, the classifiers in row 1, 5 and 11. The first of these 3 classifiers uses Session 1 for training, and Sessions 2, 3 and 4 for testing; the second uses Sessions 1 and 2 for training, and Sessions 3 and 4 for testing; and the third uses Sessions 1, 2 and 3 for training, and Session 4 for testing. We can see that as the training sessions increased, the ranking curves are becoming higher and reaches 1 more quickly. This indicates that the speaker identification classifiers becomes better when more training data are added. Another observation is that great performance improvement has been occurred when two sessions are used for training compared with just one session for training, however, the performance does not increase too much when three sessions are used for training instead of two.

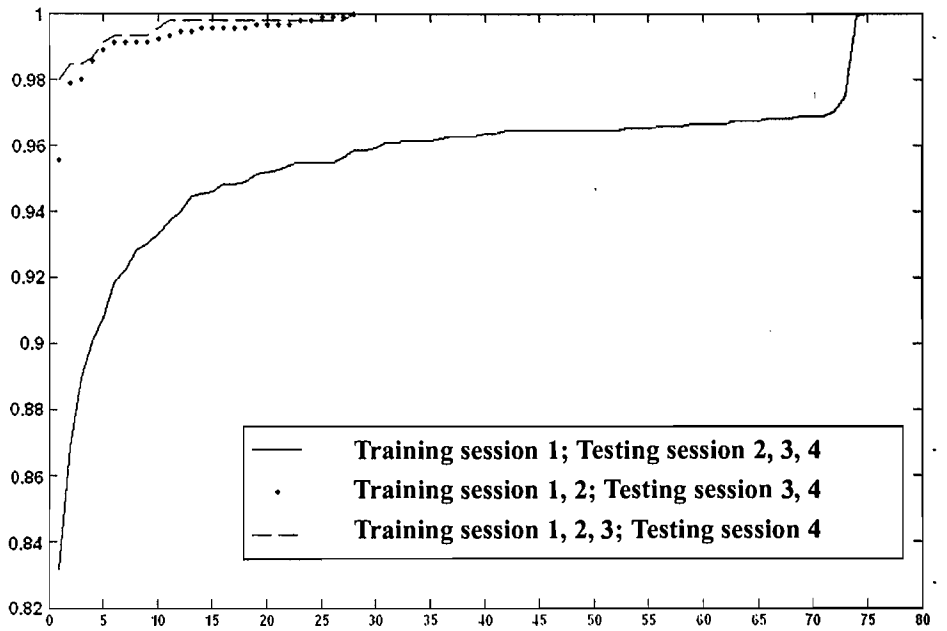


FIGURE 2.3: Ranking curves of three classifiers in Table 2.2, which are, the classifiers in row 1, 5 and 11. The first of these 3 classifiers uses Session 1 for training, and Sessions 2, 3 and 4 for testing; the second uses Sessions 1 and 2 for training, and Sessions 3 and 4 for testing; and the third uses Sessions 1, 2 and 3 for training, and Session 4 for testing. The three ranking curves are represented by the solid line, the dotted line and the dash line, respectively.

2.5 Forensic Voice Recognition

The speaker identification system built by MFCC and GMM achieves high identification rate as shown in Section 2.4.6. In this section, we will use this system to solve practical problems. In January 2006, Southampton police office has provided us a case for which our text-independent speaker identification system might be applied. We obtained some results by using our system. Because the voices contained in the audio tapes are contaminated by noise, the system didn't generate too much information that could assist the police to catch the criminal. However, we think this unsuccessful experience is valuable for pointing out the limitations of current speaker recognition technology and the intrinsic problems for forensic voice recognition. We will describe the case and our experiment in this section.

2.5.1 The Case

Two robbers came to a house at night. They chopped the door with an axe. The host and hostess of the house were awakened by the sound of chopping and came out. They quarrelled with the robbers for about two minutes, then the robbers ran away. Their conversations, together with other sounds, such as the sound of door chopping, dog barking, background noise, screaming of the host's daughter, were recorded by a sound recorder outside of the house. We refer to this recording as Recording 1. In Recording 1, the two robbers' voices only occupy a short

Recording 1	
Sound Source	**Duration (Seconds)**
Robber 1	10
Robber 2	3
The Hostess	15
Scream of the host's daughter	1
Sound of chopping the door	27
Dog Barking	10
Background Noise	17
Recording 2	
Sound Source	**Duration (Seconds)**
Suspect	894
Police	337
Woman Police	13

TABLE 2.3: Sound sources and durations in the two recordings.

time period. One robber, whom we refer to as Robber 1, shouted 'Come out' for about 10 seconds in order to frighten the host and hostess. Another robber, whom we refer as Robber 2, said, 'John, come on', which lasted only 1 second, but it provided important information that Robber 1's name is John. Several days later, the police arrested a suspect whose name is also John. He is alleged to be Robber 1. The whole interrogation is also recorded by a sound recorder, which consists of the voices of a policeman, a policewoman and the suspect. We refer to the interrogation recording as Recording 2.

Our aim is to provide evidence that whether the voice of Robber 1 in the first recording and the voice of the suspect in the second recording are produced by the same person. We investigate this problem by using the GMM-based speaker identification system.

2.5.2 Experimental Settings and Results

Admittedly, it is difficult to decide whether the voices in these two recordings are generated by the same speaker, especially when the voice in one of the two recordings is very short. Several researchers studied this problem with the background of forensic acoustics. Most of them obtain negative results (Boë, 2000; Broeders, 2001; Bonastre *et al.*, 2003). As concluded by Boë (2000) and Bonastre *et al.* (2003), currently it is not possible to completely determine whether the similarity between two recordings is due to the speaker or to other factors.

Here we try to tackle this problem by using our GMM-based speaker identification system. First, we separate the sound in the two recordings into small sound segments. Then we concatenate the sound segments produced by the same sound source together. Details of each sound source and its duration are shown in Table 2.3.

The settings of the experiment is as follows. We use the voices of the suspect and the police in

	Suspect Model	**Police Model**
Voice of Robber 1	-78.4192	-84.7210
Voice of Robber 2	-64.3231	-70.0571
Voice of the Hostess	-76.4153	-83.1419
Scream of the host's daughter	-76.1317	-84.7993
Sound of chopping the door	-62.2292	-65.3633
Dog Barking	-63.4083	-67.2631
Background Noise	-61.0813	-64.8564

TABLE 2.4: Scores of sound segments in Recording 1, which are calculated by equation 2.4.

Recording 2 to train the GMM classifier. Because the voices are long enough (894 seconds for the suspect and 337 seconds for the police), we can estimate the trained classifier provides good speaker models. Then we use the sounds in Recording 1 to test the classifier, to see whether each sound is more like the suspect's voice or the police's voice in Recording 2. The testing scores are obtained in Table 2.4. The scores are negative because the obtained probability is converted by the log function, as shown in equation 2.4.

The GMM-based speaker identification classifier indicates that the voice of Robber 1 is more similar to the suspect model than the police model. However, the problem is far from being solved, because (1) the suspect model tends to generate greater scores than the police model for all sounds produced by these sound sources, which means that the classifier may be biased on the suspect model; (2) the background noise generates the greatest scores of all these sounds, which means the classifier is not reliable. A possible explanation of the results is that, both recordings are filled with similar kind of noise, and the voice of the suspect in Recording 2 is more like the noise, so that the suspect model always provides greater scores and the background noise in Recording 1 generates the greatest scores.

We have to say that the result is not very successful, and the problem remains unsolved. The difficulties of this problem lies in two parts. First, Robber 1's voice is very short in Recording 1 (10 seconds), so some factors which are not dependent on the speaker can not be averaged out. Second, the two recordings are recorded in different environment, so the results may not show the difference of speakers, but the difference of environment.

2.6 Summary

In this chapter, we discussed how to build an audio speaker identification system. Firstly, we shortly reviewed methods for both text-dependent and text-independent speaker identification. Then a text-independent speaker identification system was built based on MFCC features and GMM model. This system achieved more than 95% identification rate over the 295 subjects of the XM2VTS database if enough speech data is used to train the classifier.

We also discussed the possibility of using this classifier to forensic voice recognition. As indicated in Section 2.5, in this forensic case, the text-independent speaker identification method

is sensitive to background noise. Noise may severely alter the probability distribution of the speech signal, so speaker identification methods based on estimating probability distributions perform not very well when the speech signal is contaminated by noise. More research work needs to be carried out for speaker identification in noisy environment.

Chapter 3

Face Identification System

Automatic face identification has a large number of applications, including security access, law enforcement, internet communication, and computer entertainment. Research in face identification was firstly carried out in the 60's. Although the performance of face identification systems has improved significantly since the first automatic face recognition system was developed by Kanade (1973), there are many unsolved problems remaining in this area. The face identification process can be divided into two steps. First, the face region in an image should be automatically detected; Second, the detected face region should be processed by an identification system and the identification result obtained. We call the first step **face detection** and the second **face identification**. In this chapter, we will review methods which are commonly used in face detection and face identification, respectively. We will also implement algorithms for face detection and face identification, and then discuss their performance on the XM2VTS database. Readers who want to find more about these topics are encouraged to read the following literature surveys: Chellappa *et al.* (1995), Zhao *et al.* (2003), and Li and Jain (2004).

3.1 Methods for Face Detection

In realistic application scenarios, a face could occur in a complex background and in many different positions. A face detection system which can accurately localise and extract face regions is the prerequisite for good face identification performance. Because of this, detecting faces is one of the key first steps in face recognition (Chellappa *et al.*, 1995).

Face detection methods can be effectively organised into two broad categories, distinguished by their different approaches to utilising face knowledge (Hjelmas, 2001). The techniques in the first category make explicit use of two kinds of information. The first is the shape information of faces and face features, such as edges and colour information of eyes, mouth and nose. The second information source is the geometry relationships of facial landmarks and features. Since features are the main ingredients, these techniques are termed as the **feature-based approach**. Another approach, which is called **image-based approach**, is to regard the

face detection problem as general pattern recognition problem. Image-based representations of faces, for example in 2D intensity arrays, are directly classified into a face group using training algorithms without feature derivation and analysis. Unlike the feature-based approach, these relatively new techniques incorporate face knowledge implicitly into the system through mapping and training schemes (Valentin *et al.*, 1994). The reader may refer to Hjelmas (2001), Yang *et al.* (2002), and Li and Jain (2004) for reviews of face detection.

3.1.1 Feature-Based Approach for Face Detection

As discussed above, there are two kinds of information which can be used for the feature-based approach. The first is the shape information of facial features, such as eyes, mouth and nose. The second is the geometry relationships of facial features, such as the relative widths of features, the distances among face features, etc. Normally speaking, solely using one of these two kinds of information can not guarantee good performance. A robust detection is based on incorporating these two kinds of information together. Based on how to use these two kinds of information, the feature-based approach can be further divided into top-down methods and bottom-up methods (Yang *et al.*, 2002).

For the top-down methods, face candidates are first found by the geometry information of facial features, then detailed information for facial features is incorporated to accept or reject these candidates. For example, a frontal face often appears in an image with two eyes that are symmetric to each other, a nose and a mouth on the symmetrical line of these two eyes. If the image is under-sampled into low resolution, the face image will shrink to two line segments which represent eyes, with two line segments on the symmetrical line of the two eyes which represent nose and mouth respectively. The face candidates can be obtained by finding the clusters of these line segments. In Wang and Yuan (2001), wavelet decomposition is first performed on the images. Then feature candidates are obtained by properly thresholding the intensity values of the wavelet image. Yang and Huang (1994) used a hierarchical knowledge-based method to detect faces. Their system consists of three levels of rules. The rules at a higher level are general descriptions of what a face looks like while the rules at lower levels rely on details of facial features.

In contrast to the top-down approach, researchers have been trying to find invariant features of faces for detection. Many methods have been proposed to detect facial features and then, based on the extracted features, a statistical model is built to describe their geometrical relationship and to verify the existence of a face. Yuille *et al.* (1989) used deformable templates to find the position of eyes and mouths independently. The face region is automatically obtained after eyes and mouth are accurately located. Compared with other features, the eyes are easier to extract. To locate the positions of eyes accurately, Lam and Yan (1996) used active contours to extract eyes. The position of the eyes is coarsely obtained by edge detection, then an active contour, or snake (Kass *et al.*, 1998; Gunn and Nixon, 1997), is first initialised at the proximity around the eyes. It then locks onto nearby edges and subsequently assumes the shape of eyes. Some

other bottom-up methods do not accurately detect facial features. Instead, these methods first detect candidates for facial features from information provided by low-level image operators (e.g., edges and colours), then the information about the geometrical relationship of facial features is applied to select facial features from candidates. Govindaraju (1996) tried to find facial feature candidates by labelling edges as the left side, hairline or right side of a frontal face and matched these edges against a face model. The labelled components are combined to form possible face locations based on a cost function. Yow and Cipolla (1996) modelled a face as a plane with six oriented facial features (two eyebrows, two eyes, one nose and one mouth). Each facial feature is modelled as a pair of oriented edges. The feature selection process starts with candidate points, followed by edge detection and linking, and tested by a statistical model which models the geometrical relationship of facial features. Hsu *et al.* (2002) used colour information for face detection. A skin colour model is first implemented to extract face candidates. Then candidates of eyes and mouths are detected by the individual colour models for eyes and mouths. Finally an *eye-mouth triangle* is formed for all possible combinations of the two eye candidates and one mouth candidate. Each eye-mouth triangle is verified by checking (1) lumina variations and average gradient orientations of eye and mouth blobs; (2) geometry and orientation of the triangle; and (3) the presence of a face boundary around the triangle. A score is computed for each verified eye-mouth triangle and all triangles that exceed a threshold are retained as a detected face. Another example of the bottom-up approach is the active shape models (ASM), which was firstly proposed by Cootes *et al.* (1995). Cootes *et al.* extends their work and introduced another powerful approach to deformable template models, namely, the active appearance models (AAM), which utilised a combined PCA of the landmarks and pixel values inside the object for object detection. Several publications discussed the issue of using ASM and AAM in face detection and modelling (Edwards *et al.*, 1998; Ahlberg, 2001; Wan *et al.*, 2005; Sukno *et al.*, 2007).

3.1.2 Image-Based Approach for Face Detection

Face detection by explicit modelling of facial features has been troubled by the unpredictability of face appearance and environmental conditions. Although some of the recent feature-based attempts have improved the ability to cope with the unpredictability, most are still limited in noisy and complex environments. There is a need for techniques that can perform in more hostile scenarios such as detecting multiple faces with different poses in clutter-intensive backgrounds. This requirement has inspired the research area of image-based face detection. By formulating the problem as one of learning to recognise a face pattern from examples, the specific application of face knowledge is avoided, or more accurately speaking, the requirement of face knowledge is implicitly 'hidden' into the data training process. This eliminates the potential of modelling error due to incomplete or inaccurate face knowledge. The basic approach in recognising a face pattern is via a training procedure which classifies examples into face and non-face prototype classes. Comparison between these classes and a 2-D intensity array extracted from an input image allows the decision of face existence to be made.

An early approach was to represent the face images as a subspace of the overall image space. Sirovich and Kirby (1987) developed a technique using principal component analysis (PCA) to represent human faces. Turk and Pentland (1991) later developed this technique for face detection and recognition. Their method exploits the distinct nature of the weights of eigenfaces in individual face representation. Since the face reconstruction by its principal components is an approximation, a residual error is defined in the algorithm as a preliminary measure of 'faceness'. This residual error, which they termed 'distance-from-face-space' (DFFS), gives an indication of face existence, and can be used for face detection. Section 3.3.1 will discuss PCA and subspace methods in more detail for face identification.

Neural networks have become a popular technique for pattern recognition problems, including face detection. The first neural network approach which reported results on a large, difficult database was by Rowley *et al.* (1998). Their system incorporates face knowledge in a connected neural network which is designed to look at windows of 20 × 20 pixels (thus 400 input units). There is one hidden layer with 26 units, where 4 units look at 10 × 10 pixel subregions, 16 look at 5 × 5 subregions, and 6 look at 20 × 5 pixels overlapping horizontal stripes. The input window is pre-processed through lighting correction and histogram equalisation. A problem that arises with window scanning techniques is overlapping detections. Rowley *et al.* (1998) deal with this problem through two heuristics:

1. if the number of detections in a small neighbourhood surrounding the current location is above a certain threshold, a face is present at this location;
2. when a region is classified as a face according to thresholding, then overlapping detections are likely to be false positives and are thus rejected.

To improve performance further, they train multiple neural networks and combine the output with an arbitration strategy (AND, OR, voting, or a separate arbitration neural network).

Other statistical learning approaches are also applied to the problem of face detection. These include distribution-based classifier (Sung and Poggio, 1998); support vector machines (Osuna *et al.*, 1997); naive Bayes classifier (Schneiderman and Kanade, 1998); hidden Markov model (Rajagopalan *et al.*, 1998); and information theoretical approach (Lew, 1996; Colmenarez and Huang, 1997). Recently, Haar-like features and AdaBoost (Freund and Schapire, 1997) learning-based face detection methods have achieved good performance in face detection (Viola and Jones, 2001; Lienhart *et al.*, 2002; Viola and Jones, 2002; Li and Zhang, 2004). Li and Jain (2004) recommended AdaBoost methods because the performance of these methods is comparable to the neural network method of Rowley *et al.* (1998), but they are several times faster. In Section 3.2.3, we will discuss how to build a face detection system based on the AdaBoost algorithm.

Although face detection technology is now sufficiently mature to meet the minimum requirements of many practical applications, its performance is still far behind human performance. More research work needs to be carried out to fill this gap.

3.2 Face Detection System

In this section, two face detection systems are presented. One is feature-based and the other is image-based. Both systems are tested on the XM2VTS database. The feature-based method is very similar to methods used by Yow and Cipolla (1996) and Jeng *et al.* (1998). Images are first input into a face colour detection module, then the skin-tone pixels are detected using an elliptical skin model in the transformed space. The feature-selection process starts with interest skin-tone pixels, followed by edge detection and linking, and tested by a statistical model which models the geometrical relationship of facial features.

The image-based face detection system is implemented by using Haar-like features and the AdaBoost method which was proposed by Viola and Jones (2002). Recent research shows that this method can achieve robust performance under various situations.

3.2.1 The Feature-Based Detection Procedure

The first step of the feature-based face detection system is to extract skin colour points on images. Human skin has its own colour distribution that differs from most nonface objects. The skin colour distribution can be used to obtain candidate regions of faces. A skin colour likelihood model can be derived from skin samples. This may be done in different colour spaces, such as the HSV space, RGB space, or YC_rC_b space (Martinkauppi, 2002; Zarit *et al.*, 1999; Hsu *et al.*, 2002). In our application, we use the method proposed by Hsu *et al.* (2002). First, the RGB space is converted to YC_rC_b space. Then by linear transformation, the skin colour tends to gather into an ellipse. We regard colours which are fall into that ellipse as skin colours, and filter out others which are out of that ellipse. The following are the original face image and the corresponding image which is obtained by this model. Based on the skin colour model, we can divide disconnected skin colour pixels into one or several face regions. The red square of Figure 3.2 indicates the face region which is detected from Figure 3.1(b).

The obtained face image is then processed by a Sobel edge detector (Gonzalez and Woods, 2002, chap.3, pp.75–146). The edge image is thresholded with a value which minimises the within-group variance (Haralick and Shapiro, 1992, Vol.1, Chap.2, pp.13–58). Figure 3.3(a) and Figure 3.3(b) show the edge image generated by the Sobel edge detector and the edge image after the thresholding process.

It can be observed that the main facial features such as eyes, nose and mouth are retained by the remaining white pixels in Figure 3.3(b). The next step is to locate these features. First, the so-called *run-length local table method* is employed to group these white pixels into maximal connected blocks (Haralick and Shapiro, 1992). Figure 3.4(a) shows the blocks which are grouped by this algorithm (different colours represent different blocks). We can see that facial features such as eyes, nose and mouth are grouped as different blocks. However, some of these blocks are not facial features, such as the face contour (the block with red colour) and the tiny

(a)

(b)

FIGURE 3.1: Face detection using skin colour model. (a) The original face image. (b) The image after skin colour detection.

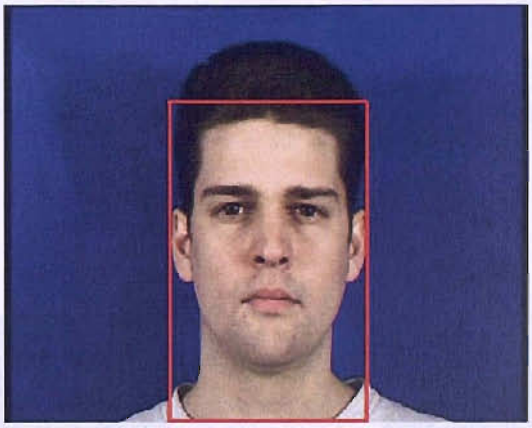


FIGURE 3.2: The detected face region, which is represented by the red square in this image.

blocks at the top of the image which are caused by the hair. The following procedure shows how to find facial features exactly from these blocks.

For each block B, its width w, height h, centre of mass $(\overline{x}, \overline{y})$ and orientation θ can be obtained as follows (Jain, 1989):

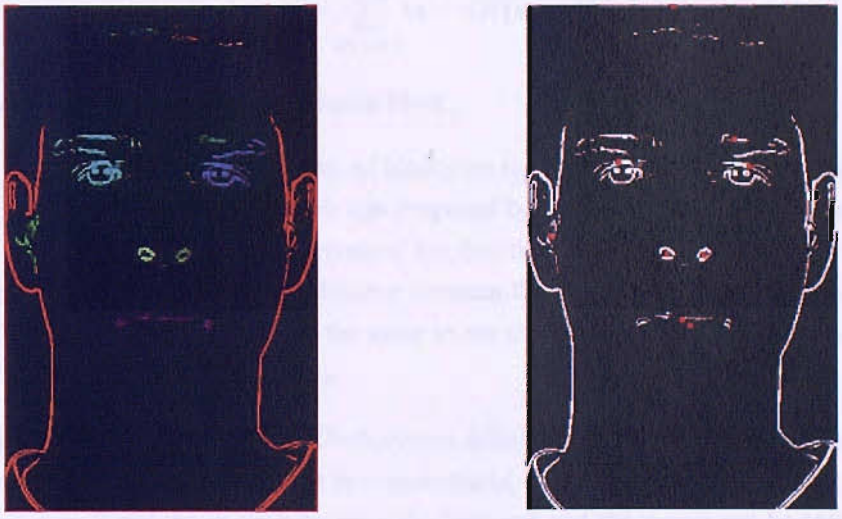
$$w = \max_{(x,y)\in B}(x) - \min_{(x,y)\in B}(x), \quad h = \max_{(x,y)\in B}(y) - \min_{(x,y)\in B}(y)$$

$$\overline{x} = \frac{1}{N}\sum_{(x,y)\in B} x, \quad \overline{y} = \frac{1}{N}\sum_{(x,y)\in B} y$$

$$\theta = \frac{1}{2}\tan^{-1}\left(\frac{2\mu_{1,1}}{\mu_{2,0} - \mu_{0,2}}\right) \tag{3.1}$$

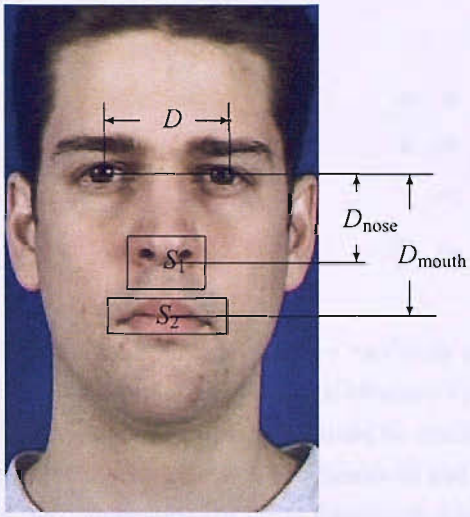
(a) (b)

FIGURE 3.3: Face image after edge detection and thresholding. (a) After skin colour detection, the face image is sent to a Sobel edge detector. (b) The obtained edge image is thresholded with a value which minimises the within-group variance.



(a) (b)

FIGURE 3.4: Blocks and block centres. (a) The blocks of the face image. Different blocks are represented with different colours. (b) Centres of blocks which are represented by the red points.



D: The distance between two eyes.

D_{nose}: The distance between the base line and the nose. In this model, we take $D_{nose} = 0.6D$.

D_{mouth}: The distance between the base line and the mouth. In this model, we take $D_{mouth} = D$.

S_1: The region for searching the nose point.

S_2: The region for searching the mouth point.

FIGURE 3.5: The geometrical facial model which was proposed by Jeng *et al.* (1998). This model incorporates the fact that the distance between the nose and eyes is approximately 0.6 times the distance between the two eyes, and the distance between the mouth and eyes is approximately the same to the distance between the two eyes. Using this information, it firstly find regions which are near the estimated nose and mouth positions, as indicated by S_1 and S_2 on the graph, then regards face features candidates falling into these regions as nose and mouth candidates. By using equation 3.5, 3.6 and 3.7, the final energy E is obtained for each combination of eye, nose and mouth candidate, and the combination which maximises the energy is selected as the final result.

Here $\mu_{p,q}$ is the (p, q)th central moment and can be calculated by

$$\mu_{p,q} = \sum_{(x,y)\in B} (x - \overline{x})^p (y - \overline{y})^q \tag{3.2}$$

Refer to Figure 3.4(b) for the centre of each block.

After the labelling and grouping process, all blocks are regarded as facial feature candidates. We use the geometrical facial model which was proposed by Jeng *et al.* (1998) for face detection. This is a very simple model. It incorporates the fact that the distance between the nose and eyes is approximately 0.6 times the distance between the two eyes, and the distance between the mouth and eyes is approximately the same to the distance between the two eyes. Such a relationship is illustrated in Figure 3.5).

The line passing through the centres of both eyes is called the *base line*. Let (x_1, y_1) and (x_2, y_2) be the centres of the left eye and right eye respectively. Then, the coefficients of the base line $ax + by + c = 0$ and the angle θ between the base line and the x-axis can be calculated as follows:

$$\begin{aligned} a &= y_2 - y_1 \\ b &= x_1 - x_2 \\ c &= x_2 y_1 - x_1 y_2 \\ \theta &= \tan^{-1}\left(-\frac{b}{a}\right) \end{aligned} \tag{3.3}$$

The matching process starts by randomly selecting two blocks as eyes. Their corresponding base line and the angle θ can be obtained by using equation (3.3). Because of the constraints of face orientation, the angle θ should be checked to see if it is within the range of $-45°$ to $45°$. If not, the current pair will be abandoned and another pair will be considered. In this step, most of the combinations of facial feature candidates will be rejected. For those surviving pairs, their orientation θ_1 and θ_2, as indicated by equation (3.1), and 'normalised widths' between the two eyes, l_1 and l_2 are calculated. The normalised widths of the two eyes are obtained by dividing the width between the two eyes by the distance between their centres. Suppose the widths of the two eye candidates are w_1 and w_2, respectively, their centres are (x_1, y_1) and (x_2, y_2). Then, the normalised widths of the two eyes are calculated as follows:

$$\begin{aligned} D &= \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2} \\ l_1 &= w_1/D, \; l_2 = w_2/D \end{aligned} \tag{3.4}$$

After obtaining l_1 and l_2, the following evaluation function is employed to evaluate whether the current block pair is the eyes or not:

$$E_{\text{eye}} = \exp\left\{-\left[(l_1 - l_2)^2 + (l_1 + l_2 - 2)^2 + (\theta_1 - \theta)^2 + (\theta_2 - \theta)^2\right]\right\} \tag{3.5}$$

The first term $(l_1 - l_2)^2$ accounts for the fact that the length of two eyes should be similar; the second term $(l_1 + l_2 - 2)^2$ enforces the constraints that distance between the two eyes should be about the width of one eye; the last two terms $(\theta_1 - \theta)^2$ and $(\theta_2 - \theta)^2$ enforce the constraint that both eyes should align with the base line. By using the exponent, this evaluation function is normalised in the range of 0 to 1.

For each pair of eye candidates, the relative regions S_1 and S_2 in the geometrical face model will be searched for nose and mouth, respectively. To achieve this, every block located within S_1 and S_2 will be evaluated. We use the mouth feature as an example to clarify the process. Let the centre of a block k be located at $(\overline{x_k}, \overline{y_k})$. The distance from the centre to the base line can thus be calculated as:

$$d_{\text{mouth}} = \frac{|a\overline{x_k} + b\overline{y_k} + c|}{\sqrt{a^2 + b^2}}$$

The following evaluation function can be employed to evaluate the probability of the block k

being the mouth feature

$$E_{\text{mouth}} = \exp\left[-4 \times \left(\frac{d_{\text{mouth}} - D_{\text{mouth}}}{D}\right)^2\right] \tag{3.6}$$

The term $\left(\frac{d_{\text{mouth}} - D_{\text{mouth}}}{D}\right)^2$ which accounts for the distance from the mouth centre to the base line should be appropriately constrained according to the geometrical face model. Refer to Figure 3.5 for the value of D and D_{mouth}. The divisor D and the outer multiplier -4 make the evaluation value have an equal influence to that of the eye feature (because there are four items to be added in equation (3.5), and only one item in equation (3.6). A multiplier of 4 will balance this difference. This evaluation function also ranges from 0 to 1. After the evaluation value for each block is obtained, the block having the largest value will be regarded as the mouth feature corresponding to the current pair of eye candidates. The corresponding nose feature can similarly be obtained by using the evaluation function as follows:

$$E_{\text{nose}} = \exp\left[-4 \times \left(\frac{d_{\text{nose}} - D_{\text{nose}}}{D}\right)^2\right] \tag{3.7}$$

After the above evaluation process have been finished for each pair of eye candidates, an overall evaluation function is defined as the weighted sum of the evaluation values of each facial feature:

$$E = 0.6E_{\text{eye}} + 0.2E_{\text{mouth}} + 0.2E_{\text{nose}} \tag{3.8}$$

The weighting parameters 0.6, 0.2 and 0.2 are set up by our prior experience that eyes are more prominent features than mouth and nose. We will choose the pair of eye candidates which maximises the overall evaluation function as the detected eyes.

By running the above process on Figure 3.1(a), we can finally obtain the facial features (eyes, nose and mouth) as shown in Figure 3.6.

3.2.2 Experimental Results and Discussions

This face detection method was tested with the fifth disk of the XM2VTS database. This disk contains face images of 295 subjects, each of which has 8 images which are taken in 4 sections (2 for each section). We take the images of the first 75 subjects as the testing set.

We define 'successful detection' as the centres of the two detected eye blocks exactly falling into the eye regions. This is checked manually. Based on such a criterion, this face detection method is successful in 324 of the 600 face images (75 × 8=600). The detection rate is 54.0%.

The relatively low detection rate is mainly due to two reasons. First, in some cases, the hair has similar colour to the skin, so the skin colour model can not filter out all the hair. Then the hair provides many tiny blocks after the edge detection, which make face detection a more difficult task (refer to Figure 3.7).

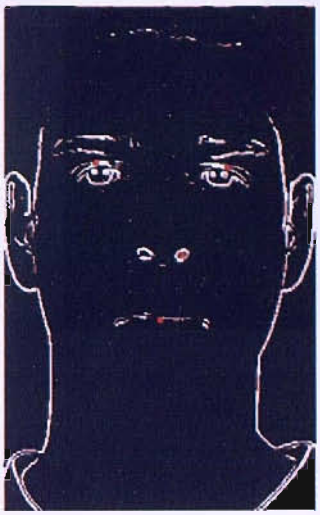
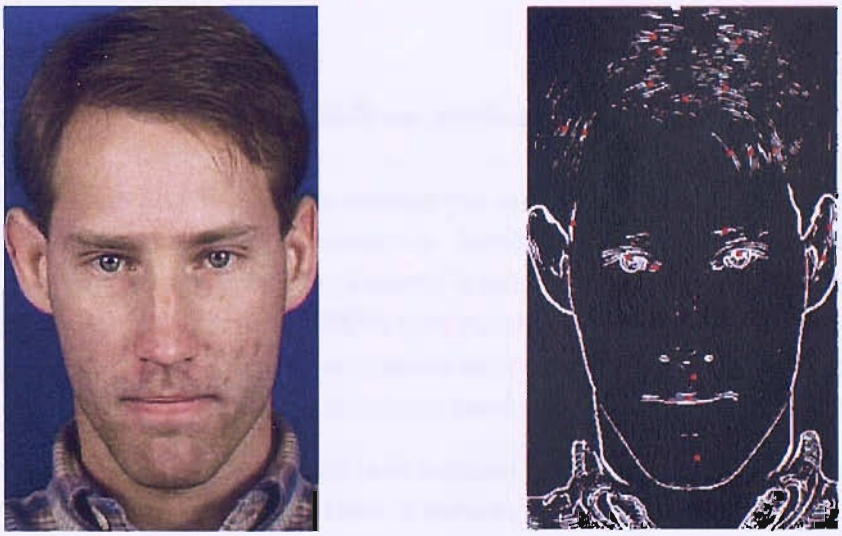


FIGURE 3.6: The detected facial features. Centres of facial features are represented by red points.



(a) (b)

FIGURE 3.7: An example in which the hair provides many tiny blocks. (a) The original face image. (b) Blocks of the face image. The centre of each block is represented by a red point.

(a) (b)

FIGURE 3.8: Another example in which the eye glasses link the two eyes together. (a) The original face image. (b) The blocks of the face image. Different blocks are represented by different colours.

Second, in cases when people wear glasses, the edges of glasses are likely to mix with the edges of eyes. If the eyes are linked with the glasses as one block, the detection method can not work. Figure 3.8 is one example in which eye glasses link the two eyes together.

The above two examples indicate the drawbacks of the feature-based approach. This approach is vulnerable to the unpredictability of face appearance and environmental conditions. Because of this, recent researches in face detection prefer image-based methods, which will be discussed in the next section.

3.2.3 Haar-like Features and AdaBoost Method for Face Detection

As can be seen from Section 3.2.2, the detection rate is comparatively low (54.0%) because of the unpredictability of environmental conditions. However, the image-based approach has the potential to overcome such a problem. Recently, AdaBoost image-based methods, which was firstly proposed by Viola and Jones (2004), have received much attention because so far they are the most successful ones in terms of detection accuracy and speed. In this section, we will shortly introduce the method, then build a system based on it and discuss its performance.

The features used in this method are Haar basis functions which have been used by Papageorgiou *et al.* (1998). The method uses three kinds of features, namely, two-rectangle features, three-rectangle features and four-rectangle features, which is shown in Figure 3.9. These Haar-like features are interesting because, firstly, powerful face/nonface classifiers can be constructed based on these features; secondly, they can be computed efficiently using the integral image technique which was proposed by (Crow, 1984).

A weak classifier can be built based on each Haar-like feature. However, the exhaustive set of Haar features is very large, but only a small portion of features has the ability to

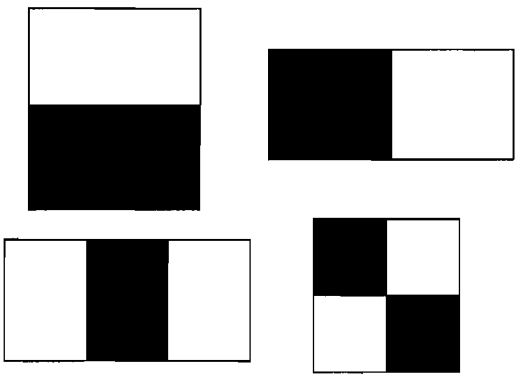


FIGURE 3.9: Four types of rectangular Haar wavelet-like features. A feature is a scalar calculated by summing up the pixels in the white region and subtracting those in the dark region. This figure is regenerated from Viola and Jones (2002).

discriminate face/nonface images. Thus, Viola and Jones (2002) proposed the AdaBoost algorithm to select features which are suitable for face detection. In its original form, the AdaBoost learning algorithm is used to boost the classification performance of a simple learning algorithm. It does this by combining a collection of weak classification functions to form a stronger classifier (Freund and Schapire, 1997). It has been proved that the training error of the strong classifier generated by AdaBoost approaches zero exponentially in the number of rounds (Freund and Schapire, 1997). A number of important results were later proved about its generalisation performance (Schapire *et al.*, 1997). Because of its resistance to over-fitting, AdaBoost has become a benchmark algorithm for classifier combination. Recently, Li and Zhang (2004) proposed another boosting algorithm, which is called 'FloatBoost', to replace the AdaBoost algorithm for selecting these Haar-like features.

Viola and Jones (2002) noticed that based on performance measured using a validation training set, the strong classifier which is trained by the AdaBoost algorithm can be adjusted to detect 100% of the faces with a false positive rate of 50%. Nevertheless the classifier can significantly reduce the number of sub-windows that need further processing, so they proposed the classifier cascade to solve this problem. The main idea of the classifier cascades is that they try to reject as many negative subwindows (subwindows which are regarded as nonface) as possible in the earliest stage, while retaining positive subwindows (subwindows which are regarded as face) to the final detection results. Subsequent classifiers are trained using those examples which pass through all the previous stages.

The above is only a short introduction to the Haar-like features and AdaBoost algorithm for face detection. Refer to Viola and Jones (2001); Lienhart *et al.* (2002); Viola and Jones (2002); Li and Zhang (2004) for details of this method.

(a) A face image

(b) The first five Haar-like features

FIGURE 3.10: One face image and the first five Haar-like features overlapped on it. We can see that these five features have very direct meanings for face detection. The first shows the intensity difference between the region of the eyes and the region across the upper cheeks. The second feature measures the difference in intensity between the region of the right pupil and the region of the right cheek. This analysis could be applied on the three other features here. Similar pictures are also illustrated in Viola and Jones (2002).

3.2.4 Experimental Results

In this section, we will use the Haar-like features and AdaBoost algorithm for face detection. This algorithm is implemented using the Intel Open Source Computer Vision Library (OpenCV) which was developed by the Intel Corporation (Bradski and Pisarevsky, 2000). OpenCV contains a special package which implemented the AdaBoost face detection algorithm. The classifier cascade consists of 20 stages, and for each stage, 10-15 features are selected. Figure 3.10 shows the first five Haar-like features in the first stage, which can be regarded as the most powerful features chosen by the algorithm to discriminate face and nonface images. It can be seen that these five features have very direct meanings for face detection.

The face detection algorithm is tested on the same dataset described in Section 3.2.2. Of the 600 face images, 554 images are correctly identified. The detection rate is 92.33%. The criteria for correct detection are that (1) the detected face contains the eyebrows, eyes, and mouth, and (2) no false detection occurs. Compared with the 54.0% detection rate by using the featuer-based method as discussed in Section 3.2.2, we can see that this method can obtain much better detection results. Figure 3.11 gives an example of the AdaBoost face detector.

The face images on which the detection is failed can be categorised into three groups. The first group is that, although the detector correctly finds the face, it also detects other smaller blocks which generate false alarms. The second group is that, the detector correctly find the face, however, it also regards other bigger blocks as faces. The third is that the detector does not

FIGURE 3.11: The face detection result generated by the AdaBoost algorithm. The face is represented by a red square.

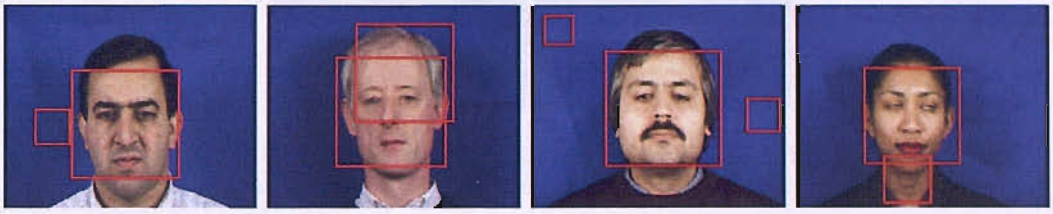


FIGURE 3.12: Representative detection errors in the first group. There are 39 images falling into this group, in which the correct faces are the biggest blocks generated by the detector. All detected blocks are marked by red squares.

detect face images at all. The first group occupies most of the failures. Of the 46 failures, There are 39 images falling into the first group, 6 in the second group, and only 1 in the third group. These images are shown in Figure 3.12, 3.13 and 3.14, respectively.

Because there are 39 images in the first group, we just select four representative images in Figure 3.12. We also list all images in the second and third groups (Figure 3.13 and 3.14). We can see that half of the errors in the second group are generated by one person, which reminds that the AdaBoost algorithm can fail in some face appearances. There is only one image which the AdaBoost algorithm fails detection (Figure 3.14). In this image, the subject lowered down her head, generating an extreme pose. Since the detector is trained by frontal-view images, it can not solve problems in such an extreme condition.

Because every image contains exactly one face, the detection failures in the first group can be be easily corrected by retaining only the biggest detected block in each image. We can fix most of the detection errors by using this method. By using the AdaBoost algorithm, we have almost solved the face detection problem. In Section 6.1, we combine the AdaBoost method with the dynamic programming (DP) algorithm to detect face in video files. We can then achieve 100% detection rate in the video files.

3.3 Image-Based Approach for Face Identification

Since the face detection problem is solved, the next step is face identification from the detected face images. Similar to face detection, face identification methods can roughly be divided into **feature-based approaches** and **image-based approaches**. Feature-based methods use the

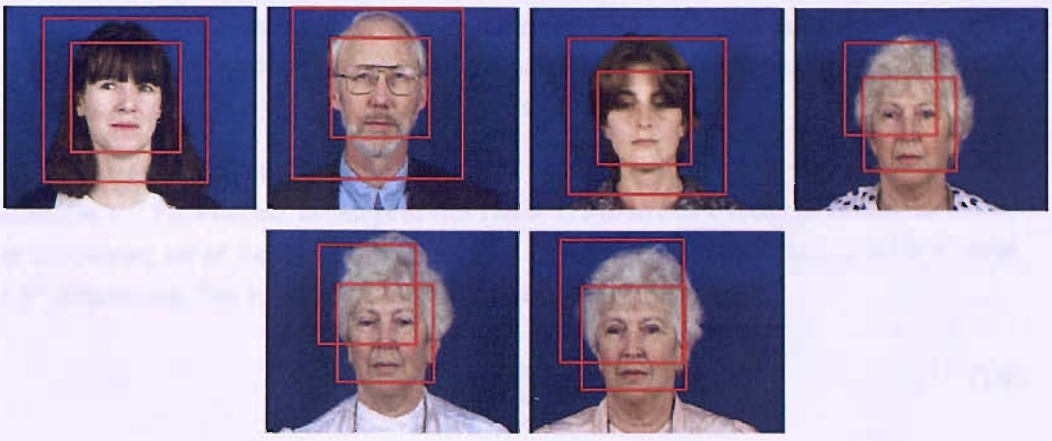


FIGURE 3.13: The 6 images in the second group, in which the correct face is not the biggest block generated by the detector. All detected blocks are marked by red squares.

FIGURE 3.14: There is only one face image in the third group, which the AdaBoost algorithm fails to detect at all.

information of the local statistics (geometric and/or appearance) of facial features such as the eyes, nose and mouth; while image-based methods use the whole face region as the raw input to a recognition system. Just as the human perception system uses both local features and the whole face region to recognise faces (Bartlett and Searcy, 1993; Tanaka and Farah, 1993), some face identification algorithms also incorporate both feature-based methods and image-based methods to achieve better identification results. We will discuss image-based methods in this section, and the feature-based methods in the following section.

3.3.1 A Short Review

The most commonly-used image-based method for face identification is the subspace method. Face images, represented as high-dimensional pixel arrays, often belong to a manifold of intrinsically low dimension. It is common to model the face space as a (possibly disconnected) *principal manifold* embedded in the high-dimensional image space. Its *intrinsic dimensionality* is determined by the number of degrees of freedom within the face space. There are several methods for reducing the dimensionality of face images, such as principal component analysis (PCA), linear discriminant analysis (LDA), and independent component analysis (ICA).

Principal component analysis (Jolliffe, 1986) is a dimensionality-reduction technique based on extracting the desired number of principal components of the multidimensional data.

Implementing PCA for face analysis and representation starts from the ground-breaking work of Kirby and Sirovich (1990). Their paper was followed by the 'eigenfaces' technique by Turk and Pentland (1991), the first application of PCA to face recognition.

Let a face image $I(x, y)$ be a two-dimensional N by N array of intensity values, or a vector of dimension N^2. For example, an image of size 128 × 128 describes a vector of dimension 16384. Let the training set of face images be $\Gamma_1, \Gamma_2, \ldots, \Gamma_M$. Each Γ_i $(i = 1, 2, \ldots, M)$ is a vector of N^2 dimensions. The average face of the set is defined by

$$\Psi = \frac{1}{M} \sum_{i=1}^{M} \Gamma_i \tag{3.9}$$

Each face differs from the average by the vector $\Phi_i = \Gamma_i - \Psi$. The eigenvectors are obtained by solving the eigenvalue problem

$$\Lambda = U^T C U \tag{3.10}$$

where Λ is a diagonal matrix, with eigenvalues on the main diagonal; U is an orthonormal matrix, which means that $U^{-1} = U^T$, or $U^T U = I$; C is the covariance matrix of the data:

$$\begin{aligned} C &= \sum_{i=1}^{M} \Phi_i \Phi_i^T \\ &= AA^T \end{aligned} \tag{3.11}$$

where the matrix $A = [\Phi_1\ \Phi_2\ \ldots\ \Phi_M]$, is a N^2 by M matrix. The matrix C, however, is N^2 by N^2, and determining the N^2 eigenvectors and eigenvalues is an intractable task for typical image sizes. Fortunately, it is possible to solve this problem by first solving a much smaller M by M matrix problem, and taking linear combinations of the resulting vectors. The following theorem indicates this possibility.

Theorem 3.1. *If $v_1, v_2, \ldots, v_M$ are eigenvectors of the matrix $A^T A$, with corresponding eigenvalues $\lambda_1, \lambda_2, \ldots, \lambda_M$, then $Av_1, Av_2, \ldots, Av_M$ are eigenvectors of AA^T, with the same eigenvalues $\lambda_1, \lambda_2, \ldots, \lambda_M$.*

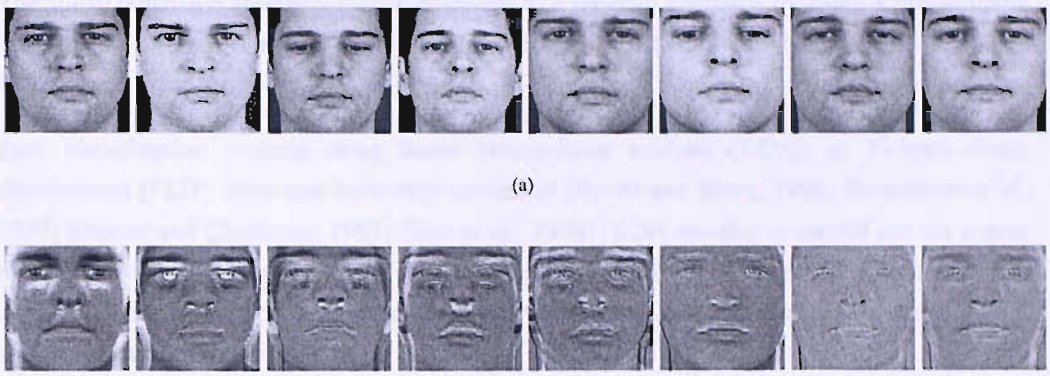
Proof: Since v_k $(k = 1, 2, \ldots, M)$ is an eigenvector of the matrix $A^T A$, with corresponding eigenvalue λ_k, then by the definitions of eigenvectors and eigenvalues, we have

$$(A^T A)v_k = \lambda_k v_k \tag{3.12}$$

Then the following deduction proves that Av_k is an eigenvector of AA^T.

$$\begin{aligned} (AA^T)(Av_k) &= A(A^T Av_k) \\ &= A(\lambda_k v_k) \\ &= \lambda_k (Av_k) \end{aligned} \tag{3.13}$$

That completes the proof.



(a)

(b)

FIGURE 3.15: Eight eigenfaces extracted from eight face images. (a) Eight face images. (b) The corresponding eigenfaces sorted by their eigenvalues.

The above theorem shows that in order to solve the eigenvalue problems on the $N^2 \times N^2$ matrix AA^T, we can firstly solve the problem on the $M \times M$ matrix A^TA, then obtain the eigenvectors of AA^T by multiplying the eigenvectors with A. With this method the calculations are greatly reduced from the order of the number of pixels in the images (N^2) to the number of images in the training set (M). In practice, the training set of face images will be relatively small ($M \ll N^2$), and the calculations become quite manageable. The associated eigenvalues allow us to rank the eigenvectors according to their usefulness in characterising the variation among the images. Figure 3.15 shows an example of the 8 eigenfaces calculated from 8 face images. Each face image is projected into the principal subspace.

Once the eigenfaces are created, identification becomes a pattern recognition task. The eigenfaces span an M'-dimensional subspace of the original N^2 image space. The M' significant eigenvectors of the matrix AA^T are chosen as those with the largest associated eigenvalues. A test face image Γ is transformed into its eigenface components (projecting into 'face space') by a simple operation, $p_k = u_k^T(\Gamma - \Psi)$, for $k = 1, 2, \ldots, M'$ (u_k is the kth significant eigenvector of AA^T). This describes a set of point-by-point image multiplications and summations. These weights form a weight vector $P^T = [p_1, p_2, \ldots, p_{M'}]$ that describes the contribution of each eigenface in representing the input face image.

Suppose $P_1, P_2, \ldots, P_M$ are weight vectors which are generated by the M training face images $\Gamma_1, \Gamma_2, \ldots, \Gamma_M$. For a test image Γ, we firstly generate its weight vector P, then we find the image in the training set which is most similar to it.

$$\text{Find } k = \arg\max_{i=1}^{M} \frac{|P \cdot P_i|}{\|P\|\|P_i\|} \tag{3.14}$$

We will classify the test image Γ into the class to which the training image Γ_k belongs.

$$\text{Classify } \Gamma \in \omega_s, \text{ if } \Gamma_k \in \omega_s \tag{3.15}$$

The above eigenface approach can be extended to a Baysian approach by using a probabilistic measure of similarity, instead of the simple Euclidean distance (Moghaddam and Pentland, 1997).

Face identification systems using linear discriminant analysis (LDA), or Fisher's linear discriminant (FLD), have also been very successful (Swets and Weng, 1996; Belhumeur *et al.*, 1997; Etemad and Chellappa, 1997; Zhao *et al.*, 1998). LDA training is carried out via scatter matrix analysis (Fukunaga, 1989). For an M-class problem (in the face identification example, M is the number of people to be identified), the within- and between-class scatter matrices S_w, S_b are computed as follows:

$$\begin{aligned} S_w &= \sum_{i=1}^{M} P(\omega_i)\Sigma_i \\ S_b &= \sum_{i=1}^{M} P(\omega_i)(m_i - m_0)(m_i - m_0)^T \end{aligned} \tag{3.16}$$

where $P(\omega_i)$ is the prior class probability, and is usually replaced by $1/M$ with the assumption of equal priors, m_i is the average face vector of the ith person, and m_0 is the average face vector of all M persons. Here S_w is the *within-class scatter matrix*, showing the average scatter Σ_i of the sample vectors x of different classes ω_i around their respective means m_i: $\Sigma_i = E\left[(x(\omega) - m_i)(x(\omega) - m_i)^T | \omega = \omega_i\right]$ and can be calculated by equation (3.11). Similarly, S_b is the *between class scatter matrix*, representing the scatter of the conditional mean vector m_i around the overall mean vector m_0. A commonly used measure for quantifying discriminatory power is the ratio of the determinant of the between-class scatter matrix of the projected samples to the determinant of the within-class scatter matrix:

$$J(T) = \frac{\left|T^T S_b T\right|}{\left|T^T S_w T\right|} \tag{3.17}$$

The optimal projection matrix Φ which maximises $J(T)$ can be obtained by solving a generalised eigenvalue problem:

$$S_b \Phi = S_w \Phi \Lambda_\Phi \tag{3.18}$$

where Λ_Φ is a diagonal matrix which contains all the eigenvalues of Φ along its diagonal.

Intuitively, LDA finds the projection of the data in which the classes are most linearly separable It can be proved that the dimension of Φ is at most $M - 1$. Because in practice S_w is usually singular, the Fisherfaces algorithm first reduces the dimensionality of the data with PCA and then applies LDA to reduce further the dimensionality to $M - 1$. The identification is then accomplished by a neural network classifier in this final subspace.

Based on the argument that for tasks such as face recognition much of the important information is contained in high-order statistics, it has been proposed by Bartlett *et al.* (1998) to use independent component analysis (ICA) to extract features for face recognition. ICA minimises

higher-order dependencies, and the components found by ICA are designed to be non-Gaussian. Suppose x is a N-dimensional vector ($x = [x_1, x_2, ..., x_M]^T$), ICA produces a $N \times M$ matrix A that projects x to an M-dimensional vector $y = [y1, y2, ..., y_M]$ ($M < N$) so that each component of y is independent (Jutten and Herault, 1991; Comon, 1994).

$$\begin{aligned} x &= Ay \\ A^T A &\neq I \\ P(y) &\approx \prod_{i=1}^{M} P(y_i) \end{aligned} \tag{3.19}$$

Bartlett *et al.* (1998) investigated the use of ICA framework for face recognition in two different architectures: the first is used to find a set of statistically independent source images that can be viewed as independent image features for a given set of training images, and the second is used to find image filters that produce statistically independent outputs (a factorial code method). In both architectures, PCA is used first to reduce the dimensionality of the original image size.

Other image-based approaches are also used for face identification. These include: kernel-PCA and kernel-Fisher methods (Schölkopf *et al.*, 1998; Yang, 2002), which are kernel-based extensions of PCA and LDA; support vector machines (Phillips, 1998); genetic algorithm (Liu and Wechsler, 2000); feature lines (Li and Lu, 1999); probabilistic decision-based neural networks (Lin *et al.*, 1997).

Recent advances of the image-based face recognition is to use 3-D images for training and testing (Bronstein *et al.*, 2005; Mpiperis *et al.*, 2007). 3-D face recognition has the potential to achieve better accuracy than its 2-D counterpart by measuring geometry of rigid features on the face. This avoids such pitfalls of 2D face recognition algorithms as change in lighting, different facial expressions, make-up and head orientation. Another approach is to use the 3-D model to improve accuracy of traditional 2-D recognition by transforming the head into a known view. One example of this approach is to use 3-D morphable models to reconstruct illumination- and pose-invariant face models from 2-D images, and use the reconstructed face models for recognition (Blanz *et al.*, 2002; Weyrauch *et al.*, 2003).

3.3.2 Face Identification System Based on PCA

In this section, a face identification system is implemented by using the PCA method. First, the face images are automatically detected by the AdaBoost face detection algorithm. For those images which fail in detection, the face regions are manually selected. Following the original PCA face recognition paper (Turk and Pentland, 1991), we bilinearly resample all face images to smaller size of 128 × 128 ($N = 128$), which forms a 16384-dimensional face space. By using PCA for the training images, the first 50 eigenvalues are extracted ($M' = 50$). Thus, every training face image in the 16384-dimensional space is projected to the 50-dimensional subspace. Then we also project the test images to this subspace, and classification results are obtained by

Training Session(s)	Testing Session(s)	Identification Rate (%)
1	2,3,4	65.11
2	1,3,4	63.78
3	1,2,4	68.00
4	1,2,3	66.22
1,2	3,4	76.00
1,3	2,4	81.33
1,4	2,3	79.33
2,3	1,4	79.33
2,4	1,3	77.00
3,4	1,2	74.33
1,2,3	4	84.00
1,2,4	3	81.33
1,3,4	2	82.67
2,3,4	1	76.67

TABLE 3.1: Identification rate of the PCA method. The identification rate is associated with the number of training images. It is shown that the identification rate is around 65% if only one session is used for training (as shown in the first 4 lines in the table); and when two sessions are used for training, the identification rate increases to around 79% (the next 6 lines). When three sessions are used for training, the identification rate further increases to around 81% (the last 4 lines).

using equations (3.14) and (3.15).

We use the face images in the fifth disk of the XM2VTS database to test the PCA face identification method. As indicated in Section 2.4.5, the fifth disk contains frontal face images for the 295 subjects. There are 8 images for each person, which are taken in 4 sessions (2 images for each session). We choose the first 75 subjects for the experiments, which is the same as we tested the speaker identification system in Section 2.4.6.

The settings of the experiments are similar to when we tested the speaker identification classifier in Section 2.4.6. We iteratively change the training sessions and testing sessions. First, we use one session for training, and the other three for testing; then use two sessions for training and the other two for testing; finally use three sessions for training and the other for testing. The identification results are shown in Table 3.1. We can see that the identification rate is associated with the number of training images. It is shown that the identification rate is around 65% if only one session is used for training (as shown in the first 4 lines in the table); and when two sessions are used for training, the identification rate increases to around 79% (the next 6 lines). When three sessions are used for training, the identification rate further increases to around 81% (the last 4 lines).

Similar to Section 2.4.6, we draw the ranking curves of the PCA method here. We also choose the ranking curves of 3 classifiers, which are, the classifiers in row 1, 5 and 11 of Table 3.1. The first uses Session 1 for training, and Sessions 2, 3 and 4 for testing; the second uses Sessions 1 and 2 for training, and Sessions 3 and 4 for testing; and the third uses Sessions 1, 2 and 3 for training, and Session 4 for testing. Compared with Figure 2.3, we can see that generally the

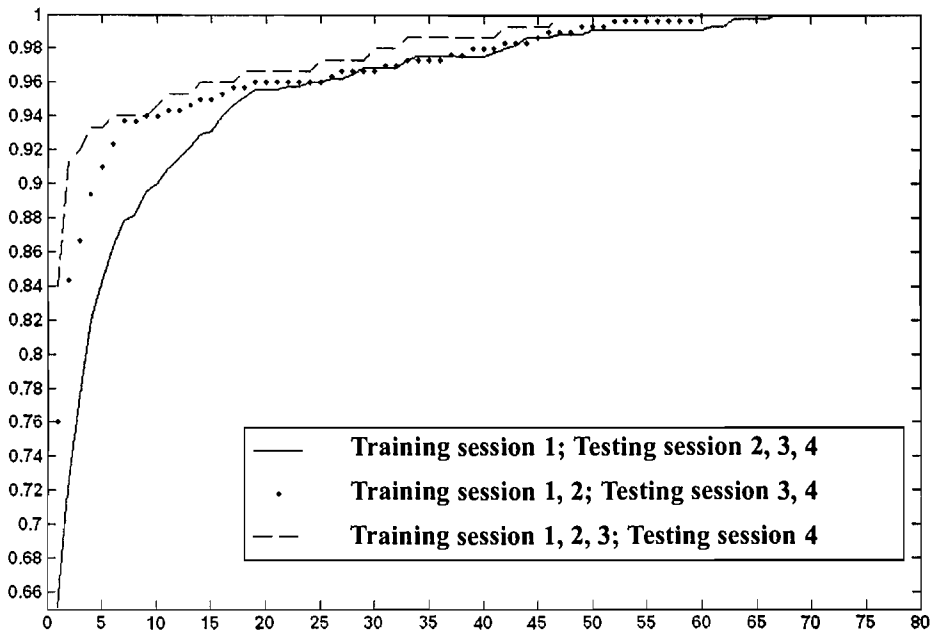


FIGURE 3.16: Ranking curves of three classifiers in Table 3.1, which are, the classifiers in row 1, 5 and 11. The first of these 3 classifiers uses Session 1 for training, and Sessions 2, 3 and 4 for testing; the second uses Sessions 1 and 2 for training, and Sessions 3 and 4 for testing; and the third uses Sessions 1, 2 and 3 for training, and Session 4 for testing. The three ranking curves are represented by the solid line, the dotted line and the dash line, respectively.

speaker identification classifier has higher identification rate. However, when only one session is used for training, the speaker identification classifier generates poorer performance for some testing files, which make the curve reach to 1 at a later stage.

3.3.3 More Discussions on the PCA Method

In Section 3.3.2, the PCA method is used to map a 16384-dimensional vector to a 50-dimensional vector in a subspace which is formed by the eigenvalues of the training images. This process has raised a question that whether this dimensional-reduction technique increases separability of classes, i.e., whether the vectors generated by PCA are more suitable for recognition than the original vectors. In this section, we will show that the vectors generated by PCA are better than the original ones for face recognition. Firstly, we extract face images of two different subjects (Subject Number: '000' and '001'). As described in Section 3.3.2, each subject has 8 images, which are taken in 4 sessions. These face images are also resampled to a smaller size of 128 $\times$ 128, as shown in Figure 3.17. By using PCA, the original 16384-dimensional vectors are mapped to 16-dimensional vectors (Here we can only obtain 16-dimensional vectors, contrasting to the 50-dimensional vectors in Section 3.3.2, because there are only 16 images).

A mapping algorithm, which is called Sammon's mapping (Sammon, 1969), is used to map

(a) Face images for subject '000'

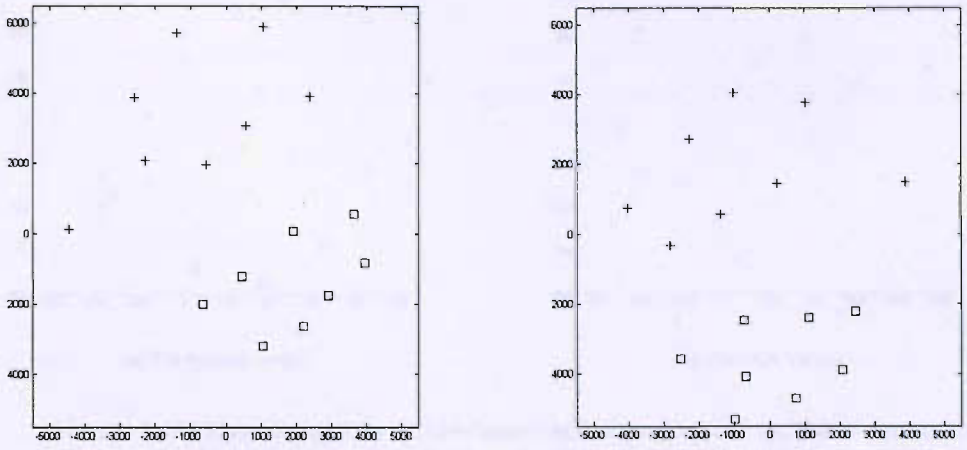
(b) Face images for subject '001'

FIGURE 3.17: Face images of two subjects, with subject number '000' and '001' in the XM2VTS database.

the high-dimensional vectors to two-dimensional so that we could draw these vectors on a graph. This algorithm intends to preserve the relative distance between the input points approximately. Figure 3.18(a) and 3.18(b) shows the 16384-dimensional original vectors and the 16-dimensional PCA vectors after the Sammon's mapping, respectively. The vectors which are labelled by plus signs are generated by face images of subject '000', and the vectors labelled by squares are generated by subject '001'. The figure shows that both the original vectors and the PCA vectors are completely separated with respect to these two classes, which indicate that both PCA and simple vector comparison method can perform well when the intra-class distances are big enough.

However, when two classes are similar and overlapped in the feature space, the PCA method can perform better than the simple comparison method. This situation is shown in Figure 3.19 and 3.20. Figure 3.19 shows face images of two subjects with subject number '010' and '211' in the database. They are similar in appearances.

The original and the PCA vectors of the face images are shown in Figure 3.20. Vectors which are labelled by plus signs are generated by subject '010', and those labelled by squares are generated by subject '211'. These subjects are similar in appearances, so their vectors mixed with each other. We can see that the PCA vectors are more separable on the graph. A linear classifier can correctly separate most PCA vectors of these two classes, while more complex hyperplane needs to be assumed if we need to separate the vectors in their original form. By using PCA, we have increased the separability of data, so potentially achieving better recognition results than simple vector comparison method.



(a) The original vectors

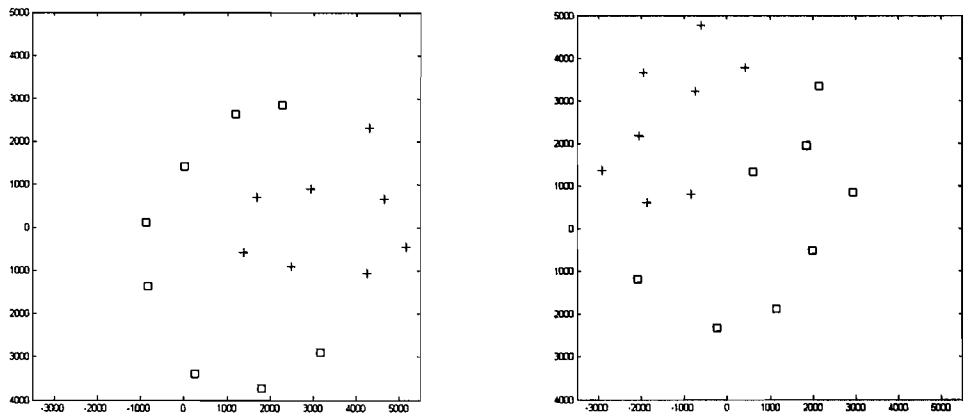
(b) The PCA Vectors

FIGURE 3.18: Illustrations of the 16384-dimensional original vectors and the 16-dimensional PCA vectors. These vectors are converted to two-dimensional vectors by the Sammon's mapping, which can be shown on the plane. The vectors which are labelled by plus signs are generated by face images of subject '000', and the vectors labelled by squares are generated by face images of subject '001'.

(a) Face images for subject '010'

(b) Face images for subject '211'

FIGURE 3.19: Face images of two subjects, with subject number '010' and '211' in the XM2VTS database. They have very similar appearances.



(a) The original vectors

(b) The PCA Vectors

FIGURE 3.20: Illustrations of the 16384-dimensional original vectors and the 16-dimensional PCA vectors of subject '010' and '211'. Vectors which are labelled by plus signs are generated by subject '010', and those labelled by squares are generated by subject '211'. These subjects are similar in appearance, so their vectors mixed with each other on the graph. We can see that the PCA vectors are more separable than the original ones.

3.4 Feature-Based Approach for Face Identification

In this section, the feature-based approach for face identification is firstly reviewed, then a face identification system based on dynamic link architecture (Lades *et al.*, 1993) is presented and tested on the XM2VTS database.

3.4.1 Feature-Based Approach for Face Identification

Many methods in the feature-based category have been proposed, including early methods based on geometrical relationships of local features (Kelly, 1970; Kanade, 1973) as well as 1D (Samaria and Young, 1994) and pseudo-2D (Samaria, 1994) HMM methods. One of the most successful methods is the elastic bunch graph matching (EBGM) system (Wiskott *et al.*, 1997), which is based on dynamic link architecture (DLA) (Buhmann *et al.*, 1990; Lades *et al.*, 1993; Okada *et al.*, 1998). Wavelets, especial Gabor wavelets, play a building-block role for facial representation in these graph matching methods. A typical local-feature representation consists of wavelet coefficients for different scales and rotations based on fixed wavelet bases. These locally estimated wavelet coefficients has some robustness to illumination changes, translation, distortion, scaling and rotation of images. DLA attempts to solve some of the conceptual problems of conventional artificial neural networks, the most prominent of these being the representation of syntactical relationships. DLA uses synaptic plasticity and is able to form sets of neurons grouped into structured graphs while maintaining the advantages of neural systems.

The DLA architecture was extended to elastic bunch graph matching (Wiskott *et al.*, 1997). This

is similar to DLA, but instead of attaching only a single jet to each vertex, the authors attached a set of jets, each derived from a different face image (refer to Section 3.4.2 for definitions of 'jet' and 'vertex'). To handle the pose-variation problem, the pose of the face is first determined using prior class information (Kruger *et al.*, 1997), and the 'jet' transformations under pose variation are learned using training images (Maurer and Malsburg, 1996). The success of the DLA and EBGM system may be due to its resemblance to the human visual system (Biederman and Kalocsai, 1998).

3.4.2 System Description

In this system, we use Gabor wavelets to represent the face image. Let $I(\vec{x})$ be the grey-level distribution of the input image, where $\vec{x} = [x_1, x_2]$ represents the two coordinates of each pixel on the image. The Gabor wavelet transform can be written as a convolution of the image I with a family of kernels $\psi_{\vec{k}}$. The parameter $\vec{k}$ determines the wavelength and orientation of the kernel $\psi_{\vec{k}}$. The Gabor wavelet operator W symbolises the convolution with all possible $\vec{k}$:

$$(WI)(\vec{k}, \vec{x}) = \int \psi_{\vec{k}}(\vec{x} - \vec{x_0}) I(\vec{x_0}) \mathrm{d}\vec{x_0} = (\psi_{\vec{k}} * I)(\vec{x}) \tag{3.20}$$

The kernel $\psi_{\vec{k}}$ takes the form of a plane wave restricted by a Gaussian envelope function:

$$\psi_{\vec{k}}(\vec{x}) = \frac{\vec{k}^2}{\sigma^2} \exp\left(-\frac{\vec{k}^2 \vec{x}^2}{2\sigma^2}\right) \left[\exp(i\vec{k} \cdot \vec{x}) - \exp\left(\frac{-\sigma^2}{2}\right)\right] \tag{3.21}$$

The first term in the square brackets determines the oscillatory part of the kernel. The second term compensates for the dc value of the kernel, to avoid unwanted dependence of the filter response on the absolute intensity of the image. The complex-valued $\psi_{\vec{k}}$ combines an even (cosine-type) and odd (sine-type) part. The Fourier transform of $\psi_{\vec{k}}$ is given by:

$$F(\psi_{\vec{k_0}}) = \exp\left(-\frac{\sigma^2(\vec{k_0} - \vec{k})^2}{2\vec{k}^2}\right) - \exp\left(-\frac{\sigma^2(\vec{k_0}^2 + \vec{k}^2)}{2\vec{k}^2}\right) \tag{3.22}$$

The first exponential centred at the characteristic frequency $\vec{k}$ provides a bandpass filter. The second exponential removes the dc component of $\psi_{\vec{k}}$. The reader may refer to Mallet (1999) and Chui (1992) for a detailed description of the wavelet transform and Gabor wavelets.

To generate a local description of a face image, we sample W at five logarithmically spaced frequency levels and eight orientations indexed by $v \in \{0, ..., 4\}$ and $\mu \in \{0, ..., 7\}$, which was suggested by Lades *et al.* (1993).

$$\vec{k}_{v\mu} = [k_v \cos(\phi_\mu), k_v \sin(\phi_\mu)], \text{ with } k_v = \frac{k_{\max}}{f^v}, \text{ and } \phi_\mu = \frac{\pi\mu}{8} \tag{3.23}$$

where f is the spacing factor between kernels in the frequency domain. In this system, we take $f = \sqrt{2}$, $k_{max} = \frac{\pi}{2}$, and variance $\sigma = 1$. The magnitude of $(WI)(\vec{k}_{v\mu}, \vec{x})$ forms a 40-

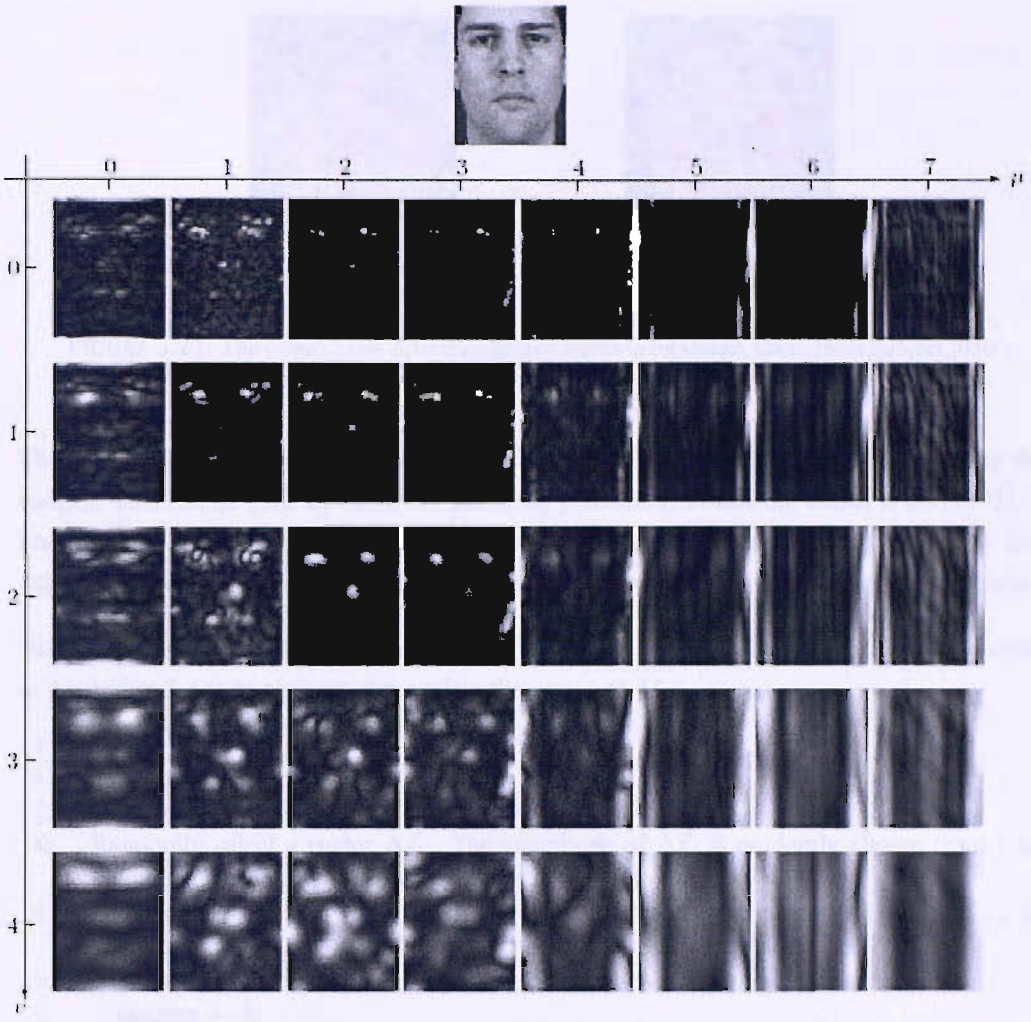
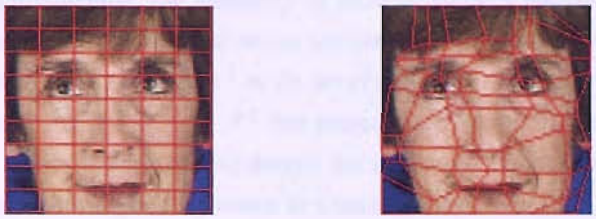


FIGURE 3.21: Gabor wavelet representation for a face image. These 40 subimages are obtained with υ varying from 0 to 4 and μ from 0 to 7.

dimensional feature vector $J(\vec{x})$ for each pixel $\vec{x}$ in the face image, which will be referred to as a 'jet':

$$J(\vec{x}) := \left\{ J_{\upsilon\mu}(\vec{x}) | J_{\upsilon\mu}(\vec{x}) = \left| (WI)(\vec{k}_{\upsilon\mu}, \vec{x}) \right| \right\}, \ \upsilon = \{0, ..., 4\} \text{ and } \mu = \{0, ..., 7\} \qquad (3.24)$$

Figure 3.21 shows the 40 Gabor wavelet components of a face image. Several observations need to be mentioned here. Firstly, as υ becomes greater, the Gabor wavelet image becomes more and more blurred. Such a phenomenon is very similar to the Fourier transform of images. From equation (3.23), the greater υ corresponds to a lower frequency component, which makes the image blurred. The second observation is that the Gabor wavelet components have greater values near facial features, so it makes the positions of eyes, nose and mouth brighter than other areas. This demonstrates the Gabor wavelet's ability to select and analyse facial features.



(a) (b)

FIGURE 3.22: The original and deformed grid of the training image, taken from Subject '050'. (a) A training image with a square grid on it. (b) The deformed grid on this training image.

The first step of our system is to construct face models automatically from the training face images. This can be done by using the following procedure. Following Lades *et al.* (1993), we firstly place a 7×10 square grid M onto a training image, as indicated in Figure 3.22(a). Each intersection point of this grid is called a 'vertex'. Thus, we obtain 70 vertices for each image.

We use the 70 feature vectors at a vertex of M to form a template. Then the procedure described in Algorithm 3.1 is implemented to update the vertex of M.

counter $\leftarrow 0$
while counter $< N$ **do**
  Pick a vertex in M at random. Its position is denoted as $\vec{x}_i$.
  Randomly select a vector $\Delta\vec{x}_i$. The magnitude of $\Delta\vec{x}_i$ is randomly chosen from 1 to a predefined value $\Delta_{\max}$. The direction of $\Delta\vec{x}_i$ is also randomly chosen.
  if the point $(\vec{x}_i + \Delta\vec{x}_i)$ falls into the area surrounded by the neighbour vertex of $\vec{x}_i$, and $\|J(\vec{x}_i + \Delta\vec{x}_i)\| > \|J(\vec{x}_i)\|$ **then**
    $\vec{x}_i \leftarrow \vec{x}_i + \Delta\vec{x}_i$
    counter $\leftarrow 0$
  else
    counter $\leftarrow$ counter $+ 1$
  end if
end while

ALGORITHM 3.1: Algorithm for updating the vertices on training images.

The original paper, Lades *et al.* (1993), sets $N = 100$ and $\Delta_{\max}$ equals the distance of 10 pixels (refer to Algorithm 3.1 for the meaning of these parameters). However, more accurate matching is obtained when N is greater. In this experiments, ere we set $N = 500$ and keep the same value of $\Delta_{\max}$. We also require that $(\vec{x}_i + \Delta\vec{x}_i)$ falls into the area surrounded by the neighbour vertex of $\vec{x}_i$. This requirement guarantees that the topological relationship of the grid M is not changed during the updating procedure. Figure 3.22(b) shows the deformed grid. We can see that vertices in the deformed grid gather at the prominent facial features (e.g., eyes, nose and mouth), which indicates a good modelling of face images. If there are K training images, the deformed grids of these training images can be obtained by the above procedure. We denote these K deformed grids as M_1, M_2,..., M_K.

As part of elastic graph matching, the similarity of pairs of vertices has to be evaluated. We settle for the normalised dot product of jets as our similarity function. We denote V^M as the set of all vertices in a training image M, and V^I as the set of all vertices in a testing image. We need to find a one-to-one mapping $f: V^M \rightarrow V^I$ that projects each vertex in the training image to one of the vertices in the testing image. We denote the similarity function between a vertex $\vec{x}_i$ in a training image and its corresponding vertex in a testing image as

$$S_v\left(J^M(\vec{x}_i), J^I(f(\vec{x}_i))\right) = \frac{J^M(\vec{x}_i) \cdot J^I(f(\vec{x}_i))}{\|J^M(\vec{x}_i)\| \|J^I(f(\vec{x}_i))\|} \tag{3.25}$$

where J^M and J^I are the Gabor wavelet 'jets' in the training image and the testing image, respectively.

Another similarity that relates to the topology between vertices in the training image and vertices in the testing image also needs to be considered. We model the topological similarity by the constraint of neighbouring vertices in the training image matching to neighbouring vertices in the testing image. Suppose there are two adjacent vertices $\vec{x}_i$ and $\vec{x}_j$ in the training image. We denote the Euclidean distance vector between them as:

$$\vec{\Delta}_{ij}^M = \vec{x}_j - \vec{x}_i, \quad (i, j) \in E \tag{3.26}$$

where E is the set of all adjacent vertex pairs in the training image. We can also denote their corresponding Euclidean distance vector in the testing image as:

$$\vec{\Delta}_{ij}^I = f(\vec{x}_j) - f(\vec{x}_i), \quad (i, j) \in E \tag{3.27}$$

The vertex labels of the testing image are compared to the corresponding ones in the training image by a quadratic comparison function

$$S_e(\vec{\Delta}_{ij}^I, \vec{\Delta}_{ij}^M) = (\vec{\Delta}_{ij}^I - \vec{\Delta}_{ij}^M)^2 \tag{3.28}$$

The second step of our system is elastic matching of a training image M to a testing image I, which amounts to a search for a one-to-one mapping f: $V^M \rightarrow V^I$ of vertex positions which simultaneously optimises S_v and S_e. We evaluate the quality of a mapping according to the cost function:

$$S(M, f) = \lambda \sum_{(i,j)\in E} S_e(\vec{\Delta}_{ij}^I, \vec{\Delta}_{ij}^M) - \sum_{i \in V^M} S_v\left(J^I(\vec{x}_i), J^M(f(\vec{x}_i))\right) \tag{3.29}$$

which is a linear combination of the S_e term and the S_v term. The coefficient λ controls the rigidity of the image graph, large values penalising distortion of I with respect to M. In our system, we choose $\lambda = 3 \times 10^{-4}$.

The elastic matching procedure is very similar to the first step. Firstly, we bilinearly resize both the training image and testing image to 128×128 pixels, then directly put the grid of the training image M onto the testing image. That is, we start the searching process by the self-

(a) (b)

FIGURE 3.23: Elastic template matching for a testing image of Subject '050'. (a) The grid as shown in Figure 3.22(b) is directly put to a testing image. (b) The deformed grid on the testing image after the elastic matching process. We can see that the deformed grid contains more prominent facial features, thus achieving a better matching.

mapping $f(\vec{x}_i) = \vec{x}_i$ for all vertices $\vec{x}_i$. After that, Algorithm 3.2 is used to find a mapping f which minimises the cost function $S(M, f)$.

counter $\leftarrow 0$
while counter $< N$ **do**
  Pick a vertex in V^M at random. Its position is denoted as $\vec{x}_i$.
  Suppose $f(\vec{x}_i) = \vec{x}_j$ ($\vec{x}_j \in V^M$). Randomly select a vector $\Delta\vec{x}_j$. The magnitude of $\Delta\vec{x}_j$ is randomly chosen from 1 to a predefined value $\Delta_{\max}$. The direction of $\Delta\vec{x}_j$ is also randomly chosen.
  if the point $(\vec{x}_j + \Delta\vec{x}_j)$ falls *out of* the area surrounded by the neighbour vertices of $\vec{x}_j$ in the image domain. **then**
    counter $\leftarrow$ counter $+ 1$
  else
    Define a new mapping f' that $f'(\vec{x}_i) = \vec{x}_j + \Delta\vec{x}_j$ and for other $t \neq i$, $f'(\vec{x}_t) = f(\vec{x}_t)$.
    if $S(M, f') < S(M, f)$ **then**
      $f \leftarrow f'$
      counter $\leftarrow 0$
    else
      counter $\leftarrow$ counter $+ 1$
    end if
  end if
end while

ALGORITHM 3.2: Algorithm for finding a mapping from the model graph to the image graph

We also choose $N = 500$ and $\Delta_{\max}$ equals the distance of 10 pixels, the same as in the first step. Figure 3.23 shows the elastic matching procedure as discussed above. We could see that the deformed grid contains more prominent facial features, thus achieving a better matching.

If we put a testing image X into each of the K face models M_1, M_2,..,M_K, we can obtain the values of $S(M_1, f)$, $S(M_2, f)$,...,$S(M_K, f)$ respectively by using Algorithm 3.2. We can then identify the testing image according to the rule:

$$\text{Find } k = \arg\min_i S(M_i, f), \text{ then decide } X \in \omega_s, \text{ if } M_k \in \omega_s$$

Training Session(s)	Testing Session(s)	Identification Rate (%)
1	2,3,4	78.44
2	1,3,4	82.22
3	1,2,4	80.22
4	1,2,3	83.56
1,2	3,4	88.00
1,3	2,4	91.00
1,4	2,3	92.33
2,3	1,4	91.67
2,4	1,3	94.00
3,4	1,2	88.33
1,2,3	4	92.67
1,2,4	3	95.33
1,3,4	2	90.67
2,3,4	1	90.67

TABLE 3.2: Identification rate of the DLA method. It is shown that the identification rate is around 80% if only one session is used for training (as shown in the first 4 lines of the table); and when two sessions are used for training, the identification rate increases to around 90% (the next 6 lines). When three sessions are used for training, the identification rate further increases to around 92% (the last 4 lines).

3.4.3 Experimental Results and Discussions

The settings of the experiments to test the DLA face identification method are the same as we used to test the PCA method in Section 3.3.2. The identification results are shown in Table 3.2. Compared with Table 3.1 , we can see that the identification rate of DLA outperforms PCA by around ten percentage points. The expense is that DLA is two times slower than the PCA method in the training process, and ten times slower in the testing process. More research needs to be carried out to improve the performance of PCA and the speed of the DLA method.

Here we also draw the ranking curve of the DLA face identification classifier in Figure 3.24. The settings are the same with those in Figure 2.3 and 3.16. We can see that the identification rate is increased when more images are used for training.

3.5 Summary

In this chapter, we have discussed the problem of building an automatic face identification system. This problem could be further divided into two steps – face detection and face identification. Two methods are used for face detection and their results are compared. The first one is based on extracting facial features by lower-order information such as facial colour, edges and geometry of facial features. The second is based on Haar-like features and AdaBoost. Experiments indicate that the second method achieves better results than the first one. Because of its advantages in speed and performance, the Haar-like features and Boosting algorithms have become a standard of current face detection research. However, as we can see from the

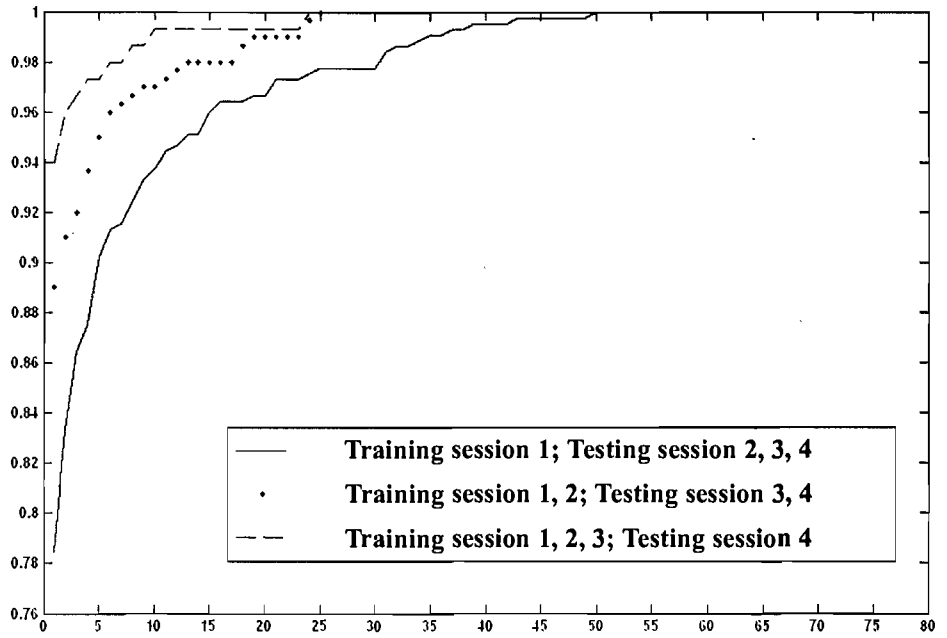


FIGURE 3.24: Ranking curves of three classifiers in Table 3.2, which are, the classifiers in row 1, 5 and 11. The first of these 3 classifiers uses Session 1 for training, and Sessions 2, 3 and 4 for testing; the second uses Sessions 1 and 2 for training, and Sessions 3 and 4 for testing; and the third uses Sessions 1, 2 and 3 for training, and Session 4 for testing. The three ranking curves are represented by the solid line, the dotted line and the dash line, respectively.

experimental results, there are still some situations which the detector can not produce accurate performance.

Two algorithms are used for the face identification step, which are principle component analysis (PCA) and dynamic link architecture (DLA). The PCA method has poorer identification results than DLA, however, it is much faster. There should be some trade-off between time and performance for these two methods. In addition, better algorithms also needs to be carried out in the face recognition area to increase both speed and performance.

Chapter 4

Combining Classifiers

4.1 Introduction

In Chapters 2 and 3, we discussed the process of building a speaker identification system and a face identification system, respectively. In this chapter, we will concentrate on the problem of how to combine the scores of these two systems together to obtain better identification performance. The idea of combining scores of multi-classifiers appears under a variety of names in the literature: classifier fusion (Gader *et al.*, 1996; Keller *et al.*, 1994); classifier combination (Kittler *et al.*, 1998a; Lam and Suen, 1995; Woods *et al.*, 1997; Xu *et al.*, 1992); mixture of experts (Jacobs, 1995; Jacobs *et al.*, 1991; Jordan and Xu, 1995; Nowlan and Hinton, 1991); committees of neural networks (Bishop, 1995; Drucker *et al.*, 1994); consensus aggregation (Benediktsson *et al.*, 1997; Ng and Abramson, 1992; Benediktsson and Swain, 1992); voting pool of classifiers (Battiti and Colla, 1994); classifier ensembles (Drucker *et al.*, 1994; Filippi *et al.*, 1994). In this chapter, we will refer to this problem by using either **classifier fusion** or **classifier combination**.

It is widely agreed that the audio and visual modalities can be combined at three different levels, which is defined by Lucey *et al.* as early integration, middle integration and late integration, respectively. For early integration, the feature vectors of audio and visual signals are extracted separately, then vector concatenation is employed to form a new feature vector, finally, this new feature vector is used for recognition (Adjoudani and Benoît, 1995; Luettin, 1997). For late integration, the audio and visual classifiers are built separately, then fusion methods are implemented to combine the scores generated by the audio and visual classifiers (Kittler *et al.*, 1998b; Ben-Yacoub *et al.*, 1999; Toh and Yau, 2004). Another level of integration, namely middle integration, is also frequently used for combining audio and visual modalities. Examples of this level integration are multistream hidden Markov models (Fu *et al.*, 2003; Bengio, 2003; Lucey *et al.*, 2005).

However, the most frequently-used methods appearing in the literature are based on late integration. This is because of two reasons. First, compared with early integration and

middle integration, late integration is simple. It does not take into account the correlation and interaction of audio and visual signals, thus circumventing the problem of synchronising audio and visual signals. Instead, it treats these two modalities separately, obtaining two separate classifiers, then processing scores generated by these two classifiers. Second, late integration achieves commensurate, if not better, recognition rates compared with early integration and middle integration. Lucey *et al.* compared different approaches of the early integration, middle integration and late integration, and found that late integration is superior in terms of classifier flexibility and its ability to dampen independent errors coming from either modality.

Of all approaches in late integration, the simplest are based on some fixed rules, e.g., the sum rule, product rule etc (Kittler *et al.*, 1998b; Duin, 2002). The scores generated by the audio and visual classifiers are combined by some fixed functions, and training the combined classifier is not needed. It has been shown that by using fixed rules, the performance of the person recognition system can be greatly improved (Kittler *et al.*, 1998b; Erzin *et al.*, 2005). Kittler *et al.* (1998b) attempted to build a theoretical framework for the fixed rules. Their experimental results for combining the scores from three experts (two face experts and a text-dependent speaker expert) showed that the sum rule outperformed the product rule. A small revision to the fix rules is to assigning weighting parameter(s) to each modality based on the performance of that modality, the so called weighted sum rule and weighted product rule. Experiments showed that weighted sum and product rules performed better than fixed sum and product rules (Chibelushi *et al.*, 1993; Brunelli and Falavigna, 1995; Maison *et al.*, 1999; Wark, 2000; Sanderson and Paliwal, 2003). Although various methods are proposed to choose weighting parameters, there is not a unanimous agreement on how to do this.

In this chapter, we will firstly describe the problem of late integration of classifiers. then discuss several frequently-used combination schemes. We will also propose a new method for accurately choosing the optimal weighting parameter(s) for audio-visual person identification. Finally the proposed method is compared with three other well-established methods. Using the bootstrapping method, we conclude that our approach can both reduce the bias and variance, thus achieving a better estimation for the optimal weighting parameter.

4.2 General Problem for Late Integration of Classifiers

Suppose the task of a classifier is to assign an input vector X to one of K classes $\omega_1, \omega_2,...,\omega_K$. This classifier consists of K discriminant functions (or scores), $f^1(X)$, $f^2(X),...,f^K(X)$ respectively. The decision rule in terms of discriminant functions is:

$$\text{decide } X \in \omega_s \text{ if } s = \arg\max_{k=1}^{K} f^k(X) \tag{4.1}$$

Here, the discriminant functions $f^1(X)$, $f^2(X),...,f^K(X)$ are specified by the classification algorithm. They can be the probability that X belongs to each class, or the distance between

X and the centre of each class, or take any other form.

Now consider there are M classifiers. As discussed above, each classifier has K discriminant functions. In this chapter, we will always use M as the number of classifiers, and K as the number of classes. Let $f_m^1(X)$, $f_m^2(X)$,...,$f_m^K(X)$ denote the K discriminant functions of the mth classifier ($m = 1, 2, ..., M$). Obviously the decision rule of the mth classifier is :

$$\text{decide } X \in \omega_s \text{ if } s = \arg\max_{k=1}^{K} f_m^k(X) \tag{4.2}$$

The combination process of the M classifiers can be described as finding a set of combination functions $F_k(X)$, ($k = 1, 2, ..., K$), so that there will be a new set of discriminant functions.

$$\begin{aligned} f_{\text{comb}}^1(X) &= F_1\left(\{f_m^k(X)\}\right) \\ f_{\text{comb}}^2(X) &= F_2\left(\{f_m^k(X)\}\right) \\ &\vdots \\ f_{\text{comb}}^K(X) &= F_K\left(\{f_m^k(X)\}\right) \end{aligned} \tag{4.3}$$

The set $\{f_m^k(X)\}$ contains all discriminant functions with m varying from 1 to M, and k from 1 to K. The decision rule for combining these M classifiers is an extension of the decision rule (4.1):

$$\text{decide } X \in \omega_s \text{ if } s = \arg\max_{k=1}^{K} f_{\text{comb}}^k(X) \tag{4.4}$$

Based on the discussion above, we can define the general problem of combination of classifiers as follows.

Definition 4.1. Suppose there are M classifiers, each of which will classify an input vector X into one of K classes. Let $f_m^k(X)$ denote the scores of the mth classifier assigning to the kth classifier ($m = 1, 2, \ldots, M$ and $k = 1, 2, \ldots, K$). The problem of combination of classifiers is to find a set of combination functions so that the correct identification rate is maximised on testing data.

Table 4.1 provides an example to clarify the combination problem. Suppose there are 2 classifiers ($M = 2$), each of which classifies an input vector into one of 2 classes ($K = 2$). In this table, there are 4 training vectors, $X1$, $X2$, $X3$, $X4$. For each of these training vectors, we know both the class label and the scores provided by each of the two classifiers.

For example, we can see from Table 4.2(a) that the training vector $X1$ belongs to class 1, and the first classifier assigns score 7 to class 1, and 6 to class 2. (refer to the $m = 1$ column in Table 4.2(a). According to decision rule (4.2), the first classifier correctly identifies that $X1$ belongs to class 1. Similarly, the second classifier assigns score 5 to classifier 1 and 3 to classifier 2. It also correctly identifies that $X1$ belongs to class 1. Similar analysis can be applied to the training data $X2$, $X3$, $X4$.

TABLE 4.1: An example to clarify the problem of classifier combination. The information of the training data $X1$, $X2$, $X3$, $X4$ is provided in (a)–(d). A class label needs to be assigned to the testing data $X5$.

	$f_m^1(X1)$	$f_m^2(X1)$	Class label
$m = 1$	7	6	1
$m = 2$	5	3	

(a) Class label and scores for $X1$

	$f_m^1(X2)$	$f_m^2(X2)$	Class label
$m = 1$	10	12	2
$m = 2$	8	9	

(b) Class label and scores for $X2$

	$f_m^1(X3)$	$f_m^2(X3)$	Class label
$m = 1$	12	10	1
$m = 2$	5	6	

(c) Class label and scores for $X3$

	$f_m^1(X4)$	$f_m^2(X4)$	Class label
$m = 1$	9	5	2
$m = 2$	4	10	

(d) Class label and scores for $X4$

	$f_m^1(X)$	$f_m^2(X)$	Class label
$m = 1$	10	12	?
$m = 2$	8	9	

(e) Only scores are provided for the testing data $X5$. A class label needs to be assigned.

From Definition (4.1), the problem of combination of classifiers is to devise an algorithm by using the information in Table 4.2(a)–4.2(d) to maximise the correct identification rate on testing data. Suppose we input testing data $X5$ into these two classifiers. We need to assign a class label (the question mark in Table 4.2(e)) to $X5$ based on both the training data (data in Table 4.2(a)–4.2(d)) and the scores of $X5$ (Data in Table 4.2(e)).

From the discussion above, we can see that the problem of classifier combination is very similar to many machine learning problems. According to Mitchell (1997), in order to have a well-defined learning problem, we must identify three features: the class of tasks, the measure of performance to be improved, and the source of experience. The problem of classifier combination can be defined as:

- Task T: Find a combination algorithm which can properly combine scores of classifiers.
- Performance measure P: percentage of correct identification on some set of testing data.
- Training experience E: Scores and class labels provided by classifiers on some set of training data.

Since the problem of combination of classifiers is a special case of machine learning problems, many methods widely used in machine learning can also be applied to this problem. These include: neural networks (Granger, 2001; Giacinto *et al.*, 2003; Shi *et al.*, 2005), support vector machines (Sehgal *et al.*, 2004; Solewicz and Koppel, 2004; Czyz *et al.*, 2004), genetic algorithm (Kuncheva and Jain, 2000; Minaei-Bidgoli *et al.*, 2004), Dempster-Shafer evidence

theory (Xu *et al.*, 1992; Hégarat-Mascle *et al.*, 1997), fuzzy theory (Tahani and Keller, 1990; Cho and Kim, 1995; Chatzis *et al.*, 1999) and decision templates (Kuncheva *et al.*, 2001) etc.

Compared with other machine learning problems, the classifier combination problem has its own advantages and disadvantages. The scores which are generated by this classifier provide much information about the final class. For example, the numbers in Table 4.1 are not random. They have some intrinsic structure. In these five tables, we can see that if the input vector belongs to class 1, then in most cases the numbers in the row of 'f_m^1(X)' are greater that numbers in the row of '$f_m^2(X)$', and vice versa. Such a clear structure supports the assumption that both of the two classifiers are 'good' classifiers, e.g., classifiers which provide useful information to discriminate classes. But for many other machine learning problems, this well-defined structure does not exist.

However, in most cases, the training data for classifier combination are very limited. Consider the case of audio-visual person identification. It is unrealistic to record video files of each person many times. From the recorded video files, we need to choose a large portion of them to train the speaker identification and face identification system. Only a small portion can be used for training the combination algorithm. In other words, the combination algorithm needs to work well in situations where training data are sparse.

The lack of training data is one of the greatest problems for classifier combination. In most cases, it will prevent the implementation of some data-consuming algorithms (e.g., Probability estimation, neural networks, genetic algorithms, etc.). In other words, parameters in the combination algorithm can not possess a very large number because of the phenomenon of 'curse of dimensionality' (Bishop, 1995, chap.2).

From the above discussions, we can set up the criteria of a good combination algorithm. First, the algorithm needs to achieve good performance in situations where training data are sparse; second, the algorithm needs to retain the specific information provided by each classifier.

4.3 Three Levels of Classifier Combination

Broadly speaking, the discriminant functions for a classifier can be classified as one of the following levels (Brunelli and Falavigna, 1995):

(1) *The measurement level*: Each discriminant function measures the confidence that the input signal belongs to one specific class. The probability that the input signal belongs to each class is a confidence measure.

$$f^k(X) = P(\omega_k|X),\ k = 1, 2, ..., K \tag{4.5}$$

Many classifiers are based on probability measures. For example, the GMM probability density function which is used in Chapter 2 belongs to this category.

Another frequently used measurement is distance. When the distance from the input signal to the centre of some class has a large value, the possibility that the input signal belongs to that class is low; while a small distance indicates the reverse. We can define the discriminant functions as the negative of the distance:

$$f^k(x) = -d(x, x_k), \ k = 1, 2, ..., K \tag{4.6}$$

Here x_k represents the centre of the kth class. The minus sign is used to keep the discriminant functions consistent with decision rule (4.1). The distance measure is widely applied in many algorithms. For example, the nearest neighbour (NN) rule will output a distance measure between the input signal and the training data which are nearest to it.

(2) *The rank level*: The value of each discriminant function is fixed to an integer from 1 to K, sorted by increasing confidence.

$$\{f^1(X), f^2(X), ..., f^k(X)\} = \{1, 2, ..., K\} \tag{4.7}$$

That is, if $f^{(i)}(X) = K$ for some $i \in \{1, 2, ..., K\}$, the possibility that X belongs to ω_i takes the highest value; if $f^i(x) = 1$ for some $i \in \{1, 2, ..., K\}$, the possibility takes the lowest value.

(3) *The abstract level*: The value of each discriminant function $f^k(x_m)$ is 0 or 1, and there is only one 1 for the set of discriminant functions $\{f^1(x), f^2(x), ..., f^K(x)\}$, and all others are 0. In another word, the classifier only indicates the classification result, and no other information is provided.

It is easy to see that the measurement level can be converted to the rank level by increasingly sorting the values of the discriminant functions; and the rank level can be converted to the abstract level by setting the discrimination function which equals to K to 1, and all others to 0. As a point of view in information theory (Cover and Thomas, 1991), we can say that the measurement level provides the greatest amount of information while the abstract level provides the smallest amount of information.

Because the measurement level contains the greatest amount of information, it is more suitable for developing and testing combination algorithms. In this chapter, we will concentrate on measurement-level combination.

4.4 Bayesian Approach for Combining Classifiers

Bayesian theory indicates a theoretically optimal rule for combination of classifiers. If we can calculate the posterior probability $P(\omega_k|X)$ for each $k = 1, 2, ..., K$, then we can easily write the decision rule according to Bayesian theory: (Refer to Webb 2002 for a general introduction to Bayesian theory.)

$$\text{decide } X \in \omega_s \text{ if } s = \arg\max_{k=1}^{K} P(\omega_k|X)$$

Because $P(\omega_k|X) = \frac{P(X|\omega_k)P(\omega_k)}{P(X)}$ and $P(X)$ is a constant for different ω_k, the above decision rule can be rewritten as

$$\text{decide } X \in \omega_s \text{ if } s = \arg\max_{k=1}^{K} [P(X|\omega_k)P(\omega_k)] \tag{4.8}$$

Bayesian theory leads to an approach of probability estimation. If we can correctly estimate the *a priori* probability $P(X|\omega_k)$ and $P(\omega_k)$ by using the training data, then we can obtain an optimal classifier. However, such a probability estimation task is impossible both theoretically and practically. Vapnik (1998, chap. 2) has proved that there does not exist a uniform convergent algorithm to estimate probability density functions. In Chapter 5, we will prove the 'No Panacea Theorem' for classifier combination, which states that any combination algorithm will perform badly in some probability distributions no matter how many training data are provided. Thus, we can not guarantee we have obtained the true probability density distributions no matter what estimation algorithm we use. In addition, no probability density estimation method can achieve good results when very sparse training data are provided, but we have stated in Section 4.2 that lack of training data is one of the intrinsic problems for classifier combination.

The problem of sparse data in probability estimation can be alleviated if all classifiers are independent (e.g., a face identification classifier and a speaker identification classifier can be regarded as independent because faces are strongly uncorrelated with voices). Suppose the input vector X is a combination of M feature vectors, that is $X = \{x_1, x_2, ..., x_M\}$. Each of the M feature vectors is extracted from one of the classifiers. For example, in audio-visual person identification, we can regard x_1 as the face feature, and x_2 as the voice feature. If all of the M classifiers are independent of each other, the combined probability density function is a multiple of the probability density functions of these classifiers:

$$P(\omega_k|X) = \prod_{i=1}^{M} P(\omega_k|x_i) \tag{4.9}$$

Notice that $P(\omega_k|x_i) = \frac{P(x_i|\omega_k)P(\omega_k)}{P(x_i)}$. By some deduction, we can obtain the following decision rule:

$$\text{decide } x \in \omega_s \text{ if } s = \arg\max_{k=1}^{K} \left[P(\omega_k)^M \left(\prod_{i=1}^{M} P(x_i|\omega_k) \right) \right] \tag{4.10}$$

Because the dimensionality of x_i is smaller than the full vector X, the probability estimation task becomes easier. However, in most cases, such a reduction is not enough to solve the 'curse of dimensionality' problem.

From the above discussion, we can see that the Baysian approach is optimal provided the probability density functions in equation (4.10) are correctly estimated. However, such an optimality can not be achieved either theoretically or practically.

4.5 Combination with Simple Rules

As shown in equation (4.3), generally, each of the combination functions has $M \times K$ free parameters. Due to the problem of the 'curse of dimensionality', in situations where the training data is sparse, we can not find an accurate combination algorithm with so many free parameters. One way to relieve this problem is to reduce arbitrarily the free parameters in equation (4.3). Many researchers assume the simplified combination functions take the following form:

$$\begin{aligned} f^1_{\text{comb}}(X) &= F_1\left(f^1_1(X), f^1_2(X), \ldots, f^1_M(X)\right) \\ f^2_{\text{comb}}(X) &= F_2\left(f^2_1(X), f^2_2(X), \ldots, f^2_M(X)\right) \\ &\vdots \\ f^K_{\text{comb}}(X) &= F_K\left(f^K_1(X), f^K_2(X), \ldots, f^K_M(X)\right) \end{aligned} \tag{4.11}$$

That is, the Kth combination function is only correlated with the Kth discriminant functions of the M classifiers. Thus the free parameters of each of the combination functions are reduced from $M \times K$ to M. The reason for this assumption is that, for example, if an input vector X belongs to the kth class, which means it is very possible that $f^k_1, f^k_2, \ldots, f^k_M$ take bigger values than other scores. It is easy to generate a combination function which combines these big values to obtain a final big value (e.g., the combination function is the sum of these scores).

Some simple combination functions do not require the training process. They are called fixed rules (Kittler *et al.*, 1998a; Duin, 2002). The most frequently used fixed rules are listed as follows.

1. Product rule: For product rule, the combined combination function is the product of the discriminant functions.

$$\begin{aligned} f^k_{\text{comb}}(X) &= F_k(f^k_1(x_1), f^k_2(x_2), \ldots, f^k_M(x_M)) \\ &= \prod_{m=1}^{M} f^k_m(x_m) \end{aligned} \tag{4.12}$$

From equation (4.10), we can see that the product rule is the optimal solution to the problem of classifier combination if (1) the discriminant functions can accurately measure the probability that the input vector belongs to each class (equation 4.5); (2) all classifiers are independent of each other (equation 4.9); (3) each class has equal probability of occurring, that is $P(\omega_1) = P(\omega_2) = \ldots = P(\omega_K)$.

2. Sum rule: The combination function is the sum of scores which are generated by each classifier.

$$\begin{aligned} f^k_{\text{comb}}(X) &= F_k(f^k_1(x_1), f^k_2(x_2), \ldots, f^k_M(x_M)) \\ &= \sum_{m=1}^{M} f^k_m(x_m) \end{aligned} \tag{4.13}$$

Kittler *et al.* (1998a) indicate that the sum rule is more robust to noise than the product rule.

3. Maximum rule: The maximum rule is described as follows:

$$\begin{aligned} f^k_{\text{comb}}(X) &= F_k(f^k_1(x_1), f^k_2(x_2), ..., f^k_M(x_M)) \\ &= \max^M_{m=1}\left[f^k_m(x_m)\right] \end{aligned} \tag{4.14}$$

The idea of the maximum rule is to select the classifier that is most confident of itself. Normally all $f^k_m(x_m)$ need to be re-scaled to the same range to make the output of all the M classifiers comparable.

4. Minimum rule: The minimum rule is just opposite to the maximum rule. It takes the minimum of top rank scores for each classifer:

$$\begin{aligned} f^k_{\text{comb}}(X) &= F_k(f^k_1(x_1), f^k_2(x_2), ..., f^k_M(x_M)) \\ &= \min^M_{m=1}\left[f^k_m(x_m)\right] \end{aligned} \tag{4.15}$$

The minimum rule will select the outcome of the classifier that has the least objection against a certain class. Similar to maximum rule, a normalisation procedure also needs to be carried out.

5. Median rule: The median rule is similar to the sum rule, but may yield more robust results because the median value is not sensitive to large errors of each classifier.

$$\begin{aligned} f^k_{\text{comb}}(X) &= F_k(f^k_1(x_1), f^k_2(x_2), ..., f^k_M(x_M)) \\ &= \text{median}^M_{m=1}\left[f^k_m(x_m)\right] \end{aligned} \tag{4.16}$$

The reason why we use fixed rules lies in three parts. First, fixed rules do not require training data. They can be implemented in situations when training data are hard to obtain. Second, fixed rules are very simple. This saves time for computation. Third, fixed rules work well in some situations. There are many examples to show that classifiers combined by these fixed rules are better than each classifier individually (Kittler *et al.*, 1998a; Ben-Yacoub *et al.*, 1999; Duin, 2002).

However, fixed rules are sub-optimal (Duin, 2002). They may be misleading if some of the classifiers perform poorly. For designing an optimal decision rule, we need to evaluate each classifier by the training data.

4.6 Combination of Classifiers by the Weighted Sum Rule

A parameter optimisation approach is proposed as another method for combination of classifiers. This approach firstly assumes the combination functions have a form involving some parameters, then uses training data to optimise these parameters.

A well-known parameter optimisation rule is the weighted sum rule, which is described as:

$$\begin{aligned} f^k_{\text{comb}}(X) &= F_k(f^k_1(x_1), f^k_2(x_2), ..., f^k_M(x_M)) \\ &= \sum_{m=1}^{M} \left[a_m f^k_m(x_m)\right], \ (k = 1, 2, ..., K) \end{aligned} \tag{4.17}$$

with the constraint that $\sum_{m=1}^{M} a_m = 1$.

The main difference between the weighted sum rule and the fixed sum rule in (4.13) is that the weighted sum rule allocates a weighting parameter a_m to each discriminant function $f^k_m(X)$. These weighting parameter(s) are optimised by applying a training algorithm.

The weighting parameters should be selected according to the relative reliability of the M classifiers. One way to do this is to optimise α so as to maximise the identification rate on some training data (Maison *et al.*, 1999; Duin, 2002), but this carries the danger of over-fitting, so reducing the ability to generalise to unseen test data.

Several methods can be used to prevent over-fitting. For example, Ney (1995) used the smoothed error rate as the cost function for optimising the weighting parameter(s), and Brunelli and Falavigna (1995) used the normalised ratio of the first- to the second-best integrated score to calculate the weighting parameter(s). Ueda (2000) used an algorithm to optimise the weighting parameter(s) based on minimising the mean square error (MSE). In the next section, we propose a new method for accurately choosing the weighting parameter(s) that directly minimises the estimated correct identification rate.

4.7 New Method for Optimising Weighting Parameter(s)

This section will propose a new method for optimising weighting parameter(s) for the weighted sum rule. Contrary to the methods which were proposed by other researchers (Brunelli and Falavigna, 1995; Ney, 1995; Ueda, 2000), we will firstly estimate the correct identification rate from score distributions, then directly maximise the estimated correct identification rate.

4.7.1 Theoretical Development

Suppose each of the audio and video classifiers consists of K discriminant functions, $f^1(X), f^2(X),\ldots,f^K(X)$. The decision rule in terms of discriminant functions is:

$$\text{decide } X \in \omega_s \text{ if } s = \arg\max_i f^i(X) \tag{4.18}$$

Here, we denote by $f^1_1(X), f^2_1(X), \ldots, f^K_1(X)$ the scores obtained from the video classifier (face identification), and by $f^1_2(X), f^2_2(X), \ldots, f^K_2(X)$ the scores obtained from the audio

classifier. Because only two classifiers are provided, equation (4.17) for describing the weighted sum rule can be rewritten as:

$$f^k_{\text{comb}}(X,\alpha) = \alpha f^k_1(X) + (1-\alpha) f^k_2(X) \qquad k = 1, 2, \ldots, K \tag{4.19}$$

The decision rule based on the specific α is as follows:

$$\text{decide } X \in \omega_s \text{ if } s = \arg\max_i f^i_{\text{comb}}(X) \tag{4.20}$$

In what follows, however, we simplify the notation for discriminant functions by dropping arguments X and α, *except* when it is necessary to distinguish among different values of these arguments.

The first step of our method is to normalise the scores of the training data. We use the so called z-score normalisation, which is calculated using the arithmetic mean and standard deviation of the given data. Refer to Jain *et al.* (2005) for an overview of score normalisation techniques in multimodal biometric systems.

The z-score normalisation process can be divided into two steps. In the first step, all scores of both audio and video classifiers have their mean subtracted and the result is then divided by their variance:

$$\overline{f^k_m(X_i)} = \frac{f^k_m(X_i) - \mu_m}{\sigma_m} \quad : \quad \text{with } \mu_m = \frac{\sum_{i=1}^{I}\sum_{k=1}^{K} f^k_m(X_i)}{I \times K} \quad \text{and } \sigma_m = \frac{\sum_{i=1}^{I}\sum_{k=1}^{K} \left(f^k_m(X_i) - \mu_m\right)^2}{I \times K} \tag{4.21}$$

Here, I is the number of training data points, K is the number of classes, and $m \in \{1, 2\}$.

The second step of normalisation is to make the correct score (i.e., that for the correct person) zero. This gives us a known reference point from which to assess scores, and simplifies the derivation of an appropriate mathematical model under Gaussian assumptions—see below. If we set the weighting parameter α to a constant value, we can obtain the combined scores $\overline{f^1_{\text{comb}}}, \overline{f^2_{\text{comb}}}, \ldots, \overline{f^K_{\text{comb}}}$ by equations (4.19) and (4.21). The second step of the normalisation process is:

$$\text{if } X \in w_i \text{ then } F^k_{\text{comb}} = \overline{f^k_{\text{comb}}} - \overline{f^i_{\text{comb}}} \qquad k = 1, 2, \ldots, K \tag{4.22}$$

Equation (4.22) is used to make the correct score zero. We can see from the decision rule, equation (4.18), that these two steps of normalisation do not change the identification result because the new scores in (4.22) are obtained only by subtracting and dividing the same number from the original scores, which does not influence the rank of the scores.

After normalisation, the next step is to estimate the probability distribution of the scores. We assume that the values of the score functions are independent. That is:

$$P\left(F^1_{\text{comb}}, \ldots, F^{i-1}_{\text{comb}}, F^{i+1}_{\text{comb}}, \ldots, F^K_{\text{comb}} | X \in w_i\right) = \prod_{k=1, k\neq i}^{K} P\left(F^k_{\text{comb}} | X \in w_i\right)$$

The reason why $k \neq i$ is that, after the normalisation, F^i_{comb} always equals zero if $X \in w_i$. We denote the correct identification rate (the probability of correct identification) when $X \in w_i$ as $C_i(\alpha)$. Since $F^i_{\text{comb}} \equiv 0$ when $X \in w_i$ after the normalisation process, we can calculate $C_i(\alpha)$ on the basis of equation (4.18) as:

$$C_i(\alpha) = \prod_{k=1, k\neq i}^{K} P\left(F^k_{\text{comb}} < 0 | X \in w_i\right) \tag{4.23}$$

4.7.2 Probability Density Estimation

To calculate the probability $P\left(F^k_{\text{comb}} < 0 | X \in w_i\right)$ for each $k = 1, 2, \ldots, K$, we first have to estimate the probability distribution $P\left(F^k_{\text{comb}} | X \in w_i\right)$ from the training data in the form of a Gaussian mixture model. But a problem of sparse data arises when we try to model the distribution this way. In essence, it is hard to estimate the density of a multi-modal data distribution reliably.

Our approach to this problem is to break the available training data up into 'sections', and to treat each of these as a unimodal Gaussian, and then to combine them. Suppose there are M training data available for deciding the weighting parameter α. Among these M files, there are M_1 files belonging to class ω_1, M_2 files belonging to class ω_2, ..., and finally M_K files belonging to class ω_K ($M_1 + M_2 + \ldots + M_K = M$).

We denote the M_i training data belonging to class ω_i as $X_1, X_2, \ldots, X_{M_i}$. The Gaussian mixture is then:

$$P\left(F^k_{\text{comb}} | X \in w_i\right) = \frac{1}{M_i} \sum_{j=1}^{M_i} \frac{1}{\sqrt{2\pi}A} \exp\left(-\frac{(F^k_{\text{comb}} - \mu_{kj})^2}{2A^2}\right) \tag{4.24}$$

where A is a parameter controlling the variance(s).

The component means μ_{kj} are obtained as $\mu_{kj} = F^k_{\text{comb}}(X_j, \alpha)$, $j = 1, 2, \ldots, M_i$. From this, we see that the means of the mixture components are the scores of the training data. When A is large, the variance of each mixture component is large; when it is small, the variance is small. In the extreme case when A becomes zero, the probability density shrinks to a series of impulse functions.

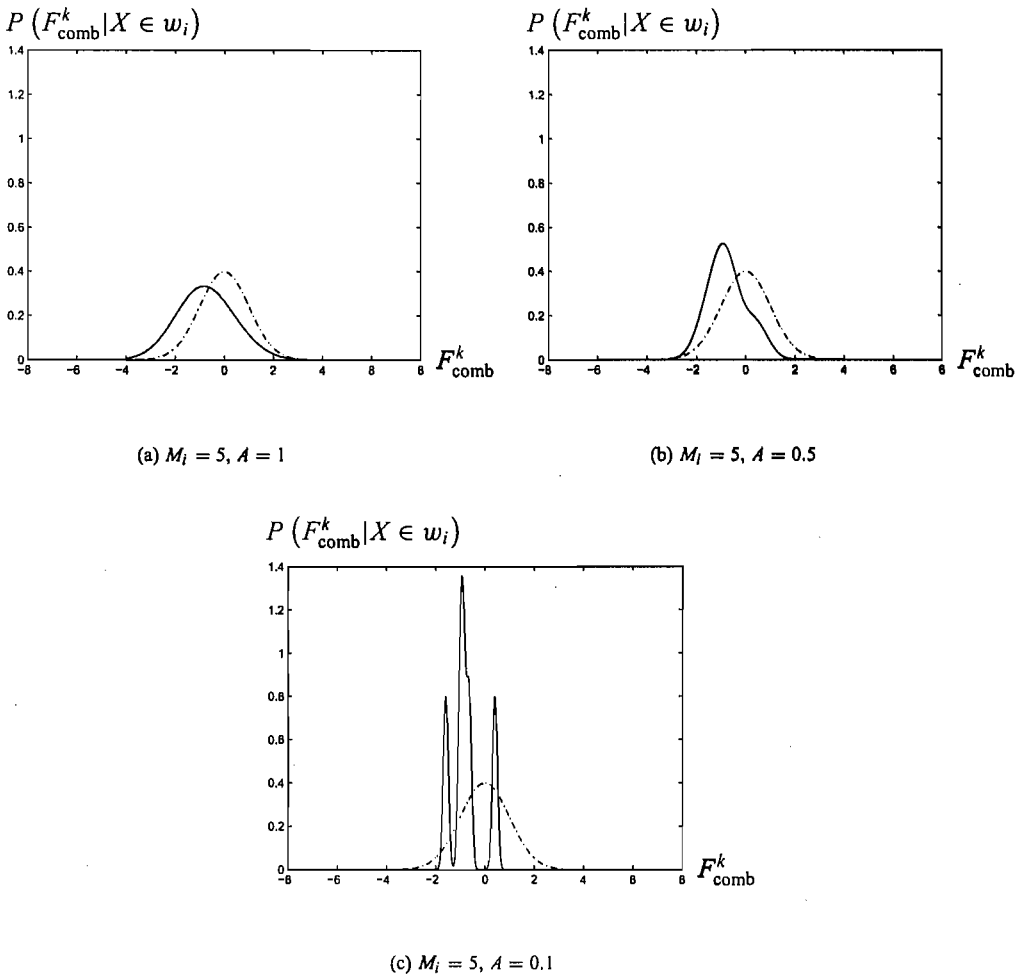
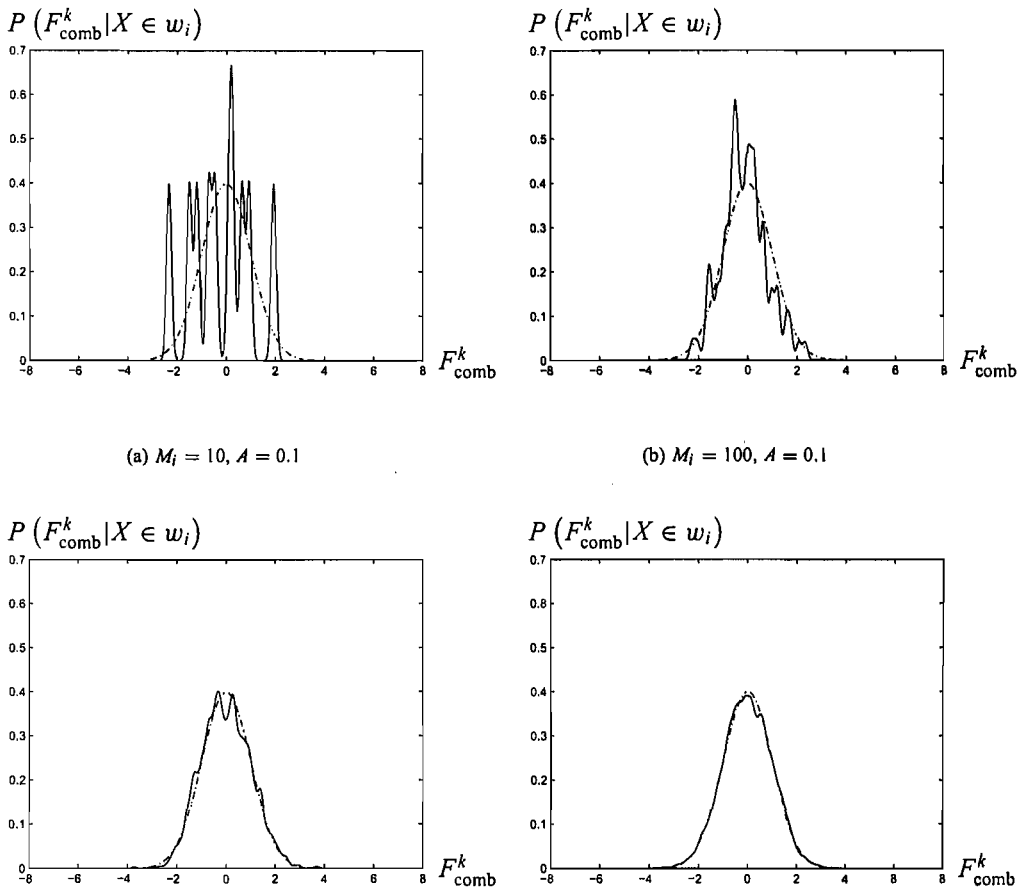


FIGURE 4.1: Probability density estimation using equation (4.24). The distribution to be estimated is a Gaussian distribution with zero mean and standard deviation of one (as indicated by the dashed lines in these three graphs). Five data points are drawn from this distribution ($M_i = 5$). Using equation (4.24), we can obtain the estimated distributions (solid lines) with (a) $A = 1$, (b) $A = 0.5$ and (c) $A = 0.1$, respectively.

Figure 4.1 demonstrates an example of estimating probability density functions using equation (4.24). The probability density function to be estimated is a Gaussian distribution with zero mean and standard deviation of one. It can be seen that in this specific example, the true density function is better estimated when A has a greater value, but this is not always correct. Other distributions may favour smaller rather than greater A. To estimate the probability density distribution using equation (4.24) with finite data, we have to choose a suitable value of A and it is not clear how this should be done.

However, Bishop (1995, pp. 54–55) proves that when the data are infinite, the expectation of the estimated probability density using the above method will converge to the true probability density. This property holds for all values of A. Figure 4.2 demonstrates the convergence procedure when the number of data points increases.



(a) $M_i = 10$, $A = 0.1$

(b) $M_i = 100$, $A = 0.1$

(c) $M_i = 1000$, $A = 0.1$

(d) $M_i = 10000$, $A = 0.1$

FIGURE 4.2: An example to illustrate how the estimated density function (solid lines) reaches the true density function (dashed lines) when increasing M_i. A is fixed to 0.1 and $M_i = 10, 100, 1000, 10000$ respectively.

4.7.3 Estimating Correct Identification Rate

Using the estimated pdf, we can now calculate the probability that $F^k_{\text{comb}}(X) < 0$ as:

$$
\begin{aligned}
&P\left(F^k_{\text{comb}}(X,\alpha) < 0 \,|\, X \in w_i\right) \\
&\quad = \frac{1}{M_i}\sum_{j=1}^{M_i} \frac{1}{\sqrt{2\pi}A}\int_{-\infty}^{0} \exp\left(-\frac{(F^k_{\text{comb}}(X,\alpha)-\mu_{kj})^2}{2A^2}\right) \mathrm{d}\left[F^k_{\text{comb}}(X,\alpha)\right] \\
&\quad = \frac{1}{M_i}\sum_{j=1}^{M_i} \Phi\left(-\frac{\mu_{kj}}{A}\right)
\end{aligned}
$$

where $\Phi(x)$ is the integral of the Gaussian distribution:

$$\Phi(x) = \int_{-\infty}^{x} \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{x^2}{2}\right) \mathrm{d}x$$

From equation (4.23), we can finally obtain $C_i(\alpha)$, which is the correct identification rate for a specified α when $X \in w_i$, as:

$$C_i(\alpha) = \frac{1}{M_i^{K-1}} \prod_{k=1, k\neq i}^{K} \left(\sum_{j=1}^{M_i} \Phi\left(-\frac{\mu_{kj}}{A}\right) \right)$$

The overall correct identification rate, denoted $C(\alpha)$, is given as:

$$C(\alpha) = \sum_{i=1}^{K} C_i(\alpha) P(X \in w_i) \tag{4.25}$$

where $P(X \in w_i)$ can be estimated as $\frac{M_i}{M}$ with M_i equal to the number of training data points which belong to class ω_i, and M equal to the total number of training data points. Thus, we have transformed the problem of choosing weighting parameter α for combining two classifiers to a problem of maximising the correct identification rate $C(\alpha)$.

$$\text{decide } \alpha = \alpha_{\text{opt}} \text{ if } \alpha = \arg\max_{\alpha} C(\alpha)$$

Once the weighting parameter α is selected using our proposed method, we assume it does not change when it is applied to the test data. Such an assumption is based on the more general one that the training data and the test data are independently drawn from the same probability distribution. However, this assumption may not hold in practice, especially when unexpected environmental noise has dramatically changed the probability distribution of the test data. In this situation, it is preferable to use adaptive methods to adjust the weighting parameter(s) (Wark *et al.*, 1999; Wark, 2000; Sanderson and Paliwal, 2003). We still assume that the probability distribution of the training data and test data is the same in this chapter, because, firstly, the XM2VTS database does not contain sufficient data to learn and test adaptive methods for combining classifiers; secondly, one of the main contributions of this thesis is to provide a method to accurately estimate the weighting parameter(s) for classifier combination, and we need to use the scores generated by the audio and visual classifier to test this method.

4.8 Results Using Real Data

We use the first and fifth disks of the XM2VTS database to test the performance of the proposed method. As indicated by Section 2.4.5, the first disk contains speech files of 75 subjects (24

files for each person, which are recorded in 4 sessions, 6 files for each session). The fifth disk contains static face images for these 75 subjects (8 images for each person, which are taken in 4 sessions, 2 files for each session). We will test the proposed combination algorithm on these 75 subjects.

For each test speech file, we randomly select three files which are not from the same session as the training set, then test this file with the trained GMM model. This train-3/test-1 strategy is applied to the 24 files for each speaker and 24 identification results are obtained.

In building the face classifier, for each image, we use the 6 images in the other 3 sessions as the training set, and then test that image. Such a strategy is applied to all 8 images of each person, obtaining 8 identification results. For each of the 4 sessions, we randomly select 2 of the 6 speaker identification results and then combine them with the 2 face identification results for that session.

Figure 4.3 shows the empirical correct identification rate $C_e(\alpha)$ as α increases from 0 to 1, with a 0.01 increment on each trial. This is done by first determining the individual scores of both the audio and video classifier, then calculating the combined scores using equation (4.19), and finally using these for identification. Let us first define the indicator function $T_k(X_i)$ as 1 when $X_i \in \omega_k$; and 0 when $X_i \notin \omega_k$.

$$T_k(X_i) = \begin{cases} 1 & X_i \in \omega_k \\ 0 & \text{otherwise} \end{cases} \tag{4.26}$$

Then $C_e(\alpha)$ is defined as follows:

$$C_e(\alpha) = \frac{1}{M}\sum_{i=1}^{M} T_{\hat{k}}(X_i), \text{ where } \hat{k} = \arg\max_{k=1}^{K} f_{\text{comb}}^k(X_i, \alpha) \tag{4.27}$$

The identification rate for the video classifier is 81.93%, and for the audio classifier it is 90.37%. The combined classifier achieves the highest identification rate (98.31%) when α equals 0.22. However, the empirical identification rate $C_e(\alpha)$ is not a very suitable function to determine the weighting parameter α because of its non-smooth nature, making it difficult to identify a clear peak corresponding to the optimum.

We can also obtain a similar curve $C_{\text{prop}}(\alpha)$ by estimating the correct identification rate, as proposed in Section 4.7. Because for the proposed method the scores for both classifiers are normalised by equation (4.21), some adjustments need to be done to eliminate the effect of normalisation. Recall equation (4.19), the combination function, is as follows.

$$f_{\text{comb}}^k(X, \alpha) = \alpha f_1^k(X) + (1-\alpha) f_2^k(X) \qquad k = 1, 2, \ldots, K$$

If we replace the original scores $f_1^k(X)$ and $f_2^k(X)$ with the normalised scores $\overline{f_1^k(X)} = \frac{f_1^k(X)-\mu_1}{\sigma_1}$ and $\overline{f_2^k(X)} = \frac{f_2^k(X)-\mu_2}{\sigma_2}$, the weighting parameter α also needs to be changed correspondingly to

FIGURE 4.3: The empirical correct identification rate using the test data, with α varying from 0 to 1.

obtain the same effect. Suppose α is changed to α', equation (4.19) can be rewritten as follows:

$$\begin{aligned}\overline{f^k_{\text{comb}}(X,\alpha')} &= \alpha'\overline{f^k_1(X)} + (1-\alpha')\overline{f^k_2(X)} \\ &= \alpha'\frac{f^k_1(X)-\mu_1}{\sigma_1} + (1-\alpha')\frac{f^k_2(X)-\mu_2}{\sigma_2} \qquad k = 1,2,\ldots,K\end{aligned}$$

To obtain the same effect, we must have:

$$\frac{\alpha}{1-\alpha} = \frac{\frac{\alpha'}{\sigma_1}}{\frac{1-\alpha'}{\sigma_2}}$$

Thus, we obtain

$$\alpha' = \frac{\sigma_1\alpha}{\sigma_2 + (\sigma_1 - \sigma_2)\,\alpha}$$

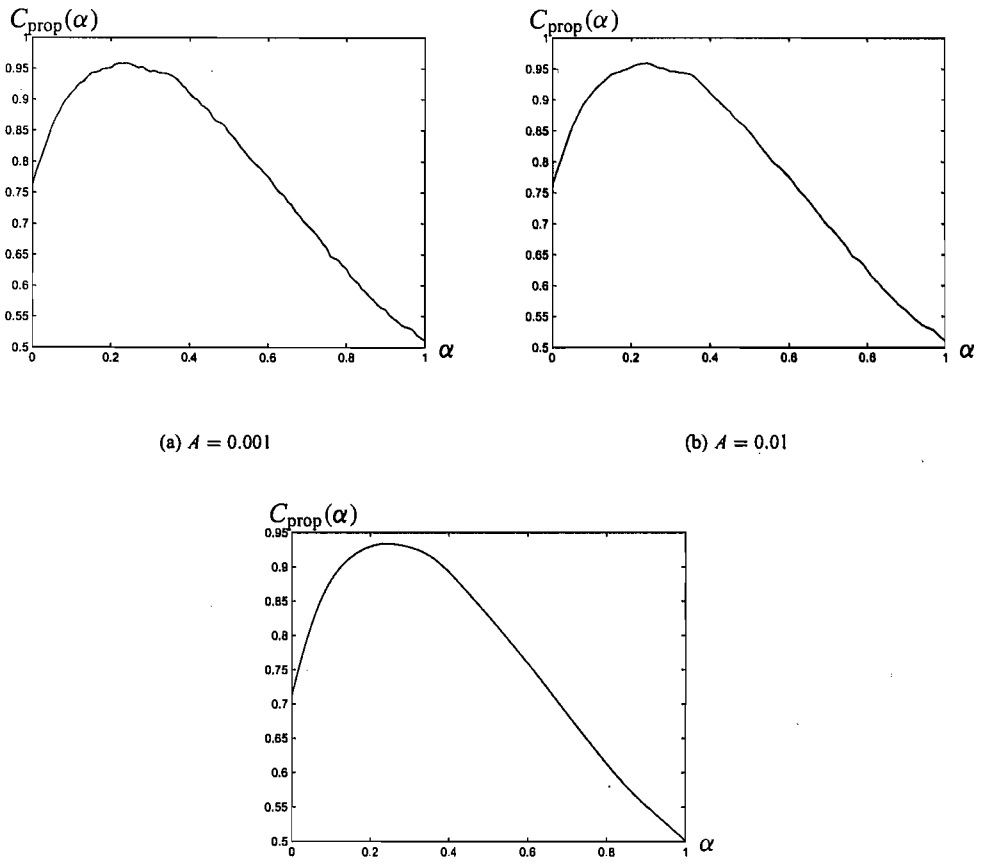
In order to make $C_{\text{prop}}(\alpha)$ comparable to $C_e(\alpha)$, we must define $C_{\text{prop}}(\alpha)$ as follows:

$$C_{\text{prop}}(\alpha) = C(\alpha') \qquad \alpha' = \frac{\sigma_1\alpha}{\sigma_2 + (\sigma_1 - \sigma_2)\,\alpha}$$

where $C(\alpha')$ is defined as in equation (4.25).

Figure 4.4 illustrates the obtained correct identification curves when $A = 0.001$, $A = 0.01$ and $A = 0.1$. When A takes a relatively large value, the curve is smoothed relative to the empirical correct identification curve, and the peak of $C_{\text{prop}}(\alpha)$ when α varies from 0 to 1 can be more clearly observed.

We can see from Figure 4.4 that the estimated correct identification rate is always smaller than the true correct identification rate. This is because the estimated score distribution as in



(a) $A = 0.001$

(b) $A = 0.01$

(c) $A = 0.1$

FIGURE 4.4: The correct identification rate $C_{prop}(\alpha)$ using the proposed method as α varies from 0 to 1: (a) $A = 0.001$; (b) $A = 0.01$: (c) $A = 0.1$.

equation (4.24) does not precisely reflect the true distribution. From Figure 4.5, we can see that even if all the scores for estimation are below 0, the estimated possibility that the score is greater than zero is still 0.02.

That is, in this special case, the estimated identification rate is 2% smaller than the true correct identification rate. If we add up all these score distributions together as in equation (4.23) and (4.25), we can also expect the estimated identification rate is smaller than the true identification rate. We can further estimate that when A becomes larger, the estimated identification rate will be even smaller. That is the reason why the values of $C_{prop}(\alpha)$ are different when A takes different values. But we can see from the above experiments that this problem does not interfere with the process of deciding the weighting parameter α, because we only need to find the α which yields the maximum value of $C_{prop}(\alpha)$.

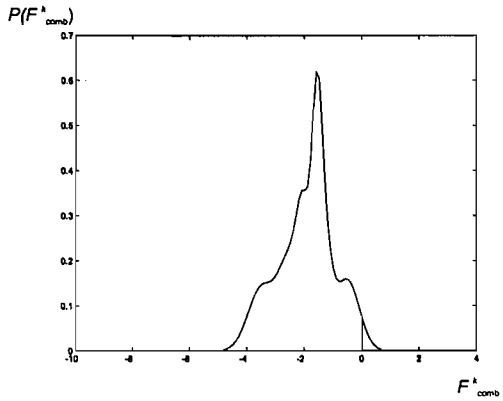


FIGURE 4.5: The estimated probability density of $f^k_{\text{comb}}(X, \alpha)$ when $A = 0.5$ and $f^k_{\text{comb}}(X_1, \alpha) = -3.48$, $f^k_{\text{comb}}(X_2, \alpha) = -2.54$, $f^k_{\text{comb}}(X_3, \alpha) = -2.02$, $f^k_{\text{comb}}(X_4, \alpha) = -1.56$, $f^k_{\text{comb}}(X_5, \alpha) = -1.34$, $f^k_{\text{comb}}(X_6, \alpha) = -0.50$. It indicates that even if all the scores for estimation are below 0, the estimated possibility that the score is greater than 0 is 0.02.

4.9 Further Results Using Bootstrapping

The empirical identification curve in Figure 4.3 shows that the maximal identification rate is achieved when $\alpha = 0.22$, while all the three identification rate curves in Figure 4.4 using the proposed method indicate that the optimal α is 0.24. Our intuition told us that the estimation by the proposed method is more accurate because it provides a smooth curve, thus reducing the possibility of over-fitting. In this section, we use bootstrapping to indicate that, compared with other frequently-used methods, the proposed one for choosing the weighting parameter(s) performs better in reducing the variance of the estimated optimal weighting parameter, thus suggesting a more accurate estimation.

The publication in 1979 of Bradley Efron's first article on bootstrap methods was a major event in statistics, at once synthesising some of the earlier resampling ideas and establishing a new framework for statistical analysis. It has been shown that bootstrap methods often perform better than traditional methods in many applications. The reader is referred to Davison and Hinkley (1997) for a detailed discussion.

The bootstrapping is performed as follows. As indicated in Section 4.8, there are 8 face identification results and 24 speaker identification results for each person. In each bootstrap process, we randomly select 8 speaker identification results out of these 24, then combine them with the 8 face identification results, and obtain estimates of the optimal α by both the empirical and proposed methods. This sampling without replacement process is repeated N times, so obtaining N estimates of the optimal α, one for each process, which are represented as $\alpha^1_{\text{opt}}, \alpha^2_{\text{opt}}, \ldots, \alpha^N_{\text{opt}}$. The mean and variance of α_{opt} can be calculated as follows:

$$\overline{\alpha_{\text{opt}}} = \frac{1}{N}\sum_{i=1}^{N}\alpha_{\text{opt}}^{i}$$

$$\sigma_{\alpha_{\text{opt}}} = \frac{1}{N-1}\sum_{i=1}^{N}\left(\alpha_{\text{opt}}^{i} - \overline{\alpha_{\text{opt}}}\right)^2$$

We have tested four methods for choosing the optimal weighting parameter using the bootstrap method:

1. the empirical method based on actual identification results;
2. the proposed method based on pdf estimation;
3. smoothed error rate estimation; and
4. a genetic algorithm, as proposed by Lam and Suen (1995).

The first two have already been described. The smoothed error rate estimation method was first used by Ney (1995), and subsequently in audio-visual person identification by Maison *et al.* (1999). This method shares some similarities with our method (2) as proposed in this chapter, which is also a smoothing technique for the correct identification curve. Instead of finding the α which maximises $C_e(\alpha)$, the smoothed error rate estimation method finds the α which maximise $C_{\text{smooth}}(\alpha)$ defined as follows:

$$C_{\text{smooth}}(\alpha) = \frac{1}{M}\sum_{i=1}^{M}\sum_{k=1}^{k} T_k(X_i)\frac{\exp\left\{\eta f_{\text{comb}}^{k}(X_i,\alpha)\right\}}{\sum_{j=1}^{K}\exp\left\{\eta f_{\text{comb}}^{j}(X_i,\alpha)\right\}} \tag{4.28}$$

Here M is the total number of training data points. K is the total number of classes and $T_k(X_i)$ has been defined in equation (4.26). We note here that it depends on choosing a parameter η which when large reduces the smoothed error rate to the empirical one.

Lam and Suen's method attempts to find the optimal weighting parameters using genetic algorithm. This method assigns a weighting parameter to each classifier, making the fusion function as follows:

$$f_{\text{comb}}^{k}(X,\alpha_1,\alpha_2) = \alpha_1 f_1^k(X) + \alpha_2 f_2^k(X)$$

Then the fitness function which the genetic algorithm needs to maximise is set as:

$$C_{\text{ga}}(\alpha_1,\alpha_2) = \frac{1}{M}\sum_{i=1}^{M} T_{\hat{k}}(X_i), \text{ where } \hat{k} = \arg\max_{k=1}^{K} f_{\text{comb}}^{k}(X_i,\alpha_1,\alpha_2)$$

Method		$\overline{\alpha_{\text{opt}}}$	$\sigma_{\alpha_{\text{opt}}}$
Empirical		0.2548	0.0545
Genetic Algorithm		0.2685	0.0547
Proposed	$A = 0.001$	0.2478	0.0419
	$A = 0.002$	0.2464	0.0421
	$A = 0.005$	0.2470	0.0425
	$A = 0.01$	0.2480	0.0400
	$A = 0.02$	0.2665	0.0386
	$A = 0.05$	0.2488	0.0358
	$A = 0.1$	0.2635	0.0308
	$A = 0.2$	0.3001	0.0355
Smoothed Error Rate	$\eta = 5$	0.2664	0.0552
	$\eta = 10$	0.3598	0.0388
	$\eta = 15$	0.3051	0.0373
	$\eta = 20$	0.2879	0.0388
	$\eta = 25$	0.2798	0.0411
	$\eta = 30$	0.2766	0.0424
	$\eta = 35$	0.2683	0.0475
	$\eta = 40$	0.2667	0.0486

TABLE 4.2: The means and variances of the four methods for estimating the optimal weighting parameter α_{opt} with the real speech and video data.

and where M is the number of training data points, K is the number of classes, and $T_k(X_i)$ is defined as in equation (4.26).

A genetic algorithm is used to search for the α_1 and α_2 which maximise $C_{\text{ga}}(\alpha_1, \alpha_2)$. Our settings of parameters are slightly different from the original paper. The population size is 20. The fractions of crossover and migration are 0.8 and 0.2 respectively. Because we used the default settings of the Matlab GA function, the reader should refer to the Matlab function `ga(fitnessfcn,nvars)` for the settings of other parameters.

The GA algorithm inspects 100 generations and picks the α_1 and α_2 which yield the largest value of $C_{\text{ga}}(\alpha_1, \alpha_2)$. To make the GA method comparable with other methods, the optimal weighting coefficient α is then found as $\frac{\alpha}{1-\alpha} = \frac{a_1}{a_2}$.

Table 4.2 shows the means $\overline{\alpha_{\text{opt}}}$ and variances $\sigma_{\alpha_{\text{opt}}}$ of these four methods using 200 bootstrap iterations ($N = 200$). We have used a range of A values for the proposed method and, similarly, a range of η values for the smoothed error rate estimation method. Those shown in table are the sub-ranges over which good estimates (i.e., low variances) were obtained.

We can see from the table that the four methods provide similar means. The proposed method and the smoothed error rate method give generally smaller variances than the other two methods (although this is of course achieved with the advantage of an adjustable parameter). The proposed method appears to give a rather smaller variance than the smoothed error rate method, but this is uncertain.

4.10 Results with Simulated Data

Table 4.2 indicates that the proposed method performs slightly better in reducing the estimation variance, but it does not show that this method is also good at reducing the estimation bias, i.e., if $\overline{\alpha_{\text{opt}}}$ estimated by this method is close to the true value of the optimal weighting parameter. With the real data used in the previous section, this question can not be answered because this true value is unknown.

In this section, we try to answer this question in some aspects. First, we construct simulated data with a known probability distribution. So the true value of the optimal weighting parameter can be exactly calculated. Finally, we use the bootstrap method with $N = 200$ to estimate the optimal weighting parameter, and see how close the estimated optimal weighting parameter is to the true value of that parameter.

Consider a 2-classifier and K-class problem. We need to construct $2K$ scores of an input X which belongs to a specific class. For example, we can assume that $X \in \omega_s$. These scores are represented as $f_m^k(X)$, where $m \in \{1, 2\}$ and $k \in \{1, 2, \ldots, K\}$. For constructing $f_m^1(X), f_m^2(X), \ldots, f_m^K(X)$ ($m \in \{1, 2\}$), we first generate K random numbers, each of which is uniformly distributed in the range of $[0, 200]$. We use $n_{m1}, n_{m2}, \ldots, n_{mK}$ to represent these K numbers. We choose n_{ms} as the maximum of these K numbers ($n_{ms} = \max\{n_{m1}, n_{m2}, \ldots, n_{mK}\}$), since it is reasonable to assume that the highest score will be obtained for the correct class. Next we generate another K random numbers, $\sigma_{m1}, \sigma_{m2}, \ldots, \sigma_{mK}$, each of which is uniformly distributed in the range of $[0, \sigma_m^{\max}]$. Here $\sigma_m^{\max}$ is a controlling parameter. The scores $f_m^1(X), f_m^2(X), \ldots, f_m^K(X)$ are generated as follows. For each $m \in \{1, 2\}$ and $k \in \{1, 2, \ldots, K\}$, $f_m^k(X)$ is a random sample drawn from a Gaussian distribution with mean n_{mk} and variance σ_{mk}.

We construct two classifiers, denoted Classifier 1 and Classifier 2. For Classifier 1, we set its $\sigma_1^{\max}$ to 10; and for Classifier 2, we set its $\sigma_2^{\max}$ to 20. Thus, Classifier 1 is a strong classifier and Classifier 2 is a weak classifier. For both, we set K to 74, equal to the number of classes in the audio-visual person identification task. For each class, we generate 8 sets of scores from Classifier 1, and 24 sets of scores from Classifier 2, which is also the same as the audio-visual person identification task. Using the bootstrap method, we then obtain the means and variances of the four methods. Since the simulated data are generated from a known distribution, we can also accurately calculate the true optimal weighting parameter, α_{true}, by using all the parameters n_{mk} and σ_{mk} ($m \in \{1, 2\}$ and $k \in \{1, 2, \ldots, K\}$). For simplicity, the process of how to calculate α_{true} is described in Appendix B. We only mention here that we can accurately calculate α_{true} since the score distributions are known.

Table 4.3 shows the estimated means and variances for normally-distributed simulated data using the four methods. Here, $\alpha_{\text{true}} = 0.650$. It can be observed that the means of the empirical, genetic algorithm and proposed methods are closer to α_{true} than the smoothed error rate method, but the proposed method gives much smaller variance than the empirical and genetic algorithm

Method		$\overline{\alpha_{\text{opt}}}$	$\sigma_{\alpha_{\text{opt}}}$
Empirical		0.6509	0.0428
Genetic Algorithm		0.6510	0.0399
Proposed	$A = 0.001$	0.6494	0.0350
	$A = 0.002$	0.6499	0.0344
	$A = 0.005$	0.6494	0.0303
	$A = 0.01$	0.6520	0.0298
	$A = 0.02$	0.6520	0.0277
	$A = 0.05$	0.6491	0.0180
	$A = 0.1$	0.6185	0.0089
	$A = 0.2$	0.5719	0.0058
Smoothed Error Rate	$\eta = 5$	0.5225	0.0045
	$\eta = 10$	0.5750	0.0047
	$\eta = 15$	0.6070	0.0078
	$\eta = 20$	0.6269	0.0113
	$\eta = 25$	0.6358	0.0124
	$\eta = 30$	0.6436	0.0159
	$\eta = 35$	0.6436	0.0159
	$\eta = 40$	0.6436	0.0159

TABLE 4.3: The means and variances of four methods for estimating α_{opt} on simulated data generated to have a Gaussian distribution. Here $\alpha_{\text{opt}} = 0.650$. The process of how to calculate this α_{true} is described in Appendix B.

methods. However, we need to remember that the simulated data are generated with a Gaussian distribution, so this could be considered an unfair test of our method. Thus, we have also carried out performance comparisons with data with a rectangular distribution. This should show our method at maximum disadvantage relative to the competitors.

As before, the sets of random numbers were generated from which we obtain n_{mk} and σ_{mk} ($m \in \{1, 2\}$ and $k \in \{1, 2, \ldots, K\}$). A uniform distribution in the range $(n_{mk} - \sigma_{mk}, n_{mk} + \sigma_{mk})$ was then generated to simulate the distribution of $f_m^k(X)$. The results in Table 4.4 for the data with uniform distribution show that the proposed method holds up well in the face of violation of the underlying assumption of normally-distributed data. The optimal weighting parameter is estimated with very low bias and low variance, certainly relative to the empirical and GA methods. Performance is slightly but noticeably better than the smoothed error rate method.

It is not suitable to use the empirical method directly to decide the optimal weighting parameter, because it gives very high variance. A better solution is to calculate the average of the optimal weighting parameters by using the bootstrap method. In situations where the training data are sparse, so that it is difficult to use the bootstrap method, the proposed method is highly recommended.

The main drawback of the proposed method is that we have to choose a suitable value of A and it is not clear how this should be done. Of course, the smoothed error rate technique shares this kind of problem, in that we have to fix a suitable value of η.

Method		$\overline{\alpha_{\text{opt}}}$	$\sigma_{\alpha_{\text{opt}}}$
Empirical		0.7324	0.0492
Genetic Algorithm		0.7368	0.0529
Proposed	$A = 0.001$	0.7359	0.0419
	$A = 0.002$	0.7367	0.0413
	$A = 0.005$	0.7346	0.0385
	$A = 0.01$	0.7343	0.0306
	$A = 0.02$	0.7327	0.0220
	$A = 0.05$	0.7270	0.0176
	$A = 0.1$	0.7177	0.0224
	$A = 0.2$	0.7541	0.0519
Smoothed Error Rate	$\eta = 5$	0.6106	0.1486
	$\eta = 10$	0.7043	0.0525
	$\eta = 15$	0.7135	0.0314
	$\eta = 20$	0.7182	0.0255
	$\eta = 25$	0.7224	0.0235
	$\eta = 30$	0.7256	0.0234
	$\eta = 35$	0.7256	0.0234
	$\eta = 40$	0.7256	0.0234

TABLE 4.4: The means and variances of four methods for estimating α_{opt} on simulated data generated with a uniform distribution. Here $\alpha_{\text{opt}} = 0.727$. The process of how to calculate α_{true} are described in Appendix B.

4.11 Summary

In this chapter, we have provided a method to estimate the optimal weighting parameter for fusion of scores in audio-visual person identification. It is based on estimation of probability density functions for the scores under a Gaussian assumption. By use of bootstrapping, the performance of this method can be strictly analysed and compared with other methods. Using simulated data, such that the pdf's are known, results indicate that this method has advantages in reducing the bias and variance of the estimation. The method is shown to perform well even when the underlying Gaussian assumption is violated. The main problem is in choosing a suitable value of smoothing parameter A. It is not clear at present how this should best be done.

The validity of the proposed method are based on two assumptions. First, the bootstrapping method as discussed in Section 4.9 is based on the assumption that the performances of the audio classifier and the visual classifier are independent. Intuitively, such an assumption is true because we have little information to imagine a person's face when only listening to his/her voice, and vice versa. Another assumption is that, as discussed in Section 4.7.3, the training and the test data are drawn independently from a fixed probability distribution. Thus, the optimal weighting parameter remains unchanged. Although this assumption is very common in theoretical pattern recognition studies (Vapnik, 1998), it may not valid in practice. Thus, adaptive methods for choosing weighting parameter(s) are preferable in practical situations.

It should be noted that, although our method is developed for the identification task, it can

be generated to verification. For verification, similar approach can be applied for choosing the optimal weighting parameter based on minimising the equal error rate (EER), instead of maximising the correct identification rate for the identification case. In Chapter 7, we have outlined some future work to generalise this method to person verification.

Chapter 5

Theoretical Study: A 'No Panacea Theorem' for Classifier Combination

As indicated in Chapter 4, it is hard to find an optimal combination algorithm which can perform well in every situation. In this chapter, we theoretically prove that there is no 'perfect' combination algorithm suitable for all situations. Such a property, which is called the 'no panacea' principle by Kuncheva (2004), appears widely acknowledged, but no strict mathematical proof exists for it.

In this Chapter, we introduce a theoretical proof of the 'no panacea' property for classifier combination. We have proved the No Panacea Theorem (NPT), which states that if the combination function is continuous and diverse (to be defined), there exists a situation in which the combination algorithm will give very bad performance. Thus, there is no optimal combination algorithm which is suitable in all situations. Although built for multiple classifier combination, this theorem is also valid for all supervised learning situations.

The No Panacea Theorem for classifier combination can be regarded as a generalisation of the No Free Lunch (NFL) Theorems (Wolpert and Macready, 1995, 1997). Wolpert and Macready (1997) proved that any two optimisation algorithms are equivalent when their performance is averaged across all possible probability density functions. Wolpert (2001) further extended the NFL idea to supervised learning and concluded that the performance of any learning algorithms are the same when averaging over all prior probability distributions. These establish the same average performance of all optimisation and supervised learning algorithms across all possible problems. There has been much work extending and generalising the NFL theorem. The reader is referred to `www.no-free-lunch.org` for details.

However, the NFL theorem only discusses the average performance of algorithms. It does not consider the problem of how good or bad the performance of a specific algorithm would be for a given probability distribution. If there exists a probability distribution which would dictate bad performance for a specified algorithm, what does it look like? This chapter will address

these two problems. We prove that if the combination functions are continuous and diverse, then we can construct probability density functions based on Gaussian mixtures in which the combination algorithm will yield very bad performance.

There has also been some research work on constructing objective functions and probability distributions for theorem proving. Oltean (2004) has explicitly constructed objective functions where random search outperforms evolutionary algorithms. Antos *et al.* (1999) construct probability distributions to prove that there does not exist a universally-superior Bayes error estimation method, no matter how many simulations are performed and how large the sample sizes are. However, the objective functions and probability distributions constructed in these previous works are a little bit 'strange', i.e., they are not likely to be encountered in real-world problems. In this chapter, we will prove the No Panacea Theorem based on constructing probability distributions of Gaussian mixtures. Based on the central limit theorem (Grimmett and Stirzaker, 1992), Gaussian mixtures are good models for many real-world problems.

Another origin of our proof comes from the Chanson theorem (Vapnik, 1998, chap. 2) in statistics, which states that for any estimator $\epsilon_l(A)$ of an unknown probability measure defined on the Borel subsets $A \subset (0, 1)$, there exists a measure P for which $\epsilon_l(A)$ does not provide uniform convergence. Our method to construct the probability density functions in Section 5.2 is very similar to the proof of this theorem.

5.1 Background

Suppose there are M classifiers. The task of each classifier is to assign an input X to one of K classes, $\omega_1, \omega_2, \ldots, \omega_K$. Each classifier generates a set of discriminant functions (or scores), $f^1(X), f^2(X), \ldots, f^K(X)$ respectively. The decision rule in terms of discriminant functions is:

$$\text{decide } X \in \omega_S \text{ if } S = \arg\max_{k=1}^{K} f^k(X)$$

For these M classifiers, we use $f_1^1(X), f_1^2(X), \ldots, f_1^K(X)$ to represent the scores generated by the first classifier, and $f_2^1(X), f_2^2(X), \ldots, f_2^K(X)$ to represent the scores generated by the second classifier, ..., and $f_M^1(X), f_M^2(X), \ldots, f_M^K(X)$ to represent scores generated by the Mth classifier. Thus, we obtain $M \times K$ score functions. For simplicity, we will use $x_1, x_2, \ldots, x_N$ to represent these score functions ($N = M \times K$). If the input has a subscript, such as X_j, we will use $x_{1j}, x_{2j}, \ldots, x_{Nj}$ to represent its scores.

The process of combining these M classifiers can be described as finding a set of combination

functions $F_k(x_1, x_2, \ldots, x_N)$ $(k = 1, 2, \ldots, K)$ with the following decision rule:

$$\text{decide } X \in \omega_S \text{ if } S = \arg\max_{k=1}^{K} F_k(x_1, x_2, \ldots, x_N) \tag{5.1}$$

A combination function divides the whole domain D of all points $\{x_1, x_2, \ldots, x_N\}$ into K regions, denoted $D_1, D_2, \ldots, D_K$:

$$\begin{aligned}
D_1 &= \left\{ \{x_1, x_2, \ldots, x_N\} \;\middle|\; \arg\max_{k=1}^{K} F_k(x_1, x_2, \ldots, x_N) = 1 \right\} \\
D_2 &= \left\{ \{x_1, x_2, \ldots, x_N\} \;\middle|\; \arg\max_{k=1}^{K} F_k(x_1, x_2, \ldots, x_N) = 2 \right\} \\
&\vdots \\
D_K &= \left\{ \{x_1, x_2, \ldots, x_N\} \;\middle|\; \arg\max_{k=1}^{K} F_k(x_1, x_2, \ldots, x_N) = K \right\}
\end{aligned}$$

From decision rule 5.1, we know that D_i $(i = 1, 2, \ldots, K)$ is the region that the combination algorithm regards as encompassing the ith class.

We define the K joint probability density functions of $x_1, x_2, \ldots, x_N$ given the input data as:

$$\begin{aligned}
p_1(x_1, x_2, \ldots, x_N) &= P(x_1, x_2, \ldots, x_N | X \in \omega_1) \\
p_2(x_1, x_2, \ldots, x_N) &= P(x_1, x_2, \ldots, x_N | X \in \omega_2) \\
&\vdots \\
p_K(x_1, x_2, \ldots, x_N) &= P(x_1, x_2, \ldots, x_N | X \in \omega_K)
\end{aligned}$$

Then according to our previous definitions, we can obtain the classification error rate given that the correct class is ω_i $(i = 1, 2, \ldots, K)$ as a function of p_i:

$$\begin{aligned}
P(\text{error}|\omega_i) &= \int_{\widetilde{D_i}} p_i(x_1, x_2, \ldots, x_N)\, dx_1\, dx_2 \cdots dx_N \\
&= 1 - \int_{D_i} p_i(x_1, x_2, \ldots, x_N)\, dx_1\, dx_2 \cdots dx_N
\end{aligned} \tag{5.2}$$

where $\widetilde{D_i}$ is the complement of D_i:

$$\widetilde{D_i} = D_1 \bigcup D_2 \bigcup \ldots \bigcup D_{i-1} \bigcup D_{i+1} \ldots \bigcup D_K$$

It refers to the region in which the classification is incorrect.

Based on these definitions, the total classification error rate can be calculated as follows:

$$\begin{aligned} P(\text{error}) &= \sum_{i=1}^{K} P(\omega_i) P(\text{error}|\omega_i) \\ &= 1 - \sum_{i=1}^{K} P(\omega_i) \int_{D_i} p_i(x_1, x_2, \ldots, x_N)\, dx_1\, dx_2 \cdots dx_N \end{aligned} \tag{5.3}$$

Here $P(\omega_1), P(\omega_2), \ldots, P(\omega_K)$ are the prior probabilities that input data X belongs to classes $\omega_1, \omega_2, \ldots, \omega_K$ respectively.

In order to build the theorem, two assumptions for the combination functions need to be added.

Assumption 1. [Continuous assumption]. For each $k \in \{1, 2, \ldots, K\}$, the combination function $F_k(x_1, x_2, \ldots, x_N)$ is continuous with respect to $x_1, x_2, \ldots, x_N$. More specifically, for any $\epsilon > 0$, there exists a $\delta = \delta(\epsilon) > 0$ such that

$$\begin{aligned} &\text{If } \sqrt{(x_1 - x_{10})^2 + (x_2 - x_{20})^2 + \cdots + (x_N - x_{N0})^2} < \delta, \\ &\text{then } |F_k(x_1, x_2, \ldots, x_N) - F_k(x_{10}, x_{20}, \ldots, x_{N0})| < \epsilon \end{aligned}$$

A useful corollary can be deduced from the continuous assumption which will be used in our proof of the NPT.

Corollary 5.1. *If $F_k(x_1, x_2, \ldots, x_N)$ is continuous for each $k \in \{1, 2, \ldots, K\}$, then for any point X_0 with scores $\{x_{10}, x_{20}, \ldots, x_{N0}\}$, suppose S is the class label which the combination algorithm assigns to X_0:*

$$S = \arg \max_{k=1}^{K} F_k(x_{10}, x_{20}, \ldots, x_{N0}),$$

then there exists an open ball $B(X_0, \delta)$ so that for any point in that open ball, the combination algorithm will assign the same class label S to this point.

Here $B(X_0, \delta)$ refers to the set of points $\{x_1, x_2, \ldots, x_N\}$ which satisfy:

$$\sqrt{(x_1 - x_{10})^2 + (x_2 - x_{20})^2 + \cdots + (x_N - x_{N0})^2} < \delta$$

Assumption 2. [Diverse assumption]. There should be two data points which make the combination functions take values of two different classes. In other words, there exists two classes ω_i and ω_j $(1 \leq i < j \leq K)$ that satisfy the following constraints.

$\exists\{x_{1\omega_i}, x_{2\omega_i}, \ldots, x_{N\omega_i}\}$ such that $i = \arg\max_{k=1}^{K} F_k(x_{1\omega_i}, x_{2\omega_i}, \ldots, x_{N\omega_i})$
$\exists\{x_{1\omega_j}, x_{2\omega_j}, \ldots, x_{N\omega_j}\}$ such that $j = \arg\max_{k=1}^{K} F_k(x_{1\omega_j}, x_{2\omega_j}, \ldots, x_{N\omega_j})$

5.2 No Panacea Theorem

Based on the definitions above, we will prove the No Panacea Theorem in the cases of M classifiers and K classes.

Suppose there are U training data points, $X_1, X_2, \ldots, X_U$. The scores of the jth data point X_j given by the M classifiers are represented as $\{x_{1j}, x_{2j}, \ldots, x_{Nj}\}$. Here $j = 1, 2, \ldots, U$. For these U training data points, we have the following theorem.

Theorem 5.2. *Given the U training points as described above, if the combination functions $F_k(x_1, x_2, \ldots, x_N)$, $(k = 1, 2, \ldots, K)$ satisfy the continuous and diverse assumptions, then there exist K continuous probability density functions $p_1(x_1, x_2, \ldots, x_N), p_2(x_1, x_2, \ldots, x_N), \ldots, p_K(x_1, x_2, \ldots, x_N)$ such that for any given $P > 0$ and any $\epsilon \in (0, 1)$, the following two properties hold:*

1. *For $j = 1, 2, \ldots, U$, if $X_j \in \omega_i$ then $p_i(x_{1j}, x_{2j}, \ldots, x_{Nj}) > P$.*
2. *For all $i = 1, 2, \ldots, k$, $P(error|\omega_i)$ calculated by equation (5.2) is greater than $(1 - \epsilon)$. Thus, the total classification error $P(error)$, which is calculated by equation (5.3), is also greater than $(1 - \epsilon)$.*

For this N-classifier, K-class problem, every combination algorithm needs to generate K combination functions $F_k(x_1, x_2, \ldots, x_N)$ $(k = 1, 2, \ldots, K)$ based on the training data points $X_1, X_2, \ldots, X_U$. But, as can be seen from equations (5.2) and (5.3), the performance of the combination algorithm is not only associated with the function $F(x_1, x_2, \ldots, x_N)$, but also with probability density functions. However, these pdf's can not be completely revealed by finite training data, so for any combination algorithm, there may exist some pdf's which make the performance very bad. Thus, properties (1) and (2) give a criterion for how bad the performance of the combination may be. Property (1) states that there exist pdf's which make the density on the training data very high. Property (2) states that such pdf's also make the error rate very high. Generally, these two properties indicates that for any combination algorithm which satisfies the continuous and diverse assumptions, there exist pdf's which can very possibly generate the training data, but the combination functions may give very poor performance. The main idea of our proof is to generate Gaussian mixture distributions which have high densities in the 'wrong' areas (where the combination functions give incorrect classification).

Proof Because $F_k(x_1, x_2, \ldots, x_N)$ satisfies the diverse assumption, which means that the combination algorithm assigns some points to one class and other points to another class,

without loss of generality, we assume these two classes are ω_1 and ω_2. That is, there exists one point $\{x_{1\omega_1}, x_{2\omega_1}, \ldots, x_{N\omega_1}\} \in \omega_1$ and another $\{x_{1\omega_2}, x_{2\omega_2}, \ldots, x_{N\omega_2}\} \in \omega_2$ such that:

$$\begin{aligned} & \arg\max_{k=1}^{K} F_k(x_{1\omega_1}, x_{2\omega_1}, \ldots, x_{N\omega_1}) = 1 \\ \text{and} \quad & \arg\max_{k=1}^{K} F_k(x_{1\omega_2}, x_{2\omega_2}, \ldots, x_{N\omega_2}) = 2 \end{aligned}$$

Because $F_k(x_1, x_2, \ldots, x_N)$ is continuous, by corollary (5.1), there exist δ_1 and δ_2 that make $B(X_{\omega_1}, \delta_1) \subseteq D_1$ and $B(X_{\omega_2}, \delta_2) \subseteq D_2$.

We further suppose that for the U training points, there are U_1 points belonging to ω_1, U_2 points belonging to $\omega_2, \ldots$, and U_N points to ω_N, where $\sum_{i=1}^{N} U_i = U$.

Based on the above definitions, we can prove the theorem in two steps.

Step 1 We first construct probability density functions $p_1(x_1, x_2, \ldots, x_N)$ and $p_2(x_1, x_2, \ldots, x_N)$, then we prove that the given p_1 and p_2 satisfy properties (1) and (2).

The form of p_1 and p_2 is given as follows:

$$\begin{aligned} p_1(x_1, x_2, \ldots, x_N) &= \left(\frac{U_1}{U_1+1}\epsilon\right) t_{11}(x_1, x_2, \ldots, x_N) + \\ &\quad \left(1 - \frac{U_1}{U_1+1}\epsilon\right) t_{12}(x_1, x_2, \ldots, x_N) \\ p_2(x_1, x_2, \ldots, x_N) &= \left(\frac{U_2}{U_2+1}\epsilon\right) t_{21}(x_1, x_2, \ldots, x_N) + \\ &\quad \left(1 - \frac{U_2}{U_2+1}\epsilon\right) t_{22}(x_1, x_2, \ldots, x_N) \end{aligned}$$

where t_{11} and t_{21} are Gaussian mixture distributions, and t_{12} and t_{22} are Gaussian distributions.

$$\begin{aligned} t_{11}(x_1, x_2, \ldots, x_N) &= \frac{1}{U_1} \frac{1}{\left(\sqrt{2\pi}\sigma_1\right)^N} \sum_{j=1}^{U} T_{\omega_1}(x_1, x_2, \ldots, x_N)\, e^{-\frac{\sum_{i=1}^{N}(x_i - x_{ij})^2}{2\sigma_1^2}} \\ t_{12}(x_1, x_2, \ldots, x_N) &= \frac{1}{\left(\sqrt{2\pi}\sigma_{\omega_2}\right)^N} e^{-\frac{\sum_{i=1}^{N}(x_i - x_{i\omega_2})^2}{2\sigma_{\omega_2}^2}} \\ t_{21}(x_1, x_2, \ldots, x_N) &= \frac{1}{U_2} \frac{1}{\left(\sqrt{2\pi}\sigma_2\right)^N} \sum_{j=1}^{U} T_{\omega_2}(x_1, x_2, \ldots, x_N)\, e^{-\frac{\sum_{i=1}^{N}(x_i - x_{ij})^2}{2\sigma_2^2}} \\ t_{22}(x_1, x_2, \ldots, x_N) &= \frac{1}{\left(\sqrt{2\pi}\sigma_{\omega_1}\right)^N} e^{-\frac{\sum_{i=1}^{N}(x_i - x_{i\omega_1})^2}{2\sigma_{\omega_1}^2}} \end{aligned}$$

Here T_{ω_i} is an indicator function which has the value 0 when $\{x_1, x_2, \ldots, x_N\} \notin \omega_i$, and 1 otherwise. The general form of these indicator functions is:

$$T_{\omega_i}(x_1, x_2, \ldots, x_N) = \begin{cases} 0 & \text{if } \{x_1, x_2, \ldots, x_N\} \notin \omega_i \\ 1 & \text{otherwise} \end{cases} \qquad (i = 1, 2, \ldots, K)$$

where σ_1, σ_2, σ_{ω_1} and σ_{ω_2} are parameters to be decided. In the following, we prove that when σ_1, σ_2, σ_{ω_1} and σ_{ω_2} are sufficiently small, properties (1) and (2) will hold.

We first prove that when σ_1 is sufficiently small, if X_j belongs to ω_1, then $P_1(x_{1j}, x_{2j}, \ldots, x_{Nj})$ will be greater than P.

If $X_j \in \omega_1$, then $T_{\omega_1}(x_{1j}, x_{2j}, \ldots, x_{Nj}) = 1$.

$$\begin{aligned} p_1(x_{1j}, x_{2j}, \ldots, x_{Nj}) &\geq \frac{U_1 \epsilon}{U_1 + 1} t_{11}(x_{1j}, x_{2j}, \ldots, x_{Nj}) \\ &\geq \frac{\epsilon}{\left(\sqrt{2\pi}\sigma_1\right)^N (U_1 + 1)} e^{-\frac{\sum_{i=1}^{N}(x_{ij} - x_{ij})^2}{2\sigma_1^2}} \\ &= \frac{\epsilon}{\left(\sqrt{2\pi}\sigma_1\right)^N (U_1 + 1)} \end{aligned}$$

So if we choose:

$$\sigma_1 < \frac{1}{\sqrt{2\pi}} \left(\frac{\epsilon}{(U_1 + 1)P} \right)^{\frac{1}{N}} \qquad (5.4)$$

we will always have $p_1(x_{1j}, x_{2j}, \ldots, x_{Nj}) > P$.

The same deduction can be used to prove that when

$$\sigma_2 < \frac{1}{\sqrt{2\pi}} \left(\frac{\epsilon}{(U_2 + 1)P} \right)^{\frac{1}{N}} \qquad (5.5)$$

if $X_j \in \omega_2$, then $p_2(x_{1j}, x_{2j}, \ldots, x_{Nj}) > P$. Thus, we have proved property (1).

For property (2), we will prove that when σ_{ω_1} and σ_{ω_2} are sufficiently small, both $P(\text{error}|\omega_1)$ and $P(\text{error}|\omega_2)$ are greater than $(1 - \epsilon)$.

$$
\begin{aligned}
P(\text{error}|\omega_1) &= \int_{\widetilde{D}_1} p_1(x_1, x_2, \ldots, x_N)\, \mathrm{d}x_1\, \mathrm{d}x_2 \cdots \mathrm{d}x_N \\
&\geq \int_{D_2} p_1(x_1, x_2, \ldots, x_N)\, \mathrm{d}x_1\, \mathrm{d}x_2 \cdots \mathrm{d}x_N \\
&\geq \int_{D_2} \left(1 - \frac{U_1}{U_1+1}\epsilon\right) t_{12}(x_1, x_2, \ldots, x_N)\, \mathrm{d}x_1\, \mathrm{d}x_2 \cdots \mathrm{d}x_N
\end{aligned}
$$

Since $B(X_{\omega_2}, \delta_2) \subseteq D_2$, we have:

$$
\begin{aligned}
&P(\text{error} \mid \omega_1) \geq \left(1 - \tfrac{U_1}{U_1+1}\epsilon\right) \int_{B(X_{\omega_2}, \delta_2)} t_{12}(x_1, x_2, \ldots, x_N)\, \mathrm{d}x_1\, \mathrm{d}x_2 \cdots \mathrm{d}x_N \\
&= \left(1 - \tfrac{U_1}{U_1+1}\epsilon\right) \frac{1}{(\sqrt{2\pi}\sigma_{\omega_2})^N} \int_{B(X_{\omega_2}, \delta_2)} e^{-\frac{\sum_{i=1}^{N}(x_i - x_{i\omega_2})^2}{2\sigma_{\omega_2}^2}}\, \mathrm{d}x_1\, \mathrm{d}x_2 \cdots \mathrm{d}x_N \\
&\geq \left(1 - \tfrac{U_1}{U_1+1}\epsilon\right) \prod_{i=1}^{N} \left(\int_{x_{i\omega_2} - \frac{\delta_2}{\sqrt{N}}}^{x_{i\omega_2} + \frac{\delta_2}{\sqrt{N}}} \frac{1}{\sqrt{2\pi}\sigma_{\omega_2}} e^{-\frac{(x_i - x_{i\omega_2})^2}{2\sigma_{\omega_2}^2}}\, \mathrm{d}x_i \right) \\
&= \left(1 - \tfrac{U_1}{U_1+1}\epsilon\right) \left(\int_{-\frac{\delta_2}{\sqrt{N}}}^{\frac{\delta_2}{\sqrt{N}}} e^{-\frac{x^2}{2\sigma_{\omega_2}^2}}\, \mathrm{d}x \right)^N
\end{aligned}
$$

For a Gaussian distribution with mean 0 and variance σ, we have the Chernoff bound (Wilson, 1996) for the integral.

$$
P(-\delta \leq X \leq \delta) = \int_{-\delta}^{\delta} \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{x^2}{2\sigma^2}}\, \mathrm{d}x \geq 1 - 2e^{-\frac{\delta^2}{2\sigma^2}}
$$

Substituting $\sigma = \sigma_{\omega_2}$ and $\delta = \frac{\delta_2}{\sqrt{N}}$ into the above, we finally obtain:

$$
P(\text{error}|\omega_1) \geq \left(1 - \frac{U_1}{U_1+1}\epsilon\right) \left(1 - 2e^{-\frac{\delta_2^2}{2N\sigma_{\omega_2}^2}}\right)^N
$$

So if we choose:

$$
\sigma_{\omega_2} < \frac{\delta_2}{\sqrt{2N \ln 2 - 2N \ln\left(1 - \sqrt[N]{\frac{U_1+1-(U_1+1)\epsilon}{U_1+1-U_1\epsilon}}\right)}} \tag{5.6}
$$

then $P(\text{error}|\omega_1)$ will be greater than $(1-\epsilon)$. Similarly, if:

$$\sigma_{\omega_1} < \frac{\delta_1}{\sqrt{2N\ln 2 - 2N\ln\left(1 - \sqrt[N]{\frac{U_2+1-(U_2+1)\epsilon}{U_2+1-U_2\epsilon}}\right)}} \tag{5.7}$$

then $P(\text{error}|\omega_2)$ will be greater than $(1-\epsilon)$. Thus, we have proved property (2).

Step 2 Here, we construct probability density functions $p_3, p_4, \ldots, p_K$, then we will prove that these distributions also satisfy properties (1) and (2). We only give details for constructing p_3; the other probability density functions are constructed by the same method.

$$\begin{aligned} p_3(x_1, x_2, \ldots, x_N) &= \left(\frac{U_3}{U_3+1}\epsilon\right) t_{31}(x_1, x_2, \ldots, x_N) + \\ &\quad \left(1 - \frac{U_3}{U_3+1}\epsilon\right) t_{32}(x_1, x_2, \ldots, x_N) \\ \text{with} \quad t_{31}(x_1, x_2, \ldots, x_M) &= \frac{1}{U_3}\frac{1}{\left(\sqrt{2\pi}\sigma_3\right)^N}\sum_{j=1}^{U_3} T_{\omega_3}(X_j)\, e^{-\frac{\sum_{i=1}^{N}(x_i - x_{ij})^2}{2\sigma_3^2}} \\ \text{and} \quad t_{32}(x_1, x_2, \ldots, x_M) &= \frac{1}{\left(\sqrt{2\pi}\sigma_{\omega_1}\right)^N} e^{-\frac{\sum_{i=1}^{N}(x_i - x_{i\omega_1})^2}{2\sigma_{\omega_1}^2}} \end{aligned}$$

By the same deduction as in Step 1, we can obtain that, if

$$\begin{aligned} \sigma_3 &< \frac{1}{\sqrt{2\pi}}\left(\frac{\epsilon}{(U_3+1)P}\right)^{\frac{1}{N}} \\ \text{and} \quad \sigma_{\omega_1} &< \frac{\delta_1}{\sqrt{2N\ln 2 - 2N\ln\left(1 - \sqrt{N}\frac{U_3+1-(U_3+1)\epsilon}{U_3+1-U_3\epsilon}\right)}} \end{aligned}$$

we will have

1. For $j = 1, 2, \ldots, U$, if $X_j \in \omega_3$, then $p_3(x_{1j}, x_{2j}, \ldots, x_{Nj}) > P$.
2. $P(\text{error} \mid \omega_3) > (1-\epsilon)$.

Because the basic idea of proving this theorem is to construct Gaussian mixture distributions which have high densities in areas in which the combination algorithm gives incorrect classification, t_{32} can also be generated with another form:

$$t_{32}(x_1, x_2, \ldots, x_N) = \frac{1}{\left(\sqrt{2\pi}\sigma_{\omega_2}\right)^N} e^{-\frac{\sum_{i=1}^{N}(x_i - x_{i\omega_2})^2}{2\sigma_{\omega_2}^2}}$$

and property (2) will also hold if

$$\sigma_{\omega_2} < \frac{\delta_2}{\sqrt{2N\ln 2 - 2N\ln\left(1 - \sqrt[N]{\frac{U_3+1-(U_3+1)\epsilon}{U_3+1-U_3\epsilon}}\right)}}$$

Similarly, we can prove that, for $i = 4, \ldots, K$, the same results also hold. Since all the $P(\text{error} \mid \omega_i)(i = 1, 2, \ldots, K)$ are greater than $(1 - \epsilon)$, the total error rate $P(\text{error})$, which is calculated by equation (5.3), is also greater than $1 - \epsilon$. Thus, we have completed the proof of the NPT.

5.3 Illustrative Examples

The following examples are for the NPT in the case of two classifiers and two classes ($M = 2$ and $K = 2$). There are four score functions for each data point. Because it is difficult to visualise four-dimensional data points, we will change some definitions in Section 5.1 to obtain two-dimensional data points.

Suppose there are two classifiers, each assigning an input X to one of two classes, ω_1 or ω_2, as described by two score functions $f_1(X)$ and $f_2(X)$. The decision rule of the kth classifier ($k = 1, 2$) is:

$$\text{Decide} \begin{cases} X \in \omega_1 \text{ if } f_k(X) > 0 \\ X \in \omega_2 \text{ otherwise} \end{cases}$$

By this definition, we only need two score functions to describe the combination algorithm. We use x_1 and x_2 to represent $f_1(X)$ and $f_2(X)$. If the input data point has a subscript, such as X_i, we will use x_{1i} and x_{2i} to represent $f_1(X_i)$ and $f_2(X_i)$. Based on these definitions, every combination algorithm defines a combination function $F(x_1, x_2)$, with the decision rule:

$$\text{Decide} \begin{cases} X \in \omega_1 \text{ if } F(x_1, x_2) > 0 \\ X \in \omega_2 \text{ otherwise} \end{cases}$$

Thus, we only need one combination function to describe the combination algorithm. We keep other definitions in Sections 5.1 and 5.2 unchanged.

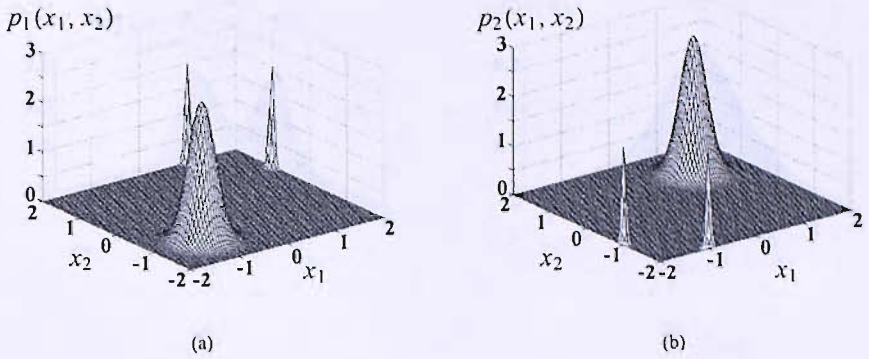


FIGURE 5.1: An example of probability density functions which give bad performance for combination by the sum rule: (a) $p_1(x_1, x_2)$; (b) $p_2(x_1, x_2)$.

Suppose we have four training data points, two of which belong to ω_1 and two of which belong to ω_2 (i.e., $U_1 = 2$ and $U_2 = 2$). The scores of the two data points from ω_1 are given as $\{x_{11}, x_{21}\} = \{1, 2\}$ and $\{x_{12}, x_{22}\} = \{2, 1\}$; the scores of the two data points from ω_2 are given as $\{x_{13}, x_{23}\} = \{-1, -2\}$ and $\{x_{14}, x_{24}\} = \{-2, -1\}$. We assume that the combination function $F(x_1, x_2)$ follows the simple sum rule:

$$\text{Decide} \begin{cases} X \in \omega_1 \text{ if } x_1 + x_2 > 0 \\ X \in \omega_2 \text{ otherwise} \end{cases}$$

It is obvious that this rule is continuous and diverse. We can choose the corresponding $\{x_{1\omega_1}, x_{2\omega_1}\} = \{1, 1\}$ and $\{x_{1\omega_2}, x_{2\omega_2}\} = \{-1, -1\}$. For the sum rule, there is a $\delta_1 = 1$ that makes $B(\{1, 1\}, \delta_1) \in D_1$ and, similarly, a $\delta_2 = 1$ that makes $B(\{-1, -1\}, \delta_2) \in D_2$. Finally, we choose $\epsilon = 0.1$ and $P = 2$. Equations (5.4), (5.5), (5.6) and (5.7) yield $\sigma_1 = \sigma_2 = 0.0515$ and $\sigma_{\omega_1} = \sigma_{\omega_2} = 0.2304$.

Figure 5.1 shows $p_1(x_1, x_2)$ and $p_2(x_1, x_2)$ obtained by setting σ_1, σ_2, σ_{ω_1} and σ_{ω_2} as above. Figure 5.1(a) shows that more than 90% of the probability that the input data belong to ω_1 (i.e., $(1 - \epsilon) = 0.9$) is accumulated near the point $\{-1, -1\}$. At this point, the sum rule gives incorrect classifications. It can also be seen that high probability is also accumulated near the training data points $\{1, 2\}$ and $\{2, 1\}$, which indicates that in such a distribution it is very possible to have these training data but impossible to obtain correct classification by the sum rule.

It may be argued that such 'strange' probability density functions, which are so biased in the 'wrong' areas and near the training data, are not at all likely to be encountered in real-world applications. However, in situations that are not so extreme, we can show that a given combination rule also can not guarantee good performance. In the previous example, if we set $\epsilon = 0.5$ and $P = 0.2$, keeping the other settings the same, we can obtain smoother pdf's (Figure 5.2). However, the error rate is still more than 50%, which is greater than the guess rate.

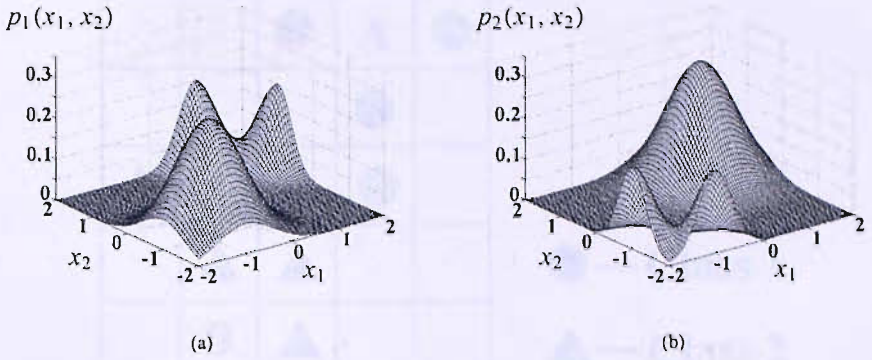


FIGURE 5.2: Further example of pdf's exhibiting poor classification performance obtained by setting $\epsilon = 0.5$ and $P = 0.2$: (a) $p_1(x_1, x_2)$; (b) $p_2(x_1, x_2)$.

5.4 Relation to NFL Theorems

Although Wolpert has proved several NFL theorems for supervised learning (Wolpert, 2001), the basic idea of these proofs is similar. The NFL theorem assumes that the state space which a computer can represent, though perhaps quite large, is finite. For simplicity, we can assume the state space of a computer is a 5×5 grid, as shown in Figure 5.4.

Suppose each square of Figure 5.4 is associated with one of two classes, Class 1 and Class 2, and we know some of these squares belong to Class 1, which are labelled by the circles on the graph; and some belong to Class 2, which are labelled by the triangles. From these, we need to deduce the class labels of the unknown squares. For example, what are the class labels of square A and B respectively? By intuition, A is more likely to be in Class 1 and B in Class 2 because they are surrounded by squares which belong to these classes. But the NFL theorem states that square A and B can belong to either Class 1 or 2 equiprobably if no further assumptions are implemented. For example, if we assume that a square which is in the vicinity of squares belonging to one class is more likely to be in that class, then we can classify A to Class 1 and B to Class 2. However, this assumption can not be mathematically proved. The NFL theorem states that we can not deduce the class labels of the unknown squares based on those we know.

In this chapter, we have further extended the NFL theorem by constructing probability distributions. In this example, if there is a classification algorithm which classifies A to Class 1 and B to Class 2, then we have proved that there exists a set of probability density functions which make A more probable to be in Class 2 and B in Class 1. Thus, the classification algorithm will perform badly under these pdf's. Furthermore, if these squares become infinitely small, the constructed probability density functions will converge to Gaussian mixture distributions, which are good models for many real-world problems based on the central limit theorem.

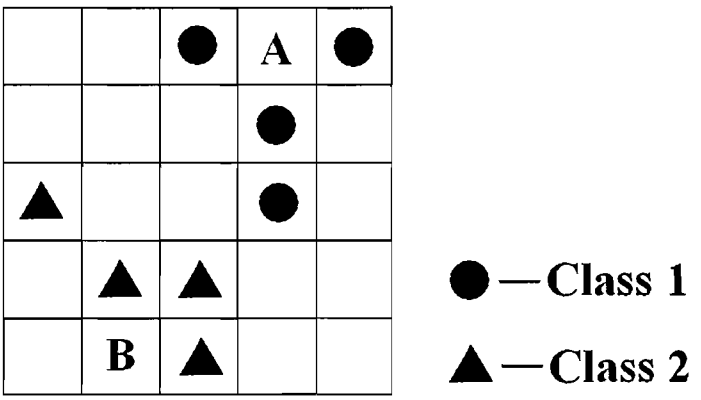


FIGURE 5.3: A figure to explain the idea of the NFL theorem. We can not deduce the class labels in square A and B if no further assumptions are implemented.

5.5 Summary

We have proved the No Panacea Theorem for classifier combination, which states that if the combination function is continuous and diverse, there exist probability density functions in which the combination algorithm will yield very bad performance. Thus, there is no optimal combination algorithm that is suitable for application in all situations.

The No Panacea Theorem is an extension of the No Free Lunch Theorem by explicitly constructing probability density functions to indicate that there exist situations which will make classification algorithm perform badly, no matter which classification algorithm is provided.

Our aim in presenting this theorem is not to criticise any particular algorithm(s) for combining classifiers, but rather to point out the difficulties we might encounter in this area. The pdf's we constructed to prove the theorem are Gaussian mixtures, which are good models for many applications. We also show (by example) that the probability density functions could be very smooth. These observations indicate that there is no universally optimal combination algorithm in real-world applications. From this theorem, we see that a good combination algorithm is not only dependent on combination functions, but also on the probability density functions. So studying the pdf's becomes the first step in finding a good combination algorithm. For example, which kind of probability density functions are more frequently occurring than others? How to incorporate the probability density functions into the design of combination algorithms? These problems remain largely unsolved.

Another fact to be noted is that, although the theorem is built on the background of multiple classifier combination, it can be easily generalised to all supervised learning problems. If we regard data points $\{x_1, x_2, \ldots, x_N\}$ as features in a supervised learning problem, the whole deduction of the algorithm is still correct. Thus, the NPT is valid for all supervised learning problems.

Chapter 6

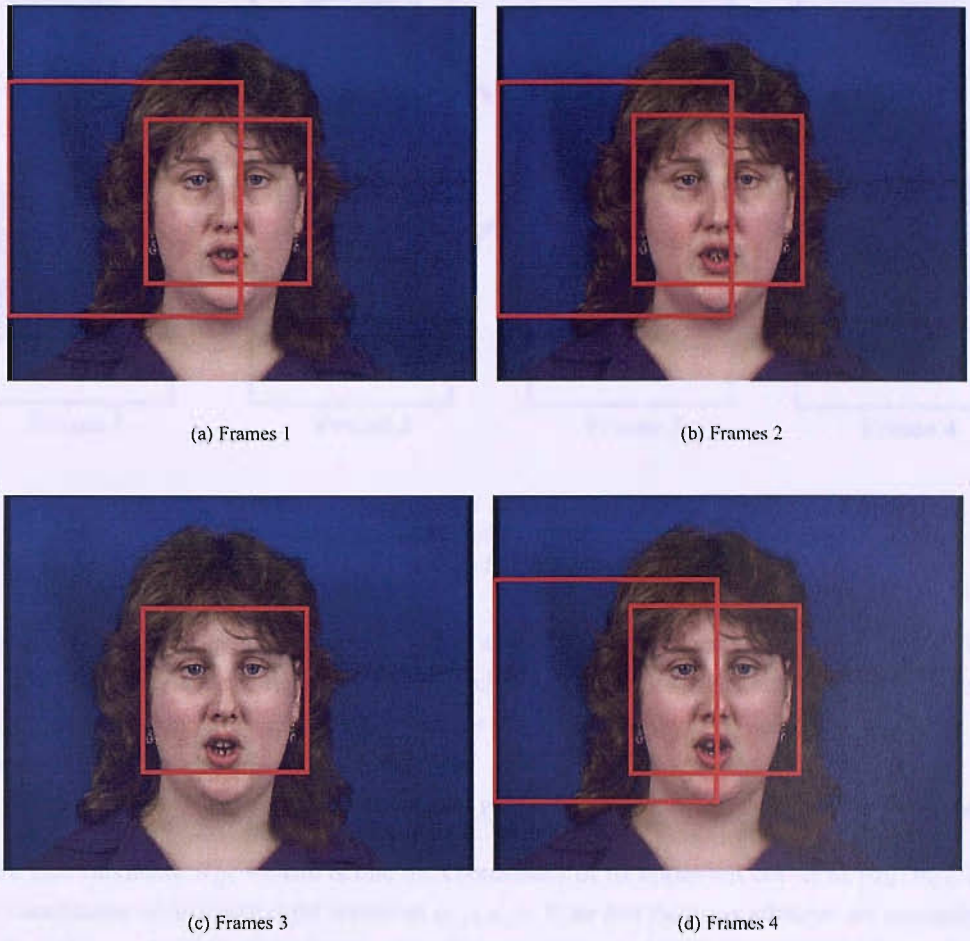
Combining Simultaneous Audio-Visual Data

In Chapter 4, we have discussed how to combine individual audio and visual classifiers. The audio classifier is trained and tested by speech files which are spanned in the time domain, but the visual classifier is trained and tested by static images. In other words, the time-related information of the visual classifier is not used in the combination scheme. In this chapter, we will discuss approaches to use the time-related information of video files. We firstly develop an algorithm for tracking face images in video files, then use these face images to build a visual classifier. Then the combination algorithm in Chapter 4 is used to combine the audio and visual classifiers. We implement this algorithm to the whole XM2VTS database, which consists of 295 subjects. The experiments indicate that the proposed audio-visual combination scheme can achieve good identification rate on this database, which is around 92% when two sessions are used for training, and 97% if three sessions are used for training.

6.1 Face Tracking in Video Files

In this session, we will develop a face tracking algorithm based on dynamic programming (DP). We have previously built a good face detection system as shown in Section 3.2. However, because video files consist of many frames, it can not be guaranteed that the detection system will succeed for every frame. For example, Figure 6.1 shows the detection results for the first four frames of a video file. It is observed that for frame 1, 2 and 4, there are two face candidates, as indicated by the red squares. One of them is the correct face image, and the other is a false alarm. For frame 3, there is only one face candidate, and it is the correct face image.

If we only look at an individual frame, we can not discriminate which face candidate is the false alarm, and which is the correct face image. But when combining the information provided by all these frames, it is possible to use an algorithm to track the true face for each frame. The problem



(a) Frames 1 (b) Frames 2

(c) Frames 3 (d) Frames 4

FIGURE 6.1: The face detection results for the first four frames of a video file. The detected faces are illustrated by red squares.

can be explained by Figure 6.2. We can define a 'path' if we choose exactly one face candidate from each frame. For all these pathes, there exists an optimal one which passes through all true faces in these frames (as shown by the red lines in the figure). Intuitively, the optimal path needs to satisfy two properties. First, each face candidate on the optimal path should really 'be a face'. For example, face candidate X_{12} (the second candidate of the first frame) does not have this property, because there is a big blue region at the left side, which is not typical for a face image. Second, there should not be a huge displacement between adjacent face candidates on the optimal path. In other words, the velocity of face movement should be smooth. Based on these two assumptions, we propose a method for tracking the face images in video files.

Our method is very similar to Lappas *et al.* (2002). In their original work, they have carried out a tracking framework by using dynamic programming (DP). In this paper, we apply their framework in the case of video face detection. First, we define an energy function for each path, which consists of an internal energy and a transit energy. Then we use dynamic programming to obtain the optimal path which minimises the energy function. The internal energy, which is a

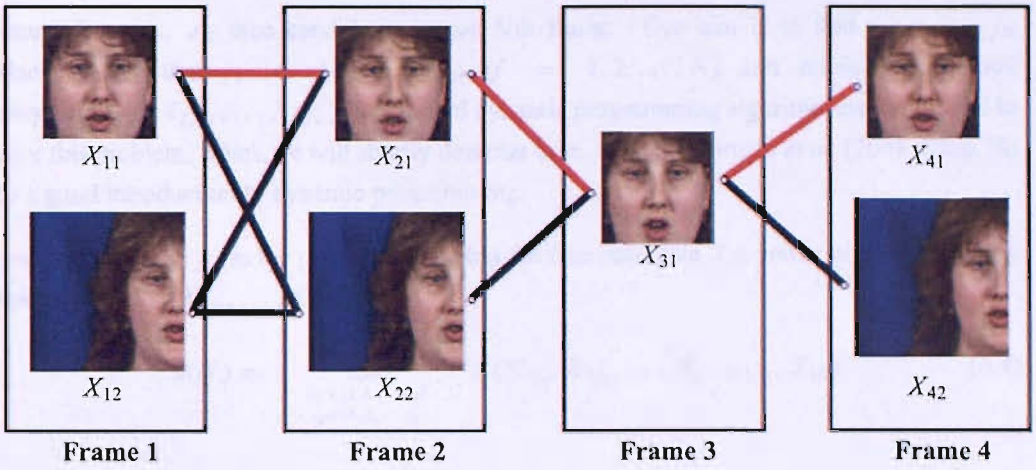


FIGURE 6.2: Face candidates and paths of the four frames shown in Figure 6.1. Here X_{ij} represents the jth face candidate of the ith frame. The optimal path is outlined with red colour.

measure of how much a face candidate is 'like a face', can be defined as follows. Suppose N is the number of pixels in the region of a face candidate X_{ij}, and M is the number of pixels which are regarded as skin colours by the skin colour model as discussed in Section 3.2.1. Then the internal energy is defined as

$$E_{\text{internal}}(X_{ij}) = \frac{M}{N} \tag{6.1}$$

For a face candidate X_{ij}, we can define the coordinates of its upper-left corner as (a_{ij}, b_{ij}), and the coordinates of its lower-right corner as (c_{ij}, d_{ij}). Note that these coordinates are normalised by the height and width of the frame. For example, suppose that the width of the frame is W, and the height is H. Further suppose that the pixel value of the upper-left corner is A_{ij} and B_{ij}. Then

$$a_{ij} = \frac{A_{ij}}{W} \qquad b_{ij} = \frac{B_{ij}}{H}$$

The transit energy from X_{ij} to $X_{(i+1)k}$ is defined as follows.

$$E_{\text{transit}}(X_{ij}, X_{(i+1)k}) = |a_{ij} - a_{(i+1)k}| + |b_{ij} - b_{(i+1)k}| + |c_{ij} - c_{(i+1)k}| + |d_{ij} - d_{(i+1)k}| \tag{6.2}$$

Finally we can define the total energy of a path $(X_{1j_1}, X_{2j_2}, \ldots, X_{Nj_N})$ as

$$E(X_{1j_1}, X_{2j_2}, \ldots, X_{Nj_N}) = w_1 \sum_{i=1}^{N-1} E_{\text{transit}}(X_{ij_i}, X_{(i+1)j_{(i+1)}}) - w_2 \sum_{i=1}^{N} E_{\text{internal}}(X_{ij_i}) \tag{6.3}$$

Here w_1 and w_2 are weights to balance the contributions of the internal and transit energies.

We further suppose there are J_1 face candidates in the first frame, J_2 face candidates in the

second frame,..., J_N face candidates in the Nth frame. Our aim is to find $j_1, j_2, \ldots, j_N$ which satisfy that $j_i \in \{1, 2, \ldots, J_i\}$ $(i = 1, 2, \ldots, N)$ and minimise the total energy $E(X_{1j_1}, X_{2j_2}, \ldots, X_{Nj_N})$. A standard dynamic programming algorithm can be applied to solve this problem, which we will shortly describe here. Refer to Cormen *et al.* (2001, Chap.16) for a good introduction to dynamic programming.

First, we define $S(ij_i)$ as the path which reaches the face candidate X_{ij_i} and minimise the energy function $E(X_{1j_1}, X_{2j_2}, \ldots, X_{ij_i})$.

$$S(ij_i) = \min_{\substack{j_1 \in \{1,2,\ldots,J_1\} \\ j_2 \in \{1,2,\ldots,J_2\} \\ \vdots \\ j_{(i-1)} \in \{1,2,\ldots,J_{(i-1)}\}}} E(X_{1j_1}, X_{2j_2}, \ldots, X_{(i-1)j_{i-1}}, X_{ij_i}) \tag{6.4}$$

The process of finding minimal energy could be written as follows.

$$\min_{\substack{j_1 \in \{1,2,\ldots,J_1\} \\ j_2 \in \{1,2,\ldots,J_2\} \\ \vdots \\ j_{(N-1)} \in \{1,2,\ldots,J_{(N-1)}\} \\ j_N \in \{1,2,\ldots,J_N\}}} E(X_{1j_1}, X_{2j_2}, \ldots, X_{(N-1)j_{N-1}}, X_{Nj_N}) = \min_{j_N \in \{1,2,\ldots,J_N\}} S(Nj_N) \tag{6.5}$$

Further note that the total energy function can be rewritten in a recursive form as in equation (6.6) and (6.7).

$$\begin{aligned} & E\left(X_{1j_1}, X_{2j_2}, \ldots, X_{(i+1)j_{(i+1)}}\right) \\ = \; & E\left(X_{1j_1}, X_{2j_2}, \ldots, X_{ij_i}\right) + w_1 E_{\text{transit}}\left(X_{ij_i}, X_{(i+1)j_{(i+1)}}\right) \\ & -w_2 E_{\text{internal}}\left(X_{(i+1)j_{(i+1)}}\right) \end{aligned} \tag{6.6}$$

with the beginning term

$$E\left(X_{1j_1}\right) = -w_2 E_{\text{internal}}\left(X_{1j_1}\right) \tag{6.7}$$

Then S_{ij_i} can also be written in the following recursive form, as in equation (6.8) and 6.9.

$$\begin{aligned} S(ij_i) = & \min_{j_{(i-1)} \in \{1,2,\ldots,J_{(i-1)}\}} \left[S_{(i-1)j_{(i-1)}} + \omega_1 E_{\text{transit}}\left(X_{(i-1)j_{(i-1)}}, X_{ij_i}\right)\right] \\ & -\omega_2 E_{\text{internal}}\left(X_{ij_i}\right) \end{aligned} \tag{6.8}$$

with the beginning term

$$S(1j_1) = -\omega_2 E_{\text{internal}}\left(X_{1j_1}\right) \tag{6.9}$$

Using equation (6.5), (6.8) and (6.9), the minimal energy path can be recursively decided from frame 1 to N. Thus, we have solved the problem of finding minimal energy path by using dynamic programming.

Disk Number	Session Number	Number of Video files
6	1	295
7	2	294
8	3	291
9	4	295

TABLE 6.1: Number of video files in each session.

This face tracking algorithm is applied to the video files in the XM2VTS database (Disk 6, 7, 8, 9). For each person, there are four video files, which are recorded in four different sessions. In each video file, the person speaks the same sentence, 'John took father's green shoe bench out'. Because there are 295 persons in the database, we should have 1180 video files. However, 5 video files are missing due to reasons that some person didn't present in one session, or some technical problems in recording the video files. Thus, the database consists 1175 video files. Table 6.1 gives the number of video files contained in each session.

We applied this algorithm to the video files, and we obtain 100% successful tracking rate. The criteria of successful tracking has been defined in Section 3.2.2, which is (1) the detected face contains the eyebrows, eyes and mouth; and (2) no false detection occurs.

6.2 Experiments for Combining Audio and Visual Classifiers

After extracting the face images in each video file, the next step is to use these images for face identification. First, we need to divide the video files into training files and testing files. For example, we can use the video files in Session 1 and 2 for training, and the video files in Session 3 and 4 for training. Because each training video file contains around 100 frames, it would be time-consuming if we use face images in all these frames. For simplicity, we only use one face image of every five frames to build the classifier (e.g., for each training video file, we use face images in the first frame, sixth frame, eleventh frame, etc to train the classifier).

The classifier is built by using the PCA method, which was discussed in Section 3.3.1. The reason why we do not use the better DLA method (as discussed in Section 3.4.1) is that the DLA method is especially time-consuming when dealing with such a big amount of frames in these video files. One of our futures work would be devising algorithms to reduce the time complexity of the DLA method, but in this stage, we just use the relatively faster PCA method.

The training process for the PCA method is the same as described in Section 3.3.1, but the testing process is slightly different. Each testing file contains many face images, and the classifier will generate an identification result for each face image. Some method needs to be implemented to combine these results into a single one. Here we use the maximum rule, which is described as follows.

Suppose $P_1, P_2, \ldots, P_M$ are weight vectors which are generated by the M training face images $\Gamma_1, \Gamma_2, \ldots, \Gamma_M$. For a testing video file which contains N faces, we assume these face

Training	Testing Session	Video Identification Rate
1,2	3	62.54
	4	62.71
1,3	2	67.69
	4	62.03
1,4	2	65.31
	3	62.59
2,3	1	67.80
	4	66.10
2,4	1	67.80
	3	68.14
3,4	1	63.73
	2	69.49

TABLE 6.2: Experimental results for face identification in video files. A cross validation test is applied to these files. The classifier is trained by video files in two sessions, and then tested by the remaining sessions. Such a process iterates for each pair of sessions, which generates 12 identification results.

images are $\Lambda_1, \Lambda_2, \ldots, \Lambda_N$, and their corresponding weight vectors are $Q_1, Q_2, \ldots, Q_N$. Then we will find one face image in the training set, and another face image in the testing set which are most similar with each other.

$$\text{Find } u, v = \arg \max_{\substack{i=1\sim M \\ j=1\sim N}} \frac{|P_i \cdot Q_j|}{\|P_i\| \|Q_j\|} \tag{6.10}$$

Then we will classify the testing video file into the class which the training image Γ_u belongs to:

$$\text{Classify the testing video file into } \omega_s, \text{ if } \Gamma_u \in \omega_s \tag{6.11}$$

A cross validation test is applied to the video files. The classifier is trained by video files in two sessions, and then tested by the remaining sessions. Such a process iterates for each pair of sessions. The identification results are summarised in Table 6.2. We can see that the classifier achieves around 65% identification rate over these 295 subjects.

We still use the audio classifier which was discussed in Section 2.4. We also use two sessions for training, and the remaining two sessions for testing. The results are shown in Table 6.3. Generally, the audio classifier has achieved more than 80% identification rate over these 295 subjects.

We use the fusion method which was proposed in Section 4.7 to combine the scores of audio and visual classifiers. To obtain the optimal weighting parameter α, we use two sessions for training the audio and visual classifiers, then use one session for training the optimal weighting parameter α, finally use the remaining session to test the effect of the obtained α. For example, if we use Session 1 and 2 to train the audio and visual classifiers, then we can use Session 4 to train the optimal weighting parameter α, and then test the performance of the chosen α on Session 3.

Training	Testing Session	Audio Identification Rate
1,2	3	85.57
	4	83.05
1,3	2	80.95
	4	84.41
1,4	2	84.69
	3	85.37
2,3	1	78.98
	4	87.46
2,4	1	74.24
	3	86.78
3,4	1	73.22
	2	84.41

TABLE 6.3: Experimental results for speaker identification in video files. A cross validation test is applied to these files. The classifier is trained by video files in two sessions, and then tested by the remaining sessions. Such a process iterates for each pair of sessions, which generates 12 identification results.

Sessions for Training Audio and Visual Classifiers	Session for Training Optimal α	Session for Testing the Performance	Estimated α Using the Empirical Method	Estimated α Using the Proposed Method	True Optimal α on the Testing Session
1,2	3	4	0.77	0.77	0.76
	4	3	0.76	0.76	0.77
1,3	2	4	0.72	0.72	0.71
	4	2	0.71	0.72	0.72
1,4	2	3	0.76	0.78	0.77
	3	2	0.77	0.77	0.76
2,3	1	4	0.76	0.76	0.77
	4	1	0.77	0.79	0.76
2,4	1	3	0.75	0.77	0.81
	3	1	0.81	0.83	0.75
3,4	1	2	0.63	0.70	0.77
	2	1	0.77	0.75	0.63

TABLE 6.4: Experimental results of the empirical and proposed methods on the 12 settings (Two sessions for training the audio and visual classifiers, one for training the weighting parameter α, and using the remaining session to test the performance of the combined classifier). We have listed the estimated α using the empirical and proposed methods. For the proposed method, we set the parameter $A = 0.01$. For comparison, we also list the true optimal α on the test session in each setting.

Alternatively, we can also use Session 3 to train the weighting parameter α, then use Session 4 to test the performance. Thus, we can obtain 12 different experimental settings by iteratively choosing 2 sessions for training the audio and visual classifiers, then choosing one session for training the optimal α, finally testing the obtained optimal α on the remaining session. Table 6.4 shows the experimental results of these 12 experimental settings (Here we set $A = 0.01$ for the proposed method).

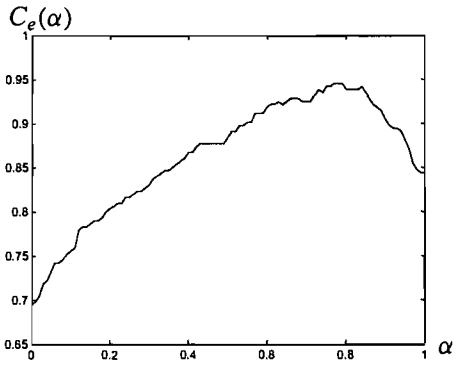
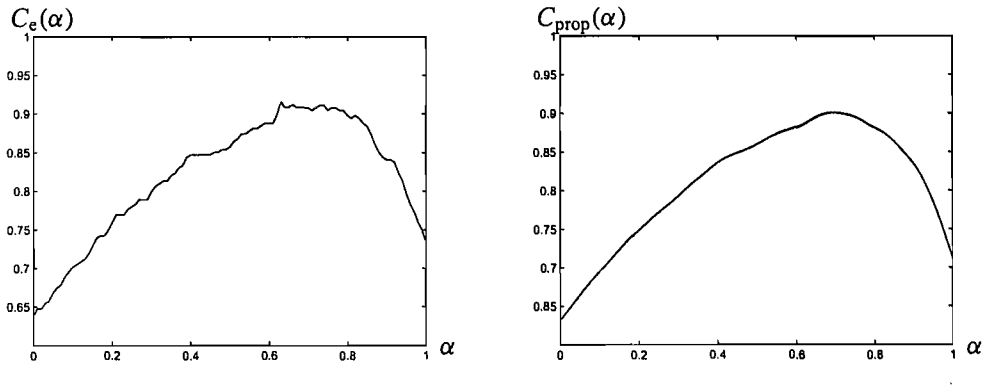


FIGURE 6.3: The empirical identification curve when using Sessions 3 and 4 to train the audio and visual classifiers, and testing the combined classifier on Session 2 with α varying from 0 to 1. The identification rate reaches its maximum when $\alpha = 0.77$.

Table 6.4 further indicates the better performance of our proposed method than the frequently-used empirical method. The fourth column shows that the optimal α's which were found by the empirical method (described in Section 4.8); and the fifth column shows the optimal α's which were found by our proposed method (described in Section 4.7). We also list the true optimal α's which maximise the identification rates of the test sessions in the sixth column. In some cases, such as row 1, 2, 3, 5, 6 and 7, the proposed and the empirical methods obtain similar estimation results. However, for row 4, 9, 11 and 12, the proposed method outperforms the empirical method. It should be noted that especially in Row 11, the results of the proposed method are much better. In Row 11, we use Sessions 3 and 4 to train the audio and visual classifiers, and use Session 1 to train the optimal α, finally use Session 2 to test the combined classifier. The estimated α of the empirical method is 0.63, and the estimated α of the proposed method is 0.70. The true optimal α in this situation is 0.77, which indicates that our proposed method is better than the empirical one in this case. Figure 6.3 and 6.4 visualise this example. In Figure 6.4, both the empirical and proposed methods generate identification curves similar to the identification curve on the testing session (Figure 6.3). The empirical estimation curve in Figure 6.4(a) has a non-smooth nature, which generates an unstable estimation of the optimal α at 0.63. However, when this curve is smoothed by the proposed method, as in Figure 6.4(b), a better estimation is obtained at $\alpha = 0.70$.

However, the proposed method does not always generate better estimation than the empirical one. As shown in Table 6.4, the empirical method is slightly better than the proposed one for Row 8 and 10, which emphasises the situation that the proposed method can not guarantee better performance in every case. It can only achieve slightly better estimation results in a statistical sense.

Table 6.5 shows the identification rates of these 12 experimental settings by using the α's generated by the proposed method. For comparison, the identification rates of the true optimal α's which maximise the identification rates on the testing sessions are also listed here. We can see that the identification rates are more than 90% on average.



(a) (b)

FIGURE 6.4: The estimated identification curve by using Sessions 3 and 4 to train the audio and visual classifiers, and using Session 1 to train the optimal α. (a) The estimated identification curve by using the empirical method, which reaches its maximum when $\alpha = 0.63$. (b) The estimated identification curve by using the proposed method, which reaches its maximum when $\alpha = 0.70$. From Figure 6.3, we can see that the optimal α equals to 0.77 when using Sessions 3 and 4 for training the audio and visual classifiers, and using Session 2 to train the weighting parameter α. The proposed method is better for estimating the optimal α than the empirical one in this case.

Sessions for Training Audio and Visual Classifiers	Session for Training Optimal α	Session for Testing the Performance	Identification Rate with the α Generated by the Proposed Method	Identification Rate with the true Optimal α
1,2	3	4	0.9051	0.9085
	4	3	0.9278	0.9313
1,3	2	4	0.9186	0.9186
	4	2	0.9116	0.9116
1,4	2	3	0.9456	0.9456
	3	2	0.9456	0.9456
2,3	1	4	0.9356	0.9390
	4	1	0.9254	0.9254
2,4	1	3	0.9492	0.9627
	3	1	0.8881	0.9017
3,4	1	2	0.9254	0.9458
	2	1	0.9051	0.9153

TABLE 6.5: Experiments results of the empirical and proposed methods on the 12 settings. We have listed the estimated α using the empirical and proposed methods, respectively. For comparison, we also list the true optimal α which maximises the identification rate on the testing session in each setting. The identification rates on the test sessions are more than 90% on average.

	Audio Identification Rate (%)	Visual Identification Rate (%)	Combined Identification Rate (%)
Fox *et al.* (2003)	98.01	93.23	100
Our Results	94.92	62.71	96.95

TABLE 6.6: Comparison of our identification results with Fox *et al.* (2003). The audio classifier by Fox *et al.* (2003) achieves 98.01% identification rate, which is 3 percent higher than our 94.92% identification rate but higher rates are only to be expected for the easier text-dependent task. Because they used a commercial face recognition software contrasting to our preliminary algorithms, their face identification rate is 93.23%, which is much higher than our 62.71% result. By combination of these two classifiers, they achieved 100% identification rate, contrasting to our 96.95% combined identification rate.

6.3 Comparison With Related Publications

Another paper Fox *et al.* (2003), which was published in 2003, also discussed the audio-visual person identification problem by using the XM2VTS database. They built a text-dependent speaker identification classifier based on hidden Markov models (HMM). For face identification, they used a commercial software FaceIt to generate the identification results. Both classifiers are tested on 291 subjects of the XM2VTS database, contrasting to 295 subjects in our experiment settings. They use session 1, 2 and 3 for training, and session 4 for testing.

In order to compare our results with theirs, we use the same experimental condition as them. In this experiment, Session 1, 2 and 3 are used for training the audio and visual classifiers, and session 4 is used for testing the performance of the system. Because there is no data available to train the optimal α, we will obtain α's value from Table 6.4. We take the estimated α value using the proposed method in this table (Column 4, Row 1), which is 0.77. This value was obtained by using Session 1 and 2 to train the audio and visual classifiers, and using Session 3 to train the optimal α, then using Session 4 to test the performance of the combined classifier.. This experimental settings are valid because session 4 is not used in the training process. Our experimental results are listed in Table 6.6. For comparison, we also list the results which was published in Fox *et al.* (2003).

We can see that the audio classifier by Fox *et al.* (2003) achieves 98.01% identification rate, which is 3 percent higher than our 94.92% identification rate but higher rates are only to be expected for the easier text-dependent task. Because they used a commercial face recognition software FACEIT, contrasting to our preliminary algorithms, their face identification rate is 93.23%, which is much higher than our 62.71% result. By combination of these two classifiers, they achieved 100% identification rate, contrasting to our 96.95% combined identification rate.

In their paper, they also built another classifier for lip motion, and combined it with the audio and visual classifiers. They also devised a fusion method to combine scores of different classifiers when the audio signal was contaminated with noise. They tested the three-classifier system in different signal-to-noise ratio (SNR), and obtained good results even when immense noise is

added to the audio signal. For example, by using the three classifiers, they can achieve 96.81% identification rate when SNR equals to 0.

6.4 Summary

In this chapter, we have discussed the possibility of extending the face identification method to video files. A face tracking method based on dynamic programming is proposed to track the face images. After that, a PCA classifier is implemented for identification. Experiments indicate that this method can achieve around 65% identification rate by using two sessions of the database for training, and the other two sessions for testing (Table 6.2). We still use the audio classifier which was discussed in Chapter 2.4 and obtain around 80% identification rate (Table 6.3). , The combination methods discussed in Chapter 4 are implemented to combine the audio and visual classifiers, and achieves slightly better identification results than the empirical one. We obtain more than 90% identification rate, as shown in Table 6.5.

We also compared our results with another publication (Fox *et al.*, 2003), their results are better than ours. This is because they use a text-dependent speaker classifier, which is an easier task than the GMM-based text-independent method in this thesis. In addition, they also use a commercial face recognition software, which generates much better identification results than our preliminary PCA classifier.

The good identification results both in this thesis and in Fox *et al.*'s paper indicate that recent computer algorithms can achieve nearly perfect identification rate for the audio-visual person identification task, provided that (1) the audio and visual signals are recorded in controlled conditions, for example, the audio signals should not contain too much noise, and the video signals should be similar in pose and illumination for training and testing; (2) the size of the database should not be too large. Good identification performance can not be guaranteed if the database includes thousands of subjects. Although promising results have been achieved in this paper, it is still a long way for computers to achieve identification results as high as human beings in this audio-visual person identification task.

Chapter 7

Conclusions and Future Work

The main contribution of this thesis lies in three parts. First, we have constructed face identification and speaker identification classifiers by using some benchmark algorithms. We show that combination of the two classifiers could achieve very high identification rate on the XM2VTS database. Thus, we have proved the potential of combining multiple classifiers for biometric recognition. Second, a novel method based on estimation of probability density functions for the scores under a Gaussian assumption is proposed for finding the optimal weighting parameters for combination of the audio and visual classifiers. Experiments indicate that this method has advantages in reducing both the bias and variance of the estimation and is superior to other three frequently-used methods. Another contribution of this thesis is that we have theoretically proved that there is no 'perfect' combination algorithm suitable for all situations (the No Panacea Theorem). As a generalisation of the No Free Lunch Theorem, we have proved that if the combination function is continuous and diverse, there exists a situation in which the combination algorithm will yield very bad performance. This theorem has denied the attempt to find an generally optimal combination algorithm.

7.1 Limitations

Several limitations of our research need to be clarified here. Firstly, as indicated in Chapter 4 and 6, although the proposed combination method is better than the empirical method in the sense of average performance, it can not guarantee good performance in every case. This re-emphasises the idea of the No Panacea Theorem that there is no generally optimal combination method.

Secondly, at the current stage, the audio-visual classifier does not run in real time. For experiments on the 295 subjects, typical running time is about one minute per test file for the PCA face classifier; and ten minutes per test file for the DLA face classifier. The running time of the PCA method can be improved to fit the requirement of a real-time system if we change

the MATLAB code to C++. However, more research work needs to be carried out to reduce the running time of the DLA face classifier.

7.2 Future Work

Although some novel approaches has been discussed in this thesis, there are still many unsolved problems remaining in the area of audio-visual person identification. From my experiences, I think the following problems are valuable directions for our future work.

1. **Developing combination algorithms which are robust to noise and unpredictable situations:** In Chapter 4, we have proposed a method to choose the optimal weighting parameter for the weighted-sum rule. This method is based on the assumption that the training the training and testing data are independently drawn from the same probability distribution. However, this assumption may not hold in practice, especially when noise or unexpected environmental conditions have dramatically changed the probability distribution of the test data. In this situation, it is more preferable to use adaptive combination methods whose parameters could be gradually adjusted according to different noise level or environmental conditions. This problem has been partially discussed in several publications (Wark *et al.*, 1999; Wark, 2000; Sanderson and Paliwal, 2003), which could be our references for further research work.

2. **Combining Visual Features with the Audio-Visual Classifier:** Visual features, especially the mouth movement information, provide information of the person's identity. It is a relatively new approach to combine the visual features with voice for person recognition. Several publications have discussed the possibility of incorporating the mouth movement information for speaker and speech recognition (Dupont and Luettin, 2000; Fu *et al.*, 2003; Lucey *et al.*, 2005), which provides a basis for further research.

3. **Research Work on Face Recognition:** It is noted that our face recognition classifier achieves around 65% identification rate over 295 subjects on the XM2VTS database. However, it is very possible to improve the face identification rate to a higher level. Face recognition is a long-standing area which has attracted the attention of many researchers. Recently, more research work concentrates on building 3-D face models to solve the problem of pose and illumination (Huang *et al.*, 2003; Blanz and Vetter, 2003; Lee *et al.*, 2004), which could be an option for our future work.

4. **Generalising the Method of Finding Optimal Weighting Parameter(s) to Person Verification Cases:** In Chapter 4, we have discussed the possibility of generalising the proposed method to verification cases. For verification, a similar approach can be applied for choosing the optimal weighting parameter based on minimising the equal error rate (EER).

5. **Theoretical Study on Multiple Classifier Combination:** In Chapter 5, we have pointed out that studying the probability density function becomes the first step in finding a good combination algorithm. More research work could be carried out on studying the pdf's. The basic idea is to classify the pdf's according to their properties, then the relationships between each class of pdf's and a specific combination algorithm could be revealed theoretically.

6. **Building a Real-Time Audio-Visual Person Recognition System:** Our long-time goal is to convert our research work to a commercial system which could be used in real-time recognition tasks. More research work and technical deployment needs to be carried out to achieve this goal.

Appendix A

The Images and Video Files in the XM2VTS Database

This appendix will show several images and video files in the XM2VTS database. We are concerned with the inter-class variations of these images and video files, and try to bring the reader some intuition of how easy or difficult the face detection and recognition tasks would be on this database.

For most of these files, the inter-class appearance variations are not very obvious. Figure A.1 is an example of the images which has little variations . Face recognition is a comparatively easy task for these images. Both the PCA and DLA methods can perform well in this situation. However, there are shape variations for some subjects. For example, Figure A.2 shows most of these variations. We can see that the subject changed hair style, earrings and glasses in these sessions, which brought some difficulties to face recognition algorithms. Experiments indicate that our algorithms perform not so well in this situation. In this case, the PCA method performs even worse than DLA because it does not contain much local information as the DLA algorithm. There are also pose variations for some subjects. Figure A.3 indicates one example of the pose variation problem. In this example, the subject turned her head with different angle in each session. This situation brought many difficulties for the face recognition task. Our PCA and DLA algorithm perform badly in this situation.

We also show images contained in a video file as an example. Figure A.4 lists all frames in the video file '000_1_3.avi', which is generated by the subject '000' during the first session, when he speaks the third sentence 'John took father's green show bench out'. The mouth movement information is obvious in this figure.

From the figures shown here, we can see that most of images and video files do not have much variations. Compared with other databases, such as FERET (Rizvi *et al.*, 1998; Phillips *et al.*, 2000), XM2VTS is a relatively easy database for face recognition. However, there are still shape and pose variations in this database, which bring some problems for face recognition algorithms.

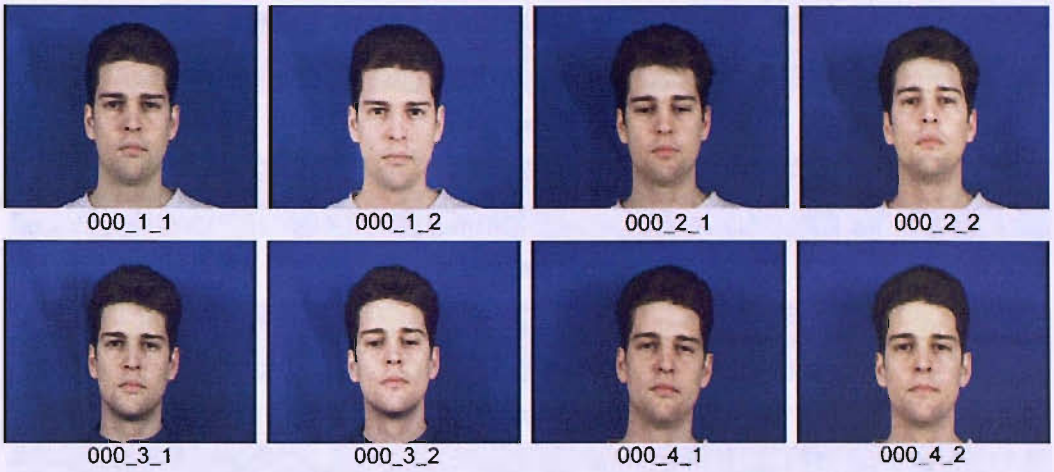


FIGURE A.1: Face images of subject '000', which is taken during four sessions, two for each session.

FIGURE A.2: Face images of subject '005', which is taken during four sessions, two for each session. The subject changed her hair style, earrings and glasses in these sessions.

FIGURE A.3: Face images of subject '050', which is taken during four sessions, two for each session. The subject turned her head with different angle in each session.

(a) Frame 1 – 10

(b) Frame 11 – 20

(c) Frame 21 – 30

(d) Frame 31 – 40

(e) Frame 41 – 50

(f) Frame 51 – 60

(g) Frame 61 – 70

(h) Frame 71 – 80

(i) Frame 81 – 90

(j) Frame 91 – 100

(k) Frame 101 – 103

FIGURE A.4: The frames in video file '000_1_3.avi', which is generated by the subject '000' during the first session, when he speaks the third sentence 'John took father's green show bench out'.

Appendix B

Calculating α_{true} in Section 4.10

This appendix will describe the procedure of calculating α_{true}, which was defined in Section 4.10. We will calculate it under assumptions of Gaussian and uniform distributions, respectively.

We first consider the case of Gaussian distributions. The construction of scores is reiterated as follows. For a 2-classifier and K-class problem. We need to construct $2K$ scores of an input X which belongs to a specific class. For example, we can assume that $X \in \omega_s$. These scores are represented as $f_m^k(X)$, where $m \in \{1, 2\}$ and $k \in \{1, 2, \ldots, K\}$. For constructing $f_m^1(X), f_m^2(X), \ldots, f_m^K(X)$ ($m \in \{1, 2\}$), we first generate K random numbers, each of which is uniformly distributed in the range of $[0, 200]$. We use $n_{m1}, n_{m2}, \ldots, n_{mK}$ to represent these K numbers. We choose n_{ms} as the maximum of these K numbers ($n_{ms} = \max\{n_{m1}, n_{m2}, \ldots, n_{mK}\}$), since it is reasonable to assume that the highest score will be obtained for the correct class. Next we generate another K random numbers, $\sigma_{m1}, \sigma_{m2}, \ldots, \sigma_{mK}$, each of which is uniformly distributed in the range of $[0, \sigma_m^{\max}]$. Here $\sigma_m^{\max}$ is a controlling parameter. The scores $f_m^1(X), f_m^2(X), \ldots, f_m^K(X)$ are generated as follows. For each $m \in \{1, 2\}$ and $k \in \{1, 2, \ldots, K\}$, $f_m^k(X)$ is a random sample drawn from a Gaussian distribution with mean n_{mk} and variance σ_{mk}.

The following procedure describes how to calculate α_{true} based on the above settings. Recall the weighted-sum rule for combining scores of two classifiers:

$$f_{\text{comb}}^k(X, \alpha) = \alpha f_1^k(X) + (1 - \alpha) f_2^k(X) \qquad k = 1, 2, \ldots, K$$

Since $f_1^k(X)$ and $f_2^k(X)$ are Gaussian distributions with mean n_{1k}, n_{2k}, and variance σ_{1k}, σ_{2k}, $f_{\text{comb}}^k(X, \alpha)$ is also a Gaussian distribution distribution with mean n_k and variance σ_k, which are given as follows:

$$n_k = \alpha n_{1k} + (1 - \alpha) n_{2k}, \quad \sigma_k = \sqrt{\alpha^2 \sigma_{1k}^2 + (1 - \alpha)^2 \sigma_{2k}^2} \quad (k = 1, 2, \ldots, K) \qquad \text{(B.1)}$$

For simplicity, we use $P_k(x, \alpha)$ to represent the probability distribution of $f_{\text{comb}}^k(X, \alpha)$,

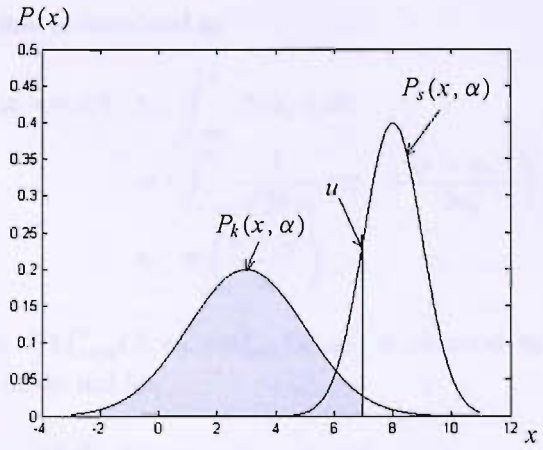


FIGURE B.1: A figure to explain equation (B.3). If $f^s_{\text{comb}}(X, \alpha)$ equals to a specific value u, the probability that $f^k_{\text{comb}}(X, \alpha) < u$ should be the area of the gray region. Finally, the probability $P\left(f^k_{\text{comb}}(X, \alpha) < f^s_{\text{comb}}(X, \alpha)\right)$ is obtained by averaging the above probability over all u's on the real line.

where $k = 1, 2, \ldots, K$. As indicated before, $P_k(x, \alpha)$ is a Gaussian distribution with mean n_k and variance σ_k.

We have assumed that $X \in \omega_s$. According to decision rule 4.20, the probability of correct recognition, which is defined as $C_{\text{true}}(\alpha)$, should be the probability that $f^s_{\text{comb}}(X, \alpha)$ is the largest of all K scores $f^k_{\text{comb}}(X, \alpha)$ $(k = 1, 2, \ldots, K)$. Since these scores are independently drawn from Gaussian distributions, this probability can be further divided as a product of individual probabilities, as shown in equation (B.2).

$$\begin{aligned} C_{\text{true}}(\alpha) &= P\left(\bigcap_{k=1, k\neq s}^{K} f^k_{\text{comb}}(X, \alpha) < f^s_{\text{comb}}(X, \alpha)\right) \\ &= \prod_{k=1, k\neq s}^{K} P\left(f^k_{\text{comb}}(X, \alpha) < f^s_{\text{comb}}(X, \alpha)\right) \end{aligned} \tag{B.2}$$

As mentioned previously, $f^k_{\text{comb}}(X, \alpha)$ is a Gaussian distributions with mean n_k and variance σ_k; and $f^s_{\text{comb}}(X, \alpha)$ is a Gaussian distribution with mean n_s and variance σ_s. Because of this, $P\left(f^k_{\text{comb}}(X, \alpha) < f^s_{\text{comb}}(X, \alpha)\right)$ can be calculated by the following equation.

$$P\left(f^k_{\text{comb}}(X, \alpha) < f^s_{\text{comb}}(X, \alpha)\right) = \frac{1}{\sqrt{2\pi}\sigma_s} \int_{-\infty}^{+\infty} \Phi\left(\frac{u - n_k}{\sigma_k}\right) \exp\left(-\frac{(u - n_s)^2}{2\sigma_s^2}\right) \mathrm{d}u$$

Figure B.1 explains how this equation comes from. If $f^s_{\text{comb}}(X, \alpha)$ equals to a specific value u, as shown in this figure, the probability that $f^s_{\text{comb}}(X, \alpha) < u$ should be the area of the gray region

in Figure B.1, which could be calculated as:

$$\begin{aligned} P_k(x < u, \alpha) &= \int_{-\infty}^{u} P_k(x, \alpha)\mathrm{d}x \\ &= \int_{-\infty}^{u} \frac{1}{\sqrt{2\pi}\sigma_i} \exp\left(-\frac{(x - n_k)^2}{2\sigma_k^2}\right)\mathrm{d}x \\ &= \Phi\left(\frac{u - n_k}{\sigma_k}\right) \end{aligned}$$

Finally, the probability $P\left(f_{\text{comb}}^k(X, \alpha) < f_{\text{comb}}^s(X, \alpha)\right)$ is obtained by averaging the above probability over all u's on the real line.

$$\begin{aligned} & P\left(f_{\text{comb}}^k(X, \alpha) < f_{\text{comb}}^s(X, \alpha)\right) \\ = & \int_{-\infty}^{+\infty} P_k(x < u, \alpha) P_s(u, \alpha)\mathrm{d}u \\ = & \frac{1}{\sqrt{2\pi}\sigma_s} \int_{-\infty}^{+\infty} \Phi\left(\frac{u - n_k}{\sigma_k}\right) \exp\left(-\frac{(u - n_s)^2}{2\sigma_s^2}\right)\mathrm{d}u \end{aligned}$$

The above equation could be further simplified as

$$\begin{aligned} & P\left(f_{\text{comb}}^k(X, \alpha) < f_{\text{comb}}^s(X, \alpha)\right) \\ = & \frac{1}{\sqrt{2\pi}\sigma_s} \int_{-\infty}^{+\infty} \Phi\left(\frac{u - n_k}{\sigma_k}\right) \exp\left(-\frac{(u - n_s)^2}{2\sigma_s^2}\right)\mathrm{d}u \\ = & \Phi\left(\frac{n_s - n_k}{\sqrt{\sigma_k^2 + \sigma_s^2}}\right) \end{aligned} \tag{B.3}$$

From equation (B.2) and (B.3), the probability of correct recognition C_{true} can be calculated as

$$\begin{aligned} C_{\text{true}}(\alpha) &= P\left(\bigcap_{k=1, k\neq s}^{K} f_{\text{comb}}^k(X, \alpha) < f_{\text{comb}}^s(X, \alpha)\right) \\ &= \prod_{k=1, k\neq s}^{K} P\left(f_{\text{comb}}^k(X, \alpha) < f_{\text{comb}}^s(X, \alpha)\right) \\ &= \prod_{k=1, k\neq s}^{K} \Phi\left(\frac{n_s - n_k}{\sqrt{\sigma_k^2 + \sigma_s^2}}\right) \end{aligned}$$

The integral of the above equation can be estimated by using numerical method. The true optimal weighting parameter, α_{true}, is selected as the value which maximises $C_{\text{true}}(\alpha)$.

$$\alpha_{\text{true}} = \arg_{\alpha \in [0,1]} \max C_{\text{true}}(\alpha) \tag{B.4}$$

In Section 4.10, α_{true} is obtained by varying α from 0 to 1, with 0.001 increment.

The procedure of obtaining α_{true} under uniform distributions is similar to the case of Gaussian distributions. In order to clarify our description, we reiterate the score-generating process. For constructing $f_m^1(X), f_m^2(X), \ldots, f_m^K(X)$ ($m \in \{1, 2\}$), we first generate K random numbers, each of which is uniformly distributed in the range of $[0, 200]$. We use $n_{m1}, n_{m2}, \ldots, n_{mK}$ to represent these K numbers. We also choose n_{ms} as the maximum of these K numbers ($n_{ms} = \max\{n_{m1}, n_{m2}, \ldots, n_{mK}\}$). Next we generate another K random numbers, $\sigma_{m1}, \sigma_{m2}, \ldots, \sigma_{mK}$, each of which is uniformly distributed in the range $[0, \sigma_m^{\max}]$. Here $\sigma_m^{\max}$ is a controlling parameter. The scores $f_m^1(X), f_m^2(X), \ldots, f_m^K(X)$ are generated as follows. For each $m \in \{1, 2\}$ and $k \in \{1, 2, \ldots, K\}$, $f_m^k(X)$ is a random sample drawn from a uniform distribution in the range $(n_{mk} - \sigma_{mk}, n_{mk} + \sigma_{mk})$.

Recall the weighted-sum rule for combining scores of two classifiers:

$$f_{\text{comb}}^k(X, \alpha) = \alpha f_1^k(X) + (1 - \alpha) f_2^k(X) \qquad k = 1, 2, \ldots, K$$

As described above, $f_1^k(X)$ and $f_2^k(X)$ are uniform distributions distributions in the range of $[n_{1k} - \sigma_{1k}, n_{1k} + \sigma_{1k}]$ and $[n_{2k} - \sigma_{2k}, n_{2k} + \sigma_{2k}]$, respectively. Then, $\alpha f_1^k(X)$ and $(1-\alpha) f_2^k(X)$ are also uniform distributions, which span in the range of $[\alpha(n_{1k} - \sigma_{1k}), \alpha(n_{1k} + \sigma_{1k})]$ and $[(1 - \alpha)(n_{2k} - \sigma_{2k}), (1 - \alpha)(n_{2k} + \sigma_{2k})]$, respectively. Because $f_{\text{comb}}^k(X, \alpha)$ is the sum of $\alpha f_1^k(X)$ and $(1 - \alpha) f_2^k(X)$, which are two uniform distributions, it has a distribution with a trapezoid shape. We refer to this distribution as a 'trapezoid distribution'.

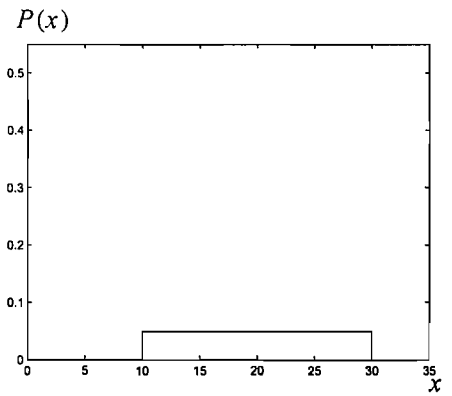
The analytical form of the trapezoid distribution is given as follows. Suppose $f_1^k(X)$ is a uniform distribution which spans in the range of $[n_{1k} - \sigma_{1k}, n_{1k} + \sigma_{1k}]$, and $f_2^k(X)$ is a uniform distribution in the range of $[n_{2k} - \sigma_{2k}, n_{2k} + \sigma_{2k}]$. We firstly calculate four parameters A, B, C and D, as in equation (B.5).

$$\begin{aligned} A &= \alpha(n_{1k} - \sigma_{1k}); & B &= \alpha(n_{1k} + \sigma_{1k}) \\ C &= (1 - \alpha)(n_{2k} - \sigma_{2k}); & D &= (1 - \alpha)(n_{2k} + \sigma_{2k}) \end{aligned} \tag{B.5}$$

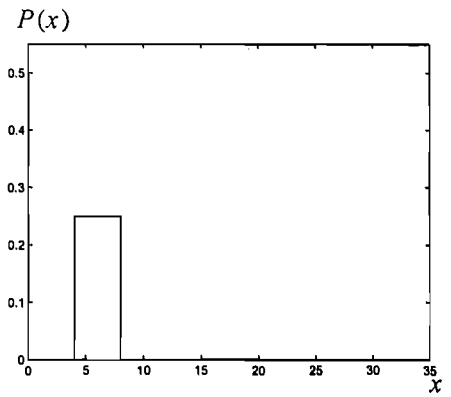
We further suppose that $(D - C) \leq (B - A)$. If $(D - C) > (B - A)$, we will swap the definitions of A, B and C, D so that this assumption still holds. Then the distribution of $f_{\text{comb}}^k(X, \alpha)$, denoted by $P_k(x, \alpha)$ is given as follows:

$$P_k(x, \alpha) = \begin{cases} 0 & x < A + C \\ \frac{1}{(B-A)(D-C)}(x - A - C) & A + C \leq x < A + D \\ \frac{1}{B-A} & A + D \leq x < B + C \\ -\frac{1}{(B-A)(D-C)}(x - B - D) & B + C \leq x < B + D \\ 0 & x > B + D \end{cases} \tag{B.6}$$

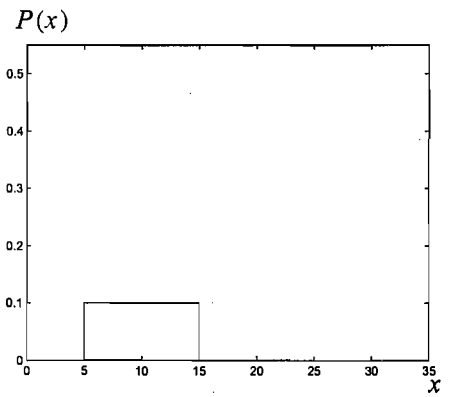
Fig B.2 illustrates the process of generating the trapezoid distribution. The shape of $P_k(x, \alpha)$ is shown in Figure B.2(e). We further introduce the distribution function of $f_{\text{comb}}^k(x, \alpha)$, which is



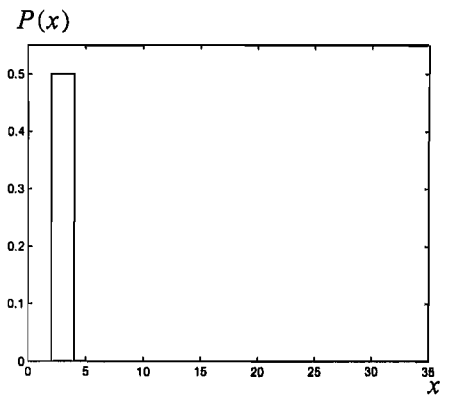
(a) Probability distribution of $f_1^k(X)$



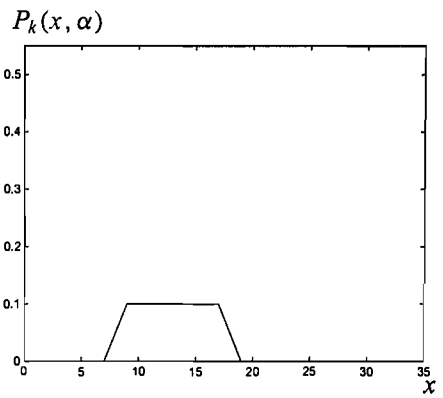
(b) Probability distribution of $f_2^k(X)$



(c) Probability distribution of $\alpha f_1^k(X)$ ($\alpha = 0.5$)



(d) Probability distribution of $(1-\alpha)f_2^k(X)$ ($\alpha = 0.5$)



(e) Probability distribution of $f_{\text{comb}}(X, \alpha) = \alpha f_1^k(X) + (1-\alpha) f_2^k(X)$ ($\alpha = 0.5$)

FIGURE B.2: Probability distributions of $f_{\text{comb}}^k(X, \alpha)$, which is a weighted sum of two uniform distributions $f_1^k(X)$ and $f_2^k(X)$. (a) The uniform distribution of $f_1^k(X)$, which spans in the range of $[n_{1i} - \sigma_{1i}, n_{1i} + \sigma_{1i}]$; (b) The uniform distribution of $f_2^k(X)$, which spans in the range of $[n_{2i} - \sigma_{2i}, n_{2i} + \sigma_{2i}]$; (c) $\alpha f_1^k(X)$ is also a uniform distribution, which spans in the range of $[\alpha(n_{1i} - \sigma_{1i}), \alpha(n_{1i} + \sigma_{1i})]$; (d) $(1-\alpha) f_2^k(X)$ is also a uniform distribution, which spans in the range of $[(1-\alpha)(n_{2i} - \sigma_{2i}), (1-\alpha)(n_{2i} + \sigma_{2i})]$; (e) $f_{\text{comb}}^k(X, \alpha) = \alpha f_1^k(X) + (1-\alpha) f_2^k(X)$ is a trapezoid distribution. The analytical form of this distribution is clarified in equation (B.5) and (B.6)

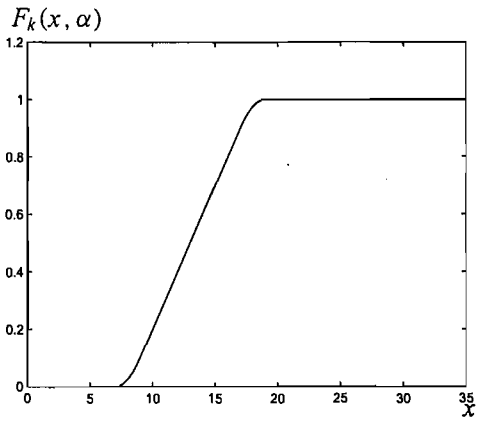


FIGURE B.3: $F_k(x, \alpha)$ is the distribution function of $f^k_{\text{comb}}(x, \alpha)$, which is the integral of $P_k(x, \alpha)$.

the integral of $P_k(x, \alpha)$.

$$F_k(x, \alpha) = \int_{-\infty}^{x} P_k(u, \alpha)\mathrm{d}u = \begin{cases} 0 & x < A + C \\ \frac{(x-A-C)^2}{2(B-A)(D-C)} & A + C \le x < A + D \\ \frac{x-A-D}{B-A} + \frac{D-C}{2(B-A)} & A + D \le x < B + C \\ 1 - \frac{(B+D-x)^2}{2(B-A)(D-C)} & B + C \le x < B + D \\ 1 & x > B + D \end{cases} \tag{B.7}$$

Figure B.3 illustrates the shape of $F_k(x, \alpha)$.

Using similar deduction as we did in the Gaussian distribution case, we could obtain $P\left(f^k_{\text{comb}}(X, \alpha) < f^s_{\text{comb}}(X, \alpha)\right)$ as follows:

$$P\left(f^i_{\text{comb}}(X, \alpha) < f^s_{\text{comb}}(X, \alpha)\right) = \int_{-\infty}^{+\infty} F_k(u) P_s(u)\mathrm{d}u$$

Finally, the correct identification rate, $C_{\text{true}}(\alpha)$, can be written as:

$$\begin{aligned} C_{\text{true}}(\alpha) &= P\left(\bigcap_{k=1, k\neq s}^{K} f^k_{\text{comb}}(X, \alpha) < f^s_{\text{comb}}(X, \alpha)\right) \\ &= \prod_{k=1, k\neq s}^{K} \int_{-\infty}^{+\infty} F_k(u) P_s(u)\mathrm{d}u \end{aligned} \tag{B.8}$$

The true optimal weighting parameter, α_{true}, is selected as the value which maximises $C_{\text{true}}(\alpha)$.

$$\alpha_{\text{true}} = \arg_{\alpha \in [0,1]} \max C_{\text{true}}(\alpha) \tag{B.9}$$

Similar to the Gaussian distribution case, α_{true} is obtained by varying α from 0 to 1, with 0.001 increment.

Appendix C

Publications Related to this Thesis

Hu, R. and R. I. Damper (2007). Combining face, voice and mouth movement information for person recognition. *Journal on Multimodal User Interfaces*, (In preparation).

Hu, R. and R. I. Damper (2007). Audio-visual person identification on the XM2VTS database. In *Proceedings of the International Conference on Auditory-Visual Speech Processing, AVSP'07*, Hivarenbeek, The Netherlands, pp. 152–157.

Hu, R. and R. I. Damper (2007). A 'No Panacea Theorem' for Classifier Combination. Submitted to *Pattern Recognition*.

Hu, R. and R. I. Damper (2007). Optimal weighting of bimodal biometric information with specific application to audio-visual person identification. Submitted to *Information Fusion*.

Hu, R. and R. I. Damper (2006). A 'no panacea theorem' for multiple classifier combination. In *Proceedings of the 18th International Conference on Pattern Recognition, ICPR'06*, Hong Kong, China, pp. 1250–1253.

Hu, R. and R. I. Damper (2005). Fusion of two classifiers for speaker identification: removing and not removing silence. *Proceedings of the 8th International Conference on Information Fusion, FUSION'05*, Philadelphia, PA, pp. 429–436.

Bibliography

Adjoudani, A. and Benoît, C. (1995). Audio-visual speech recognition compared across two architectures. In *Proceedings of the 4th European Conference on Speech Communication and Technology*, volume 2, pages 1563–1567, Madrid, Spain.

Ahlberg, J. (2001). Using the active appearance algorithm for face and facial feature tracking. In *Proceedings of the ICCV'01 Workshop on Recognition, Analysis and Tracking of Faces and Gestures in Realtime Systems*, Vancouver, Canada.

Antos, A., Devroye, L., and Györfi, L. (1999). Lower bound for Bayes error estimation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **20**(7), 643–645.

Aronowitz, H., Burshtein, D., and Amir, A. (2005). A session-GMM generative model using test utterance Gaussian mixture modeling for speaker verification. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP'05*, volume 1, pages 733–736, Philadelphia, PA.

Bartlett, J. C. and Searcy, J. (1993). Inversion and configuration of faces. *Cognition Psychology*, **25**, 281–316.

Bartlett, M. S., Lades, H. M., and Sejnowski, T. (1998). Independent component representation for face recognition. In *Proceedings of the SPIE Symposium on Electronic Imaging: Science and Technology*, pages 528–539, San Jose, CA.

Battiti, R. and Colla, A. M. (1994). Democracy in neural nets: Voting schemes for classification. *Neural Networks*, **7**(4), 691–707.

Baum, L. E. (1972). An inequality and associated maximization technique in statistical estimation for probabilistic functions of Markov chains. *Inequalities*, **3**, 1–8.

Baum, L. E. and Egon, J. A. (1967). An inequality with applications to statistical estimation for probabilistic functions of a Markov process and to a model for ecology. *Bulletin of the American Meteorological Society*, **73**, 360–363.

Baum, L. E. and Petrie, T. (1966). Statistical inference for probabilistic functions of finite state Markov chains. *Annals of Mathematical Statistics*, **37**, 1554–1563.

Baum, L. E. and Sell, G. R. (1968). Growth functions for transformations on manifold. *Pacific Journal of Mathematics*, **27**(2), 211–227.

Baum, L. E., Petrie, T., Soules, G., and Weiss, N. (1970). A maximization technique occurring in the statistical analysis of probabilistic functions of Markov chains. *Annals of Mathematical Statistics*, **41**(1), 164–171.

Belhumeur, V., Hespanha, J., and Kriegman, D. (1997). Eigenfaces vs. Fisherfaces: Recognition using class specific linear projection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **19**(7), 711–720.

Ben-Yacoub, S., Abdeljaoued, Y., and Mayoraz, E. (1999). Fusion of face and speech data for person verification. *IEEE Transactions on Neural Networks*, **10**(5), 1065–1074.

Benediktsson, J. A. and Swain, P. H. (1992). Consensus theoretic classification methods. *IEEE Transactions on Systems, Man, and Cybernetics*, **22**(4), 688–704.

Benediktsson, J. A., Sveinsson, J. R., Ingimundarson, J. I., and Sigurdsson, H. (1997). Multistage classifiers optimized by neural networks and genetic algorithms. *Nonlinear Analysis, Theory, Methods and Applications*, **30**(3), 1323–1334.

Bengio, S. (2003). Multimodal authentication using asynchronous HMMs. In *Proceedings of the 4th International Conference on Audio and Video-based Biometric Person Authentication*, pages 770–777, Guildford, UK.

Biederman, I. and Kalocsai, P. (1998). Neural and psychophysical analysis of object and face recognition. In H. Wechsler, P. J. Phillips, V. Bruce, F. F. Soulie, and T. S. Huang, editors, *Face Recognition: From Theory to Application*, pages 3–25. Springer-Verlag, Berlin, Germany.

Bishop, C. M. (1995). *Neural Networks for Pattern Recognition*. Oxford University Press, Oxford, UK.

Blanz, V. and Vetter, T. (2003). Face recognition based on fitting a 3D morphable model. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **25**(9), 1063–1074.

Blanz, V., Romdhani, S., and Vetter, T. (2002). Face identification across different poses and illuminations with a 3D morphable model. In *Proceedings of the 5th International Conference on Automatic Face and Gesture Recognition, FG'02*, pages 202–207, Washington, DC.

Boë, L. J. (2000). Forensic voice identification in France. *Speech Communication*, **31**(2-3), 205–224.

Bonastre, J. F., Bimbot, F., Boë, L. J., Campbell, J. P., Reynolds, D. A., and Margrin-Chagnolleau, I. (2003). Person authentication by voice: A need for caution. In *Proceedings of the European Conference on Speech Communication and Technology, EUROSPEECH'03*, pages 33–36, Geneva, Switzerland.

Bradski, G. R. and Pisarevsky, V. (2000). Intel's computer vision library: Applications in calibration, stereo, segmentation, tracking, gesture, face and object recognition. In *IEEE International Conference on Computer Vision and Patter Recognition, CVPR'00*, pages 796–797, Hilton Head Island, South Carolina.

Broeders, A. P. A. (2001). Forensic speech and audio analysis forensic linguistics. In *Proceedings of the 13th International Forensic Science Symposium*, volume D2, pages 54–84, Lyon, France.

Bronstein, A. M., Bronstein, M. M., and Kimmel, R. (2005). Three-dimensional face recognition. *International Journal of Computer Vision*, **64**(1), 5–30.

Brunelli, R. and Falavigna, D. (1995). Person identification using multiple cues. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **17**(10), 955–966.

Buhmann, J., Lades, M., and Malsburg, C. V. D. (1990). Size and distortion invariant object recognition by hierarchical graph matching. In *Proceedings of the International Joint Conference on Neural Networks, IJCNN'90*, pages 411–416, San Diego, CA.

Campbell, J. P. (1997). Speaker recognition: A tutorial. *Proceedings of the IEEE*, **85**(9), 1437–1462.

Campbell, W. M. (2002). Generalized linear discriminant sequence kernels for speaker recognition. In *Proceedings of the International Conference on Acoustics, Speech and Signal Processing, ICASSP'02*, pages 161–164, Orlando, FL.

Campbell, W. M., Campbell, J. P., Reynolds, D. A., Singer, E., and Torres-Carrasquillo, P. A. (2006). Support vector machines for speaker and language recognition. *Computer Speech and Language*, **20**, 210–229.

Campbell, W. M., Campbell, J. P., Gleason, T. P., Reynolds, D. A., and Shen, W. (2007). Speaker verification using support vector machines and high-level features. *IEEE Transactions on Audio, Speech and Language Processing*, **15**(7), 2085–2094.

Chao, Y. H., Wang, H. M., and Chang, R. C. (2005). GMM-based Bhattacharyya kernel Fisher discriminant analysis for speaker recognition. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP'05*, volume 1, pages 649–652, Philadelphia, PA.

Chatzis, V., Bors, A. G., and Pitas, I. (1999). Multimodal decision-level fusion for person authentication. *IEEE Transactions on Systems, Man, and Cybernetics*, **29**(6), 674–680.

Chellappa, R., Wilson, C. L., and Sirohey, S. (1995). Human and machine recognition of faces: A survey. *Proceedings of the IEEE*, **83**(5), 705–740.

Chibelushi, C. C., Deravi, F., and Mason, J. S. (1993). Voice and facial image integration for speaker recognition. In *IEEE International Symposium and Multimedia Technologies and Future Applications*, Southampton, UK.

Cho, S. and Kim, J. H. (1995). Combining multiple neural networks by fuzzy integral for robust classification. *IEEE Transactions on Systems, Man, and Cybernetics*, **25**(2), 380–384.

Chui, C. K. (1992). *An Introduction to Wavelets*. Academic Press, New York, NY.

Colmenarez, A. J. and Huang, T. S. (1997). Face detection with information-based maximum discrimination. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, CVPR'97*, pages 782–787, San Juan, Puerto Rico.

Comon, P. (1994). Independent component analysis – a new concept? *Signal Processing*, **36**(3), 287–314.

Cootes, T. F., Taylor, C. J., Cooper, D. H., and Graham, J. (1995). Active shape models – their training and application. *Computer Vision and Image Understanding*, **61**(1), 38–59.

Cootes, T. F., Edwards, G. J., and Taylor, C. J. (2001). Active appearance models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **23**(6), 681–685.

Cormen, T. H., Leiserson, C. E., Rivest, R. L., and Stein, C. (2001). *Introduction to algorithms*. The MIT Press, 55 Hayward Street, Cambridge, MA.

Cover, T. M. and Thomas, J. A. (1991). *Elements of Information Theory*. John Wiley & Sons, Inc, Hoboken, NJ.

Crow, F. (1984). Summed-area tables for texture mapping. *Proceedings of SIGGRAPH*, **18**(3), 207–212.

Czyz, J., Sadeghi, M., Kittler, J., and L.Vandendorpe (2004). Decision fusion for face authentication. In *Proceedings of the International Conference on Bioinformatics and its Applications*, pages 686–693, Fort Lauderdale, FL.

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

Davison, A. C. and Hinkley, D. (1997). *Bootstrap methods and their application*. Cambridge University Press, UK.

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

Doddington, G. (1985). Speaker recognition – identifying people by their voices. *Proceedings of the IEEE*, **73**(11), 1651–1664.

Drucker, H., Cortes, C., Jackel, L. D., LeCun, Y., and Vapnik, V. (1994). Boosting and other ensemble methods. *Neural Computation*, **6**, 1289–1301.

Duin, R. P. W. (2002). The combining classifier: To train or not to train? In *Proceedings of the International Conference on Pattern Recognition, ICPR'02*, volume 2, pages 765–770, Quebec, Canada.

Dupont, S. and Luettin, J. (2000). Audio-visual speech modeling for continuous speech recognition. *IEEE Transactions on Multimedia*, **2**(3), 141–151.

Edwards, G., Taylor, C., and Cootes, T. (1998). Interpreting face images using active appearance models. In *Proceedings of the 3rd International Conference on Automatic Face and Gesture Recognition, FG'98*, pages 300–305, Nara, Japan.

Efron, B. (1979). Bootstrap methods: another look at jackknife. *The Annals of Statistics*, **7**(1), 1–26.

Erzin, E., Yemez, Y., and Tekalp, A. M. (2005). Multimodal speaker identification using adaptive classifier cascade based on modality reliability. *IEEE Transactions on Multimedia*, **7**(5), 840–852.

Etemad, K. and Chellappa, R. (1997). Discriminant analysis for recognition of human face images. *Journal of the Optical Society of America A*, **14**, 1724–1733.

Filippi, E., Costa, M., and Pasero, E. (1994). Multi-layer perceptron ensembles for increased performance and fault tolerance in pattern recognition. In *Proceedings of the IEEE International Conference on Neural Networks*, pages 2901–2906, Orlando, FL.

Fine, S., Navrátil, J., and Gopinath, R. A. (2001). A hybrid GMM/SVM approach to speaker recognition. In *Proceedings of the International Conference on Acoustics, Speech and Signal Processing, ICASSP'01*, pages 417–420, Salt Lake City, UT.

Fox, N. A., Gross, R., de Chazal, P., Cohn, J. F., and Reilly, R. B. (2003). Person identification using automatic integration of speech, lip, and face experts. In *Proceedings of the ACM SIGMM Multimedia Biometrics Methods and Applications Workshop*, pages 25–32, Berkeley, California.

Freund, Y. and Schapire, R. (1997). A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, **55**(1), 119–139.

Fu, T., Liu, X. X., Liang, L. H., Pi, X., and Nefian, A. V. (2003). Audio-visual speaker identification using coupled hidden Markov model. In *Proceedings of the IEEE International Conference on Image Processing, ICIP'03*, volume 3, pages 29–32, Barcelona, Spain.

Fukunaga, K. (1989). *Statistical Pattern Recognition*. Academic Press, New York,NY.

Furui, S. (1981). Cepstral analysis techniques for automatic speaker verification. *IEEE Transactions on Acoustics, Speech and Signal Processing*, **ASSP-29**(2), 254–272.

Furui, S. (1986). Speaker independent isolated word recognition using dynamic features of speech spectrum. *IEEE Transactions on Acoustics, Speech and Signal Processing*, **34**(1), 52–59.

Furui, S., Itakura, F., and Saito, S. (1972). Talker recognition by longtime averaged speech spectrum. *Electronics and Communications in Japan*, **55-A**(10), 54–61.

Gader, P. D., Mohamed, M. A., and Keller, J. M. (1996). Fusion of handwritten word classifiers. *Pattern Recognition Letters*, **17**(6), 557–584.

Giacinto, G., Roli, F., and Didaci, L. (2003). Fusion of multiple classifiers for intrusion detection in computer networks. *Pattern Recognition Letters*, **24**(12), 1795–1803.

Gonzalez, R. C. and Woods, R. E. (2002). *Digital Image Processing*. Prentice Hall, Upper Saddle River, NJ.

Govindaraju, V. (1996). Locating human faces in photographs. *International Journal of Computer Vision*, **19**(2), 129–146.

Granger, E. (2001). A what-and-where fusion neural network for recognition and tracking of multiple radar emitters. *Neural Networks*, **14**(3), 325–344.

Grimmett, G. R. and Stirzaker, D. R. (1992). *Probability and Random Process*. Clarendon Press, Oxford, UK.

Gunn, S. R. and Nixon, M. S. (1997). A robust snake implementation: A dual active contour. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **19**(1), 63–68.

Haralick, R. M. and Shapiro, L. G. (1992). *Computer and Robot Vision*, volume 1, chapter 2, pages 13–58. Addison-Wesley Publishing Company, Inc., Reading, MA.

Hégarat-Mascle, S. L., Bloch, I., and Vidal-Madjar, D. (1997). Application of Dempster-Shafer evidence theory to unsupervised classification in multisource remote sensing. *IEEE Transactions on Geoscience and Remote Sensing*, **35**(4), 1018–1031.

Hjelmas, E. (2001). Face detection: A survey. *Computer Vision and Image Understanding*, **83**, 236–274.

Hsu, R. L., Mottaleb, M. A., and Jain, A. K. (2002). Face detection in color images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **24**(5), 696–706.

Huang, J., Heisele, B., and Blanz, V. (2003). Component-based face recognition with 3D morphable models. In *Proceedings of the 4th International Conference on Audio- and Video-Based Biometric Person Authentication, AVBPA'03*, pages 27–34, Guildford, UK.

Jacobs, R. A. (1995). Methods for combining experts's probability assessments. *Neural Computation*, **7**(5), 867–888.

Jacobs, R. A., Jordan, M. I., Nowlan, S. J., and Hinton, G. E. (1991). Adaptive mixtures of local experts. *Neural Computation*, **3**, 79–87.

Jain, A., Nandakumar, K., and Ross, A. (2005). Score normalization in multimodal biometric systems. *Pattern Recognition*, **38**, 2270–2285.

Jain, A. K. (1989). *Fundamentals of Digital Image Processing*. Prentice Hall,Inc, Englewood cliffs, NJ.

Jeng, S. H., Liao, H. Y. M., Han, C. C., Chern, M. Y., and Liu, Y. T. (1998). Facial feature detection using geometrical face model: An efficient approach. *Pattern Recognition*, **31**(3), 273–282.

Jolliffe, I. T. (1986). *Principal Component Analysis*. Springer-Verlag, New York, NY.

Jordan, M. I. and Xu, L. (1995). Convergence results for the EM approach to mixtures of experts architectures. *Neural Networks*, **8**(9), 1409–1431.

Jutten, C. and Herault, J. (1991). Blind separation of sources, Part I: An adaptive algorithm based on neuromimetic architecture. *Signal Processing*, **24**, 1–10.

Kanade, T. (1973). *Picture Processing by Computer Complex and Recognition of Human Faces*. Ph.D. thesis, Kyoto University, Japan.

Kass, M., Witkin, A., and Terzopoulos, D. (1998). Snakes: Active contour models. *International Journal of Computer Vision*, **1**(4), 321–331.

Keller, J. M., Gader, P., Tahani, H., Chiang, J. H., and Mohamed, M. (1994). Advances in fuzzy integration for pattern recognition. *Fuzzy Sets and Systems*, **65**, 273–283.

Kelly, M. D. (1970). Visual identification of people by computer. Technical Report AI-130, Stanford AI project, Stanford, CA.

Kharroubi, J., Petrovska-Delacretaz, D., and Chollet, G. (2001). Combining GMMs with support vector machines for text-independent speaker verification. In *Proceedings of the 7th European Conference on Speech Communication and Technology, EUROSPEECH'01*, pages 1757–1760, Aalborg, Denmark.

Kirby, M. and Sirovich, L. (1990). Application of the Karhunen-Loève procedure for the characterization of human faces. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **12**(1), 103–108.

Kittler, J., Hatef, M., Duin, R. P. W., and Matas, J. (1998a). On combining classifiers. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **20**(3), 226–239.

Kittler, J., Hatef, M., Duin, R. P. W., and Matas, J. (1998b). On combining classifiers. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **20**(3), 226–239.

Kruger, N., Potzsch, M., and Malsburg, C. V. D. (1997). Determination of face position and pose with a learned representation based on labelled graphs. *Image and Vision Computing*, **15**(8), 665–673.

Kuncheva, L. I. (2004). *Combining Pattern Classifiers–Methods and Algorithms*, chapter 4, pages 111–149. John Wiley & Sons, Inc.

Kuncheva, L. I. and Jain, L. C. (2000). Designing classifier fusion systems by genetic algorithms. *IEEE Transactions on Evolutionary Computation*, **4**(4), 327–336.

Kuncheva, L. I., Bezdek, J. C., and Duin, R. P. W. (2001). Decision templates for multiple classifier fusion: An experimental comparison. *Pattern Recognition*, **34**(2), 229–314.

Lades, M., Vorbruggen, J., Buhmann, J., Lange, J., Malburg, C. V. D., and Wurtz, R. (1993). Distortion invariant object recognition in dynamic link architecture. *IEEE Transactions on Computers*, **42**(3), 300–311.

Lam, K. M. and Yan, H. (1996). Locating and extracting the eye in human face images. *Pattern Recognition*, **29**(5), 771–779.

Lam, L. and Suen, C. Y. (1995). Optimal combination of pattern classifiers. *Pattern Recognition Letters*, **16**(9), 945–954.

Lappas, P., Carter, J. N., and Damper, R. I. (2002). Robust evidence-based object tracking. *Pattern Recognition Letters*, **23**(1–2), 253–260.

Lee, J., Moghaddam, B., Pfister, H., and Machiraju, R. (2004). Finding optimal views for 3D face shape modeling. In *Proceedings of the International Conference on Automatic Face and Gesture Recognition, FGR'04*, pages 31–36, Seoul, Korea.

Lew, M. S. (1996). Information theoretic view-based and modular face detection. In *Proceedings of the IEEE International Conference on Automatic Face and Gesture Recognition*, pages 198–203, Killington, VT.

Li, K. and Wrench, J. E. (1983). An approach to text-independent speaker recognition with short utterances. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP'83*, pages 555–558, Boston, MA.

Li, S. Z. and Jain, A. K., editors (2004). *Handbook of Face Recognition*. Springer, New York, NY.

Li, S. Z. and Lu, J. (1999). Face recognition using the nearest feature line method. *IEEE Transactions on Neural Networks*, **10**(2), 439–443.

Li, S. Z. and Zhang, Z. (2004). Floatboost learning and statistical face detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **26**(9), 1112–1123.

Lienhart, R., Kuranov, A., and Pisarevsky, V. (2002). Empirical analysis of detection cascades of boosted classifiers for rapid object detection. Technical report, Microcomputer Research Lab, Intel Corporation, Santa Clara, CA.

Lin, S. H., Kung, S. Y., and Lin, L. J. (1997). Face recognition/detection by probabilistic decision-based neural network. *IEEE Transactions on Neural Networks*, **8**(1), 114–132.

Liu, C. and Wechsler, H. (2000). Evolutionary pursuit and its application to face recognition. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, **22**(6), 570–582.

Lucey, S., Chen, T., Sridharan, S., and Chandran, V. (2005). Integration strategies for audio-visual speach processing: Applied to text-dependent speaker recognition. *IEEE Transactions on Multimedia*, **7**(3), 495–506.

Luettin, J. (1997). *Visual Speech and Speaker Recognition*. Ph.D. thesis, University of Sheffield, UK.

Maison, B., Neti, C., and Senior, A. (1999). Audio-visual speaker recognition for video broadcast news: some fusion techniques. In *Proceedings of the IEEE Workshop on Multimedia Signal Processing*, pages 161–167, Copenhagen, Denmark.

Mallet, S. (1999). *A Wavelet Tour of Signal Processing*. Academic Press, New York, NY.

Markel, J., Oshika, B., and Gray, A. (1977). Long-term feature averaging for speaker recognition. *IEEE Transactions on Acoustics, Speech and Signal Processing*, **25**, 330–337.

Martinkauppi, B. (2002). *Face colour under varying illumination- analysis and applications*. Ph.D. thesis, Department of Electrical and Information Engineering, University of Oulu, Finland.

Matsui, T. and Furui, S. (1991). A text-independent speaker recognition method robust against utterance variations. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP'91*, pages 377–380, Toronto, Canada.

Maurer, T. and Malsburg, C. V. D. (1996). Single-view based recognition of faces rotated in depth. In *Proceedings of the International Conference on Automatic Face and Gesture Recognition, FG'96*, pages 176–181, Killington, VT.

McLachlan, G. (1988). *Mixture Models*. Marcel Dekker, New York, NY.

Messer, K., Matas, J., Kittler, J., Luettin, J., and Maitre, G. (1999). XM2VTSDB: The extended M2VTS database. In *Proceedings of 2nd International Conference on Audio and Video-based Biometric Person Authentication, AVBPA'99*, pages 72–77, Washington, DC.

Miller, B. (1994). Vital signs of identity. *IEEE Spectrum*, **31**(2), 22–30.

Minaei-Bidgoli, B., Kortemeyer, G., and Punch, W. F. (2004). Optimizing classification ensembles via a genetic algorithm for a web-based educational system. In *Proceedings of the Joint IAPR International Workshops on Syntactical and Structural Pattern Recognition (SSPR) and Statistical Pattern Recognition (SPR)*, pages 397–406, Lisbon, Portugal.

Mitchell, T. M. (1997). *Machine Learning*. McGraw-Hill Companies, Inc, New York, NY.

Moghaddam, B. and Pentland, A. P. (1997). Probabilistic visual learning for object representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **19**(7), 696–710.

Mpiperis, I., Malassiotis, S., and Strintzis, M. G. (2007). 3-D face recognition with the geodesic polar representation. *IEEE Transactions on Information Forensics and Security*, **2**(3), 537–547.

Naik, J. M., Netsch, L. P., and Doddington, G. R. (1989). Speaker verification over long-distance telephone lines. In *Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP'89*, volume 1, pages 524–527, Glasgow, Scotland.

Ney, H. (1995). On the probabilistic interpretation of neural network classifiers and discriminative training criteria. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **17**(2), 107–119.

Ng, K. C. and Abramson, B. (1992). Consensus diagnosis: A simulation study. *IEEE Transactions on Systems, Man, and Cybernetics*, **22**(5), 916–928.

Nowlan, S. J. and Hinton, G. E. (1991). Evaluation of adaptive mixtures of competing experts. In R. P. Lippmann, J. E. Moody, and D. S. Touretzky, editors, *Neural Information Processing Systems*, volume 3, pages 774–780. Springer-Verlag, Berlin, Germany.

Okada, K., Steffans, J., Maurer, T., Hong, H., Elagin, E., Neven, H., and Malsburg, C. V. D. (1998). The Bochum/USC face recognition system and how it fared in FERET phase III test. In H. Wechsler, P. J. Phillips, V. Bruce, F. F. Soulie, and T. S. Huang, editors, *Face Recognition: From Theory to Application*, pages 186–205. Springer-Verlag, Berlin, Germany.

Oltean, M. (2004). Searching for a practical evidence for the no free lunch theorems. In *BioInspired Approaches to Advanced Information Technology, LNCS 3141*, Lausanne, Switzerland. Springer-Verlag.

Oppenheim, A. V., Schafer, R. W., and Buck, J. R. (1999). *Discrete-time Signal Processing*. Prentice Hall, Upper Saddle River, NJ.

O'Shaugnessy, D. (1986). Speaker recognition. *IEEE ASSP Magazine*, **3**(4), 4–17.

Osuna, E., Freund, R., and Girosi, F. (1997). Training support vector machines: An application to face detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, CVPR'97*, pages 130–136, San Juan, Puerto Rico.

Papageorgiou, C., Oren, M., and Poggio, T. (1998). A general framework for object detection. In *International Conference on Computer Vision, ICCV98*, pages 555–562, Bombay, India.

Phillips, P. J. (1998). Support vector machines applied to face recognition. In *Proceedings of the Conference on Advances in Neural Information Processing Systems, NIPS'98*, pages 803–809, Denver, CO.

Phillips, P. J., Moon, H., Rizvi, S. A., and Rauss, P. J. (2000). The FERET evaluation methodology for face-recognition algorithms. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **22**(10), 1090–1104.

Poritz, A. (1982). Linear predictive hidden Markov models and the speech signal. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP'82*, pages 524–527, Paris, France.

Quatieri, T. F. (2001). *Discrete-time Speech Signal Processing: Principles and Practice.* Prentice Hall, Upper Saddle River, NJ.

Rabiner, L. R. and Juang, B. H. (1993). *Fundamentals of Speech Recognition.* Prentice Hall, Upper Saddle River, NJ.

Rabiner, L. R. and Sambur, M. R. (1975). An algorithm for determining the endpoints of isolated utterances. *Bell Systems Technical Journal*, **54**(2), 297–315.

Rajagopalan, A., Kumar, K., Karlekar, J., Manivasakan, R., Patil, M., Desai, U., Poonacha, P., and Chaudhuri, S. (1998). Finding faces in photographs. In *Proceedings of the IEEE International Conference on Computer Vision, ICCV'98*, pages 640–645, Bombay, India.

Reynolds, D. A. and Rose, R. C. (1995). Robust text-independent speaker identification using Gaussian mixture models. *IEEE Transactions on Speech and Audio Processing*, **3**(1), 72–83.

Reynolds, D. A., Quatieri, T. F., and Dunn, R. B. (2000). Speaker verification using adapted Gaussian mixture models. *Digital Signal Processing*, **10**, 19–21.

Rizvi, S. A., Phillips, P. J., and Moon, H. (1998). The FERET verification testing protocol for face recognition. Technical Report NISTIR-6281, National Institute of Standards and Technology, NIST.

Rosenberg, A. E. and Soong, F. K. (1987). Evaluation of a vector quantization talker recognition system in text dependent and text independent modes. *Computer Speech and Language*, **22**, 143–157.

Rosenberg, A. E., Lee, C. H., and Gokcen, S. (1991). Connected word talker verification using whole word hidden Markov models. In *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP'91*, pages 381–384, Toronto, Canada.

Rowley, H. A., Baluja, S., and Kanade, T. (1998). Neural network-based face detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **20**(1), 23–38.

Samaria, F. (1994). *Face Recognition Using Hidden Markov Models.* Ph.D. thesis, University of Cambridge, UK.

Samaria, F. and Young, S. (1994). HMM based architecture for face identification. *Image Visual Computation*, **12**, 537–583.

Sammon, J. W. (11969). A nonlinear mapping for data structure analysis. *IEEE Transactions on Computers*, **C-18**(5), 401–409.

Sanderson, C. and Paliwal, K. K. (2003). Noise compensation in a person verification system using face and multiple speech features. *Pattern Recognition*, **36**(2), 293–302.

Savic, M. and Gupta, S. (1990). Variable parameter speaker verification system based on hidden Markov modeling. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP'90*, pages 281–284, Albuquerque, NM.

Schapire, R. E., Freund, Y., Bartlett, P., and Lee, W. S. (1997). Boosting the margin: A new explanation for the effectiveness of voting methods. In *Proceedings of the Fourteenth International Conference on Machine Learning*, pages 322–330, Nashville, TN.

Schneiderman, H. and Kanade, T. (1998). Probabilistic modeling of local appearance and spatial relationships for object recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, CVPR'98*, pages 45–51, Santa Barbara, CA.

Schölkopf, B., Smola, A., and Muller, K. R. (1998). Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation*, **10**(5), 1299–1319.

Sehgal, M. S. B., Gondal, I., and Dooley, L. (2004). Support vector machine and generalized regression neural network based classification fusion models for cancer diagnosis. In *International Conference on Hybrid Intelligent Systems*, pages 49–54, Kitakyushu, Japan.

Shi, Y., Ji, H., and Gao, X. (2005). A radar target multi-feature fusion classifier based on rough neural network. In *International Symposium on Neural Networks*, pages 375–380, Chongqing, China.

Sirovich, L. and Kirby, M. (1987). Low-dimensional procedure for the characterization of human faces. *Journal of Optical Society of America*, **4**, 519–524.

Solewicz, Y. A. and Koppel, M. (2004). Enhanced fusion methods for speaker verification. In *Proceedings of the International Conference on Speech and Computer, SPECOM'04*, pages 388–392, St. Petersburg, Russia.

Sukno, F. M., Ordás, S., Butakoff, C., Cruz, S., and Frangi, A. F. (2007). Active shape models with invariant optimal features: application to facial analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **29**(7), 1105–1117.

Sung, K. K. and Poggio, T. (1998). Example-based learning for view-based human face detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **20**(1), 39–51.

Swets, D. L. and Weng, J. (1996). Using discriminant eigenfeatures for image retrieval. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **18**(8), 831–836.

Tahani, H. and Keller, J. M. (1990). Information fusion in computer vision using the fuzzy integral. *IEEE Transactions on Systems, Man, and Cybernetics*, **20**(3), 733–741.

Tanaka, J. W. and Farah, M. J. (1993). Parts and wholes in face recognition. *Quarterly Journal of Psychology*, **46A**(2), 225–245.

Tishby, N. Z. (1991). On the application of mixture AR hidden Markov models to text independent speaker recognition. *IEEE Transactions on Signal Processing*, **39**(3), 563–570.

Toh, K. A. and Yau, W. Y. (2004). Combination of hyperbolic functions for multimodal biometrics data fusion. *IEEE Transactions on System, Man, and Cybernetics–Part B: Cybernetics*, **34**(2), 1196–1209.

Turk, M. A. and Pentland, A. P. (1991). Face recognition using eigenfaces. In *Proceedings of the International Conference on Pattern Recognition, ICPR'91*, pages 586–591, Atlantic City, NJ.

Ueda, N. (2000). Optimal linear combination of neural networks for improving classification performance. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **22**(2), 207–215.

Valentin, D., Abdi, H., O'Toole, A. J., and Cottrell, G. (1994). Connectionist models of face processing: A survey. *Pattern Recognition*, **27**(9), 1209–1230.

Vapnik, V. N. (1998). *Statistical Learning Theory*. Wiley, New York, NY.

Viola, P. and Jones, M. (2001). Rapid object detection using a boosted cascade of simple features. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, CVPR'01*, pages 12–14, Kauai, HI.

Viola, P. and Jones, M. (2002). Robust real-time face detection. *International Journal of Computer Vision*, **57**(2), 137–154.

Viola, P. and Jones, M. J. (2004). Robust real-time face detection. *International Journal of Computer Vision*, **57**(2), 137–154.

Voiers, W. D. (1964). Perceptual bases of speaker identity. *Journal of Acoustical Society of America*, **36**, 1065–1073.

Wan, K., Lam, K., and Ng, K. (2005). An accurate active shape model for facial feature extraction. *Pattern Recognition Letters*, **26**, 2409–2423.

Wan, V. and Renals, S. (2005). Speaker verification using sequence discriminant support vector machines. *IEEE Transactions on Speech and Audio Processing*, **13**(2), 203–210.

Wang, Y. and Yuan, B. (2001). A novel approach for human face detection from color images under complex background. *Pattern Recognition*, **34**(10), 1983–1992.

Wark, T. (2000). *Multi-modal speech processing for automatic speaker recognition*. Ph.D. thesis, School of Electrical & Electronic Systems Engineering, Queensland University of Technology, Brisbane, Australia.

Wark, T., Sridharan, S., and Chandran, V. (1999). Robust speaker verification via fusion of speech and lip modalities. In *Proceedings of the International Conference on Acoustics, Speech and Signal Processing*, volume 6, pages 3061–3064, Phoenix.

Webb, A. (2002). *Statistical Pattern Recognition*. John Wiley & Sons, Inc, Hoboken, NJ.

Weyrauch, B., Heisele, B., Huang, J., and Blanz, V. (2003). Component-based face recognition with 3d morphable models. In *Proceedings of the first IEEE Workshop on Face Processing in Video, FPIV'04*, pages 1–5, Washington, DC.

Wilson, S. G. (1996). *Digital Modulation and Coding*, chapter 2, pages 18–140. Prentice Hall.

Wiskott, L., Fellous, J. M., Krüger, N., and Malsburg, C. (1997). Face recognition by elastic bunch graph matching. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **19**(7), 775–779.

Wolpert, D. H. (2001). The supervised learning no-free-lunch theorems. In *Proceedings of the 6th Online World Conference on Soft Computing in Industrial Applications.*

Wolpert, D. H. and Macready, W. G. (1995). No free lunch theorems for search. Technical Report SFI-TR-95-02-010, Santa Fe Institute, Santa Fe, NM.

Wolpert, D. H. and Macready, W. G. (1997). No free lunch theorems for optimization. *IEEE Transactions on Evolutionary Computation*, **1**(1), 67–82.

Woods, K., Kegelmeyer, W. P., and Bowyer, K. (1997). Combination of multiple classifiers using local accuracy estimates. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **19**(4), 405–410.

Xu, L., Krzyzak, A., and Suen, C. Y. (1992). Methods of combining multiple classifiers and their applications to handwriting recognition. *IEEE Transactions on Systems, Man, and Cybernetics*, **22**(3), 418–435.

Yang, G. and Huang, T. S. (1994). Human face detection in complex background. *Pattern Recognition*, **27**(1), 53–63.

Yang, M. H. (2002). Kernel eigenfaces vs. kernel Fisherfaces: face recognition using kernel methods. In *Proceedings of the IEEE International Conference on Face and Gesture Recognition*, pages 215–220, Washington, DC.

Yang, M. H., Kriegman, D. J., and Ahuja, N. (2002). Detection faces in images: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **24**(1), 34–58.

Yow, K. C. and Cipolla, R. (1996). A probabilistic framework for perceptual grouping of features for human face detection. In *Proceedings of the International Conference on Automatic Face and Gesture Recognition, FG'96*, pages 16–22, Killington, VT.

Yuille, A. L., Cohen, D. S., and linan, P. W. H. (1989). Feature extraction from faces using deformable templates. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 104–109, San Diego, CA.

Zarit, B. D., Super, B. J., and Quek, F. K. H. (1999). Comparison of five color models in skin pixel classification. In *Proceedings of the IEEE ICCV Workshop on Recognition, Analysis and Tracking of Faces and Gestures in Real-time Systems*, pages 58–63, Corfu, Greece.

Zhao, W., Chellappa, R., and Krishnaswamy, A. (1998). Discriminant analysis of principal components for face recognition. In *Proceedings of the International Conference on Automatic Face and Gesture Recognition, FG'98*, pages 336–341, Nara, Japan.

Zhao, W., Chellappa, R., Phillips, P. J., and Rosenfeld, A. (2003). Face recognition: a literature survey. *ACM Computing Surveys*, **35**(4), 399–459.

Zheng, Y. and Yuan, B. (1988). Text-dependent speaker identification using circular hidden Markov models. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP'88*, pages 580–582, New York, USA.

Zwicker, E., Flottorp, G., and Stevens, S. S. (1957). Critical bandwidth in loudness summation. *Journal of Acoustical Society of America*, **29**, 548–557.