Off-line Signature Verification

by

Johannes Coetzer

Dissertation approved for the degree of

Doctor of Philosophy

in

Applied Mathematics

at the

University of Stellenbosch

Promoters: Prof. B.M. Herbst

Prof. J.A. du Preez

April 2005

Declaration

I, the undersigned, hereby declare that the work contained in this dissertation is my own original work and that I have not previously in its entirety or in part submitted it at any university for a degree.

Signature: _____ Date: _____

Abstract

A great deal of work has been done in the area of *off-line signature verification* over the past two decades. Off-line systems are of interest in scenarios where only hard copies of signatures are available, especially where a large number of documents need to be authenticated. This dissertation is inspired by, amongst other things, the potential financial benefits that the automatic clearing of *cheques* will have for the banking industry. The purpose of this research is to develop a *novel*, *accurate* and *efficient* off-line signature verification system. In this dissertation two systems are developed. The first system uses *dynamic time warping (DTW)* to match a test signature with an appropriate reference signature. The second system represents each writer's signature with a *hidden Markov model (HMM)*. Both of these systems use the *discrete Radon transform (DRT)* to extract a sequence of feature vectors from a signature. The HMM-based system achieves rotation invariance in an efficient and robust way. No alignment of observation sequences is required. In order to ensure rotation invariance, the DTW-based system must first align a test sequence with an appropriate reference sequence, before they are matched. As a result, the HMM-based system is computationally more efficient than the DTW-based system. Experiments are conducted on two different data sets. The systems perform almost equally well, when they are applied to the "Stellenbosch data set". Equal error rates of approximately 18% and 4.5% respectively, are achieved when only skilled forgeries and only casual forgeries are considered. When the HMM-based system is applied to "Dolfing's data set" of dynamic signatures, the results compare well with an algorithm by Dolfing which only considers the spatial coordinates of each dynamic signature. In this dissertation it is also demonstrated that the HMM-based system performs better than a typical human being. Existing systems utilise either a technique or features that are fundamentally very different from those used in this dissertation. This is especially the case for the HMM-based system. This implies that it is very likely that a combination of any existing system and the HMM-based system developed in this dissertation, will result in a superior merged system.

Opsomming

Heelwat werk is die afgelope twee dekades op die gebied van *statiese handtekeningverifikasie* gedoen. Statiese stelsels is nuttig in situasies waar daar slegs harde kopieë van handtekeninge beskikbaar is, veral waar die egtheid van 'n groot aantal dokumente getoets moet word. Hierdie werk word onder meer gedryf deur die potensiële finansiële voordele wat die outomatiese verifikasie van *tjeks* vir die bankwese sal inhou. Die doel van hierdie navorsing is om 'n *oorspronklike*, *akkurate* en *berekeningseffektiewe* statiese handtekeningverifikasiestelsel te ontwikkel. Daar is twee stelsels in hierdie tesis ontwikkel. Die eerste stelsel gebruik *dinamiese tydsverbuiging (DTW)* om 'n toetshandtekening met 'n geskikte verwysingshandtekening te vergelyk. Die tweede stelsel stel elke skrywer se handtekening met 'n *verskuilde Markov-model (HMM)* voor. Albei hierdie stelsels gebruik die *diskrete Radon-transform (DRT)* om 'n ry kenmerkvektore vanuit 'n handtekening te onttrek. Daar word aangetoon dat die HMM-gebasseerde stelsel rotasie-invariansie op 'n effektiewe en robuuste wyse bewerkstellig. Geen oplyning van rye kenmerkvektore is nodig nie. Ten einde rotasie-invariansie te bewerkstellig, moet die DTW-gebasseerde stelsel eers 'n toetsry met 'n geskikte verwysingsry oplyn, voordat hulle vergelyk word. Die HMM-gebaseerde stelsel is dus meer berekeningseffektief as die DTW-gebasseerde stelsel. Eksperimente word op twee verskillende datastelle uitgevoer. Vir die "Stellenbosch datastel" vaar die stelsels ongeveer ewe goed. Gelyke foutkoerse van ongeveer 18% en 4.5% onderskeidelik, word vir hoë en lae kwaliteit vervalsings verkry. Wanneer die HMM-gebaseerde stelsel op "Dolfing se datastel" met dinamiese handtekeninge toegepas word, vergelyk die resultate goed met dié van een van Dolfing se algoritmes wat slegs die ruimtelike koördinate van 'n dinamiese handtekening gebruik. Daar word ook aangetoon dat die HMM-gebaseerde stelsel beter vaar as 'n tipiese mens. Bestaande stelsels gebruik óf tegnieke óf kenmerke wat fundamenteel baie van die kenmerke of tegnieke, wat in hierdie tesis gebruik word, verskil. Dit is veral die geval vir die HMM-gebasseerde stelsel. Dit kom dus daarop neer dat, wanneer enige bestaande stelsel met die HMM-gebasseerde stelsel, wat in hierdie tesis ontwikkel is, gekombineer word, die gekombineerde stelsel hoogs waarskynlik beter sal wees.

Acknowledgments

I would like to extend my sincere gratitude to the following persons and institutions for enabling me to successfully complete this dissertation:

- My promoters, Ben Herbst and Johan du Preez. I got to know Ben when I was a pre-graduate student at the University of the Free State. He sparked my interest in pursuing an academic career and got me interested in image processing and pattern recognition. Ben is an enthusiastic supervisor who constantly comes up with new ideas. Johan joined this research effort at a later stage. His expertise and experience in designing hidden Markov models proved invaluable. I am also deeply indebted to Johan for allowing me to experiment with (and hopefully contribute to) *Patrec*. Patrec is a generic pattern recognition system that Johan primarily developed for speech recognition. Many of Johan's students contributed to this system over the years.

- The University of Stellenbosch and the University of the Free State, for their financial support, and my colleagues at these institutions for their contribution towards my development as lecturer and researcher.

- Hans Dolfing, who allowed me to use his database of dynamic signatures. Philips Research enabled the data collection.

- The students at the University of the Free State (and some other individuals) who provided me with their signatures. They also forged the signatures of other writers and assisted me in the scanning process.

- The lecturers, graduate students and other staff members from the Department of Applied Mathematics and the Department of Electrical and Electronic Engineering at the University of Stellenbosch. Each of these individuals authenticated a total of 765 signatures by hand. Their effort played a significant role in placing the performance of the systems developed in this dissertation into perspective.

- The examiners, for their constructive criticism, comments and suggestions.

- My parents, for their guidance, love and support.

Table of Contents

List of Tables

List of Figures

List of Symbols

The section, in which each symbol is introduced, is given in brackets. Note that this list *does not* include the symbols introduced in appendices A and B.

Patterns and pattern classes	
$'$	Transpose
T	Length of observation sequence (section 1.2.5)
d	Feature vector dimension (section 1.2.5)
x	Feature (section 3.1)
$\mathbf{x} = [x_1, x_2, \ldots, x_d]'$	Continuous observation (feature vector) (section 3.1)
Ω	Number of pattern classes (section 3.1)
ω	Pattern class (section 3.1)
N_j	Number of training samples for pattern class ω_j (section 3.1)
$\mathbf{x}_{\text{Test}}$	Test vector (section 3.1)
$\mathbf{X}_1^T = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_T\}$	Continuous observation sequence (section 4.3.1)
$\mathbf{x}_{\text{Ref}}$	Reference vector (section 5.3.1)
$\mathbf{X}_{\text{Test}}$	Test sequence (section 5.4)
$\mathbf{X}_{\text{Ref}}$	Reference sequence (section 5.4)
N_ω	Number of training samples for pattern class ω (section 5.5)

Probability density functions	
$f(\cdot)$	Probability density function (section 3.1)
$\boldsymbol{\mu}_j$	Mean vector for pattern class ω_j (section 3.1)
$\boldsymbol{\Sigma}_j$	Covariance matrix for pattern class ω_j (section 3.1)
σ_j^2	Variance of the jth feature (section 3.1)
σ_{ij}	Correlation between the ith and jth features (section 3.1)
$D_{\text{Eucl}}(\cdot)$	Euclidean distance (section 3.2)
$D_{\text{Mah}}(\cdot)$	Mahalanobis distance (section 3.3)

Discrete Radon transform	
Ψ	Total number of pixels in an image (section 4.2)
I_i	Intensity of the ith pixel (section 4.2)
N_Θ	Number of angles (projections) (section 4.2)
N_φ	Number of beams per angle (section 4.2)
R_j	The jth beam-sum (section 4.2)
w_{ij}	Contribution of the ith pixel to the jth beam-sum (section 4.2)

Dynamic time warping	
θ_r	Angle (in a clockwise or counter-clockwise direction) through which a sample sequence is allowed to rotate with respect to a template sequence (section 4.3.2)
K	Number of transitions in optimal path (section 5.3.2)
H_{Vec}	Bandwidth to which the search is restricted when two feature vectors are aligned (section 5.3.2)
$\leftarrow$	Optimal preceding node (section 5.3.2)
H_{Seq}	Bandwidth to which the search is restricted when two observation sequences are aligned (section 5.4)
$H_{\text{Seq}}^{\theta_r}$	Bandwidth to which the search has to be restricted in order to only provide for rotations through an angle of θ_r (in a clockwise or counter-clockwise direction), with respect to a template sequence (section 5.4)

Hidden Markov models	
λ	Hidden Markov model (section 6.3)
N	Number of states (section 6.3)
$\mathbf{S} = \{s_1, s_2, \ldots, s_N\}$	Individual states (section 6.3)
$\boldsymbol{\pi} = \{\pi_1, \pi_2, \ldots, \pi_N\}$	Initial state distribution (section 6.3)
$\mathbf{A} = \{a_{i,j}\}$	State transition probability distribution (section 6.3)
q_t	State at time t (section 6.3)
ℓ	Number of allotted forward links (section 8.4.2)

Similarity and distance measures	
$D_{\text{Node}}(i,j)$	Distance between features i and j (node-based cost) (section 5.3.2)
$D_{\text{Node}}^{(\text{Part})}(i,j)$	Distance associated with the partial optimal path that terminates at node (i,j) (section 5.3.2)
$D_{\text{Node}}^{(\text{Compl})}(\mathbf{x}_i,\mathbf{x}_j)$	Dynamic time warping-based distance between feature vectors $\mathbf{x}_i$ and $\mathbf{x}_j$ (section 5.3.2)
$D_{\text{NODE}}(\mathbf{x}_i,\mathbf{x}_j)$	Node-based cost when comparing feature vectors $\mathbf{x}_i$ and $\mathbf{x}_j$ (section 5.4)
$D(\mathbf{X}_i,\mathbf{X}_j)$	Dynamic time warping-based distance between observation sequences $\mathbf{X}_i$ and $\mathbf{X}_j$ (section 5.4)
$f(\mathbf{x}\|s_j,\lambda)$	The probability density function, which quantifies the similarity between a feature vector $\mathbf{x}$ and the state s_j (section 6.3)
$f(\mathbf{X}\|\lambda)$	The probability density function, which quantifies the similarity between observation sequence $\mathbf{X}$ and hidden Markov model λ (section 6.3)
$D(\mathbf{X},\lambda)$	Distance between observation sequence $\mathbf{X}$ and hidden Markov model λ (section 6.4.3)

Verification	
μ_ω	Mean of the distances between a set of training patterns and the signature model for writer ω (section 5.5)
σ_ω	Standard deviation of the distances between a set of training patterns and the signature model for writer ω (section 5.5)
τ	Global threshold value (section 7.3)

Other symbols	
X	Random variable that counts the number of successes in a number of independent trials, where the experiment follows a binomial distribution (section 8.6.2)
N_f	Number of floating point operations required to obtain the dynamic time warping-based distance between two feature vectors (section 9.3.1)

List of Acronyms

AER	Average error rate
CCW	Counter-clockwise
CW	Clockwise
DRT	Discrete Radon transform
DTW	Dynamic time warping
EER	Equal error rate
FA	False acceptance
FAR	False acceptance rate
flops	Floating point operations
FR	False rejection
FRR	False rejection rate
HMM	Hidden Markov model
NN	Neural network
PDF	Probability density function
ROC	Receiver operating characteristic
SDC	Simple distance classifier
SVM	Support vector machine

Chapter 1

Introduction

1.1 Background

The purpose of this research is to develop a system that automatically classifies handwritten signature images as authentic or fraudulent, with as few misclassifications as possible. At the same time, the processing requirements must be reasonable, so as to make the adoption of such an automated system economically viable.

This dissertation is inspired by, amongst other things, the potential financial benefits that the automatic clearing of cheques will have for the banking industry. In South Africa, banks still process millions of cheques daily. Usually, only those cheques of which the amount exceeds a certain threshold, are verified manually by an operator. This is a cumbersome process that has to be completed within a limited time period. It is therefore important that an automated system's performance is comparable to that of a human being. A system capable of screening casual forgeries (see section 1.2.8) should already prove beneficial. In fact, most forged cheques contain forgeries of this type.

In this dissertation two systems (see section 1.5) that automatically authenticate documents based on the owner's handwritten signature are developed, that is a dynamic time warping-based (DTW-based) system and a hidden Markov model-based (HMM-based) system. It should be noted that these systems assume that the signatures have already been extracted from the documents. Methods for extracting signature data from cheque backgrounds can be found in papers by Djeziri et al. (1998), Koerich et al. (1997) and Santos et al. (1997). These systems will assist commercial banks in the process of screening cheques and are not intended to replace the manual screening of cheques entirely. Those cheques of which the signatures do not sufficiently match a model of the owner's genuine signature, are provisionally rejected. Generally these rejected cheques will constitute a small percentage of the total number of cheques processed daily, and only these cheques are selected for manual screening.

In the next two sections we give some recent statistics on check fraud and briefly discuss a recent example.

1.1.1 Cheque fraud statistics

Despite an increasing number of electronic alternatives to paper cheques, fraud perpetrated at financial institutions in the United States has become a national epidemic.

In their Cheque Fraud Survey Reports of 1998 and 2000, the American Bankers Association (1998, 2000c) points out that "Paper-based cheque fraud costs financial institutions more than $500 million annually. ... The major components of cheque fraud losses to banks are forged maker's signature and counterfeit cheques. ... In 1999 $2.2 billion worth of cheque fraud was attempted against financial institutions."

According to the American Bankers Association (2000b), cheque fraud is still growing at an alarming rate: "In 1999 there was $679 million in actual losses and $1.5 billion worth of attempted fraud (compared to $512 million in actual losses and $579 million in losses avoided in 1997)." This is echoed in the National Cheque Fraud Center Report of 2000, which states that "Cheque fraud and counterfeiting are among the fastest-growing crimes affecting the nation's financial system, producing estimated annual losses exceeding $10 billion with the number continuing to rise at an alarming rate each year." The American Bankers Association (2000a) provides the following statistics: "The overall rise in fraud was up 32% from 1997 to 1999. ... Fraudulent activity at banks with less than $500 million of assets rose 35%, at banks with $500 million to $50 billion of assets rose 46%, and at banks with assets of $50 billion or more increased by 27%."

Cheque fraud is not restricted to the United States. Interpol estimates that 46% of banking fraud worldwide is cheque fraud. In South Africa the figure is probably just over 50%. According to the banking council, R151 million was lost to cheque fraud in South Africa in 1998. The Nedbank ISS Crime Index (2000) states that "Considering the size of the banking industry, the volume of cheques processed daily and the sophisticated approach used by crime syndicates in the country (South Africa), it is unsurprising that the banking sector – like most other commercial sectors – has been affected by high levels of crime in South Africa. ... On a daily basis the industry's 4 038 branches and agencies process about 2 million ATM transactions, 1.2 million electronic funds transfers (EFT) and 1.1 million cheques. An average cheque passes through about 18 pairs of hands in the cycle from issuing to payment and return. Along with high volumes, the scope for cheque fraud is compounded by the large numbers of people involved in cheque processing."

1.1.2 Example of cheque fraud

Several South African banking institutions make little effort to verify clients' signatures on cheques. When the cheque book of a client[1] of a leading South African bank was stolen, the culprit went on a spending spree. Initially the forger went through a lot of trouble to study and practice the client's signature and as a result, high quality forgeries were produced. These signatures are generally referred to as skilled forgeries (see section 1.2.8) and are generally difficult to detect, even by experts. As time passed, the forger wrote out

[1]The client is prof. Ben Herbst, who is one of the promoters of this dissertation.

cheques for higher amounts and went through less trouble in forging the client's signature, eventually producing very low quality forgeries – it is also possible that he sold some of the blank cheques to someone else. These forgeries did not resemble the client's signature, nor the client's name, and are generally referred to as random forgeries (see section 1.2.8). Although random forgeries can be easily detected, all of these cheques were accepted by this banking institution! A valid cheque, as well as two fraudulent cheques (that are relevant to this example) are shown in figure 1.1. In section 1.2.8 we introduce the different types of forgeries that may be encountered. In the next section we supply the background to the problem and discuss some key concepts.

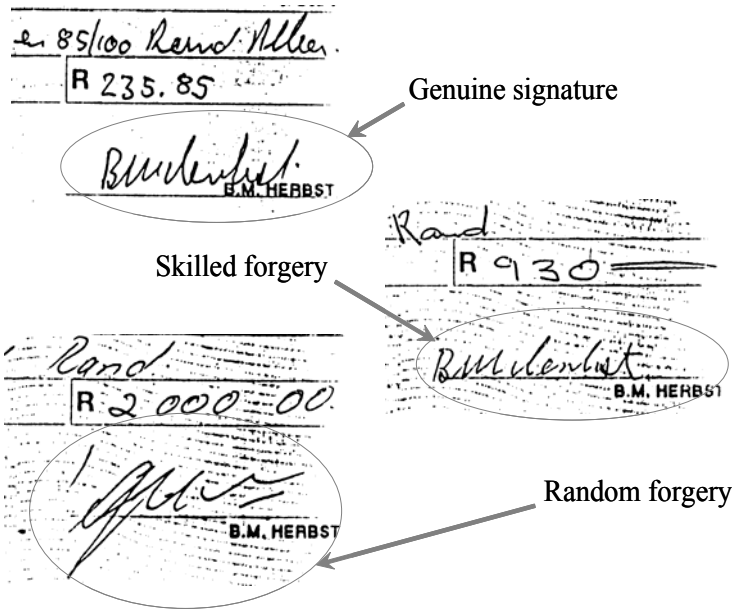


Figure 1.1: *An example of cheque fraud. A valid cheque and two fraudulent cheques.*

1.2 Key concepts

1.2.1 Pattern recognition

From an early age children are able to recognise patterns. They can, for example, discriminate between objects with different shapes. Using a broad enough interpretation, we can find pattern recognition in every intelligent activity. A pattern is any entity that can be given a name, for example a fingerprint or a handwritten digit. These entities are referred to as classes – handwritten digits, for example, consist of ten classes. The best pattern recognisers in most instances are humans, who often use this ability to make decisions. Without the human ability to recognise faces, society would be very different. The objective of pattern recognition systems is to automatically classify patterns, that

is to assign a pattern to a class, with as few misclassifications as possible. Although the human ability to recognise patterns is generally far superior to that of a computer, there are many real-life activities where machine recognition is more efficient and convenient. One of these activities involves the screening of bank cheques.

A categorisation of pattern recognition techniques is given in figure 1.2. The techniques that are relevant to this work are boxed.

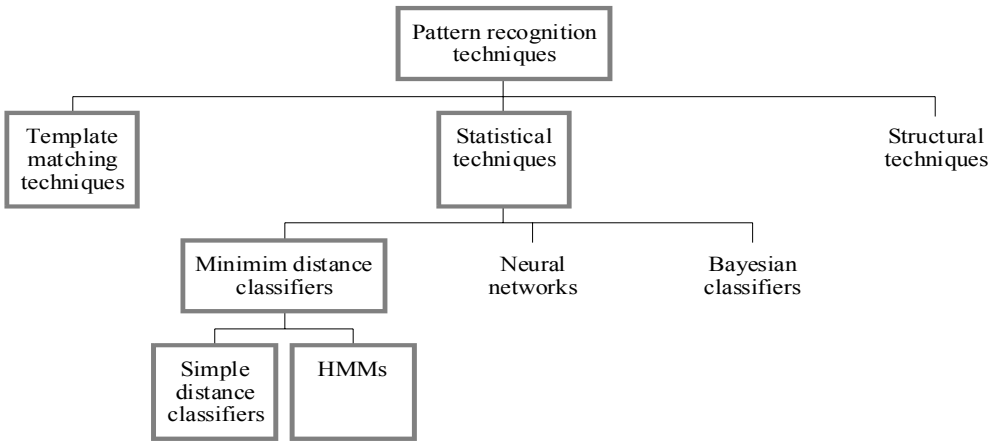


Figure 1.2*: Pattern recognition techniques.*

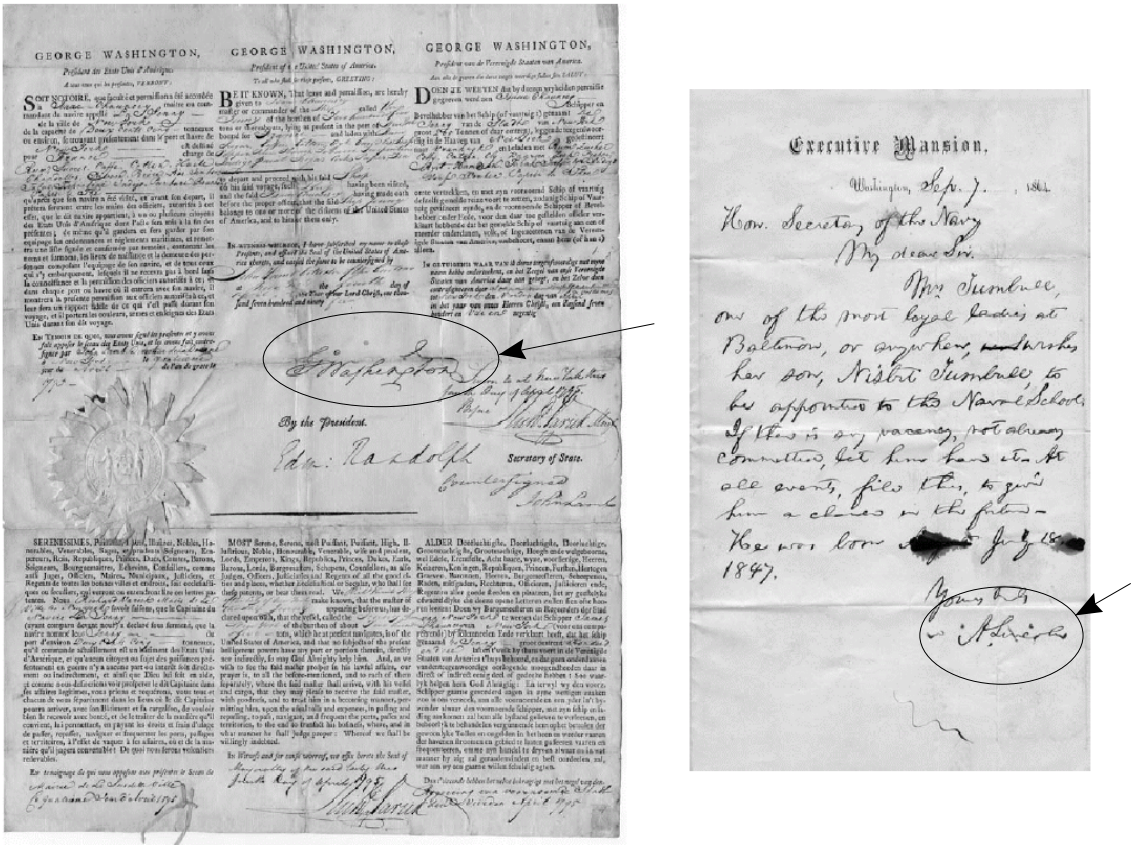
1.2.2 Handwritten signatures

Handwritten signatures are socially and legally well accepted as a convenient means of document authentication, authorisation and writer identification. Since most documents, for example bank cheques, need to be signed, automated off-line signature verification forms an essential component in the authentication of documents with embedded signatures. Figure 1.3 shows a document signed by George Washington (1795) and a letter written and signed by Abraham Lincoln (1864).

1.2.3 Automatic identification systems

Since the introduction of computers, modern society has become increasingly dependent on the electronic storage and transmission of information. In many transactions the electronic verification of a person's identity proved beneficial and this inspired the development of a wide range of automatic identification systems.

Plamondon and Shihari (2000) note that automated signature verification systems occupy a very specific niche among other automated identification systems: "One the one hand, they differ from systems based on the possession of something (key, card etc.) or the knowledge of something (passwords, personal information etc.), because they rely on a specific, well learned gesture. On the other hand, they also differ from systems based on the biometric properties of the individual (finger prints, voice prints, retinal prints, etc.),



(a) (b)

Figure 1.3: *(a) The famous "Three Language Ships"' paper signed by George Washington as president of the USA, dated April 4, 1795. This paper was written in English, French and Dutch. (b) A letter written and signed by Abraham Lincoln as president of the USA, dated September 7, 1864.*

because the signature is still the most socially and legally accepted means of personal identification."

Although handwritten signatures are by no means the most reliable means of personal identification, signature verification systems are inexpensive and non-intrusive. Handwritten signatures provide a direct link between the writer's identity and the transaction, and are therefore perfect for endorsing transactions.

A categorisation of automatic identification systems is given in figure 1.4.

1.2.4 Recognition and verification

A clear distinction should be made between verification systems and recognition systems. A *verification* system merely decides whether a particular entity belongs to a specific

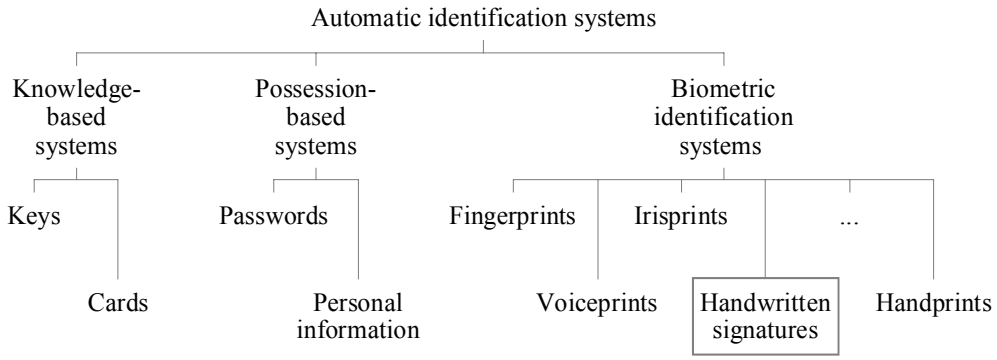


Figure 1.4: *Automatic identification systems.*

class, or not. A *recognition* system, on the other hand, has to decide which of a certain number of classes the entity belongs to.

This dissertation focuses on off-line signature verification. Throughout the dissertation we often refer to "classifiers" and "classes". The reader should note that although classifiers and verifiers are different, one may interpret a verifier as a classifier with two classes. In the context of signature verification, for example, we have a class of genuine signatures (positive class) and a class of forgeries (negative class). Samples of forged signatures are usually unobtainable. Signature recognition systems differ from signature verification systems in the sense that signature recognition systems utilise sample signatures for each class (writer).

1.2.5 Statistical pattern recognition

In the statistical approach to pattern recognition, each entity (pattern) is usually represented by a sequence of T, d-dimensional feature vectors, that is an observation sequence. Each entity can also be represented by a single feature vector, that is $T = 1$. Each component (feature) of a feature vector represents a measurement. A pattern can therefore be viewed as a point in a $d \times T$-dimensional feature space. The goal is to choose those features that allow observation sequences belonging to different classes to occupy compact and disjoint regions in the feature space. The effectiveness of the representation space (feature set) is determined by how well observation sequences from different classes can be separated. Given a set of training sequences from each class, the objective is to establish decision boundaries in the feature space which separate sequences belonging to different classes.

Statistical pattern recognition systems include minimum distance classifiers, neural networks (NNs) and Bayesian classifiers. In the case of a minimum distance classifier, each class is typically represented by a Gaussian probability density function (PDF). The observation sequence is assigned to the class for which the height (likelihood) of the PDF is the greatest or alternatively, the class for which the distance (scaled negative loglikeli-

hood) to the mean vector is the smallest. Two types of minimum distance classifiers are particularly relevant to this work, that is simple distance classifiers (SDCs), for which $T = 1$, and HMMs, for which $T > 1$.

Simple distance classifiers. In the case of a SDC, each entity (pattern) is represented by a *single* observation, that is a single d-dimensional feature vector. Each pattern class is typically represented by a Gaussian PDF in a d-dimensional feature space, where each PDF is uniquely defined by the mean vector and covariance matrix of the feature vectors that belong to the particular class.

Data scarcity. The mean vector for each class can always be estimated, even when there is only one sample available. However, a full covariance matrix can only be reliably estimated when the number of available samples is much greater than the dimension of the feature vectors. When this is not the case, one may for example assume that the off-diagonal elements of each covariance matrix are equal to zero, in which case only the diagonal entries are estimated. The features are therefore assumed to be uncorrelated. This implies that the data distribution is modelled less accurately, in the sense that no provision is made for the possible rotation of the data. In the extreme case, that is when the number of samples available is much less than the dimension of the feature vectors, the covariance matrices may for example not be estimated at all. This implies that each covariance matrix is kept equal to the identity matrix during training and that each class is modelled by only the mean vector for that class. The distribution of the data is therefore ignored.

Distance measures. When the full covariance matrix is estimated for each class, classification is based on the *Mahalanobis distance*. When only the mean vector is estimated for each class, classification is based on the *Euclidean distance*. The Euclidean and Mahalanobis distance measures are defined in sections 3.2 and 3.3, respectively.

The theory of SDCs is discussed in more detail in chapter 3.

Dynamic time warping. DTW includes a vast range of dynamic programming algorithms that are used to non-linearly align feature vectors (or observation sequences). The *aligned* feature vectors (or observation sequences) are then compared (matched). DTW can also be used to match an observation sequence with an HMM (see chapter 6). We discuss some theoretical aspects of dynamic programming in appendix A and we present the DTW-based off-line signature verification system developed in this dissertation, in chapter 5.

Hidden Markov models. HMMs are used to model a *sequence* of observations and their relationship to each other. They are extensively used in speech recognition systems (where each observation describes a segment of speech) and in on-line signature verification systems (where each observation describes the dynamics of a signature within a certain time frame). HMMs are ideally suited for these applications, since the observations are time dependent. Since off-line signatures contain no temporal information, the extraction of observations (that are time dependent) is less trivial.

An HMM consists of N *states*, where each state has two principal elements: a PDF which describes the nature of a group of observations that is associated with the state, and a histogram which describes the probability of making a transition to any of the

N states − this includes the probability of staying in the same state. We discuss some theoretical aspects of HMMs in appendix B and we present the HMM-based off-line signature verification system developed in this dissertation, in chapter 6. For a good introduction to HMMs, see Rabiner (1989).

In applications that require *classification*, an HMM is constructed for each pattern class. Each of these HMMs are appropriately initialised and then trained with a set of training patterns. When a test pattern (observation sequence) is to be classified, the assumption is made that the observation sequence was produced by one of these HMMs. In order to decide which one, the likelihood of each of them producing this observation sequence is evaluated.

In automatic identification systems that require *verification*, an HMM is trained for each client at enrollment. The likelihood of the client producing the pattern is evaluated by matching the observation sequence with the relevant HMM. The pattern is accepted when this likelihood is greater than a certain threshold, otherwise it is rejected. This threshold can for example be estimated by matching all the training sequences with the HMM in question, before considering the statistics of these likelihoods.

1.2.6 Quality performance measures

Throughout this dissertation, the *false rejection rate (FRR)*, the *false acceptance rate (FAR)*, the *equal error rate (EER)* and the *average error rate (AER)* are used as quality performance measures. The FRR is the ratio of the number of genuine test signatures rejected and the total number of genuine test signatures submitted. The FRR is also called the type I error and is defined as follows,

$$\text{FRR} = \frac{\text{Total number of genuine test patterns rejected}}{\text{Total number of genuine test patterns submitted}}. \tag{1.1}$$

The FAR is the ratio of the number of forgeries accepted and the total number of forgeries submitted. The FAR is also called the type II error and is defined as follows,

$$\text{FAR} = \frac{\text{Total number of forgeries accepted}}{\text{Total number of forgeries submitted}}. \tag{1.2}$$

When the decision threshold is altered so as to decrease the FRR, the FAR will invariably increase, and vice versa. This implies that, for a certain threshold, the FRR will be equal to the FAR. This error rate is called the EER and the corresponding threshold value is sometimes referred to as the equal error threshold. The average of the FRR and FAR is called the AER. When a threshold is used, that is close to the equal error threshold, the FRR and FAR will not differ much. In this case the AER is approximately equal to the EER.

The algorithms developed in this dissertation assign a *distance* measure to every test signature, where a distance of zero is attributed to a perfect match (genuine signature or positive class) and a distance of infinity to a complete mismatch (forgery or negative class). However, a typical threshold-based verifier assigns a *confidence* to every test pattern. A confidence is calculated in such a way that it is inversely proportional to the

distance measure. A confidence of one is attributed to a perfect match (genuine signature or positive class) and a confidence of zero is attributed to a complete mismatch (forgery or negative class).

Figure 1.5 shows ideal confidence distributions for forgeries (dotted graph) and genuine test patterns (solid graph). Usually the dotted graph is unknown, since no forgeries are available for training – more importantly, forgeries do not fall in a well-identifiable class. Area "A", that is the area bounded by the confidence distribution for the genuine test patterns, the threshold and the confidence-axis, represents the probability that a genuine test pattern is falsely rejected and is equal to the FAR (figure 1.6(a)). Similarly, area "B", that is the area bounded by the confidence distribution for the forgeries, the threshold and the confidence-axis, represents the probability that a forgery is falsely accepted and is equal to the FRR (figure 1.6(a)).

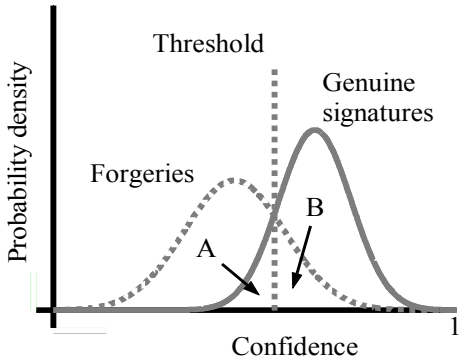


Figure 1.5*: Ideal confidence distributions. In practice, forgeries do not fall in a well-identifiable class, and is typically not represented by a Gaussian distribution as suggested here.*

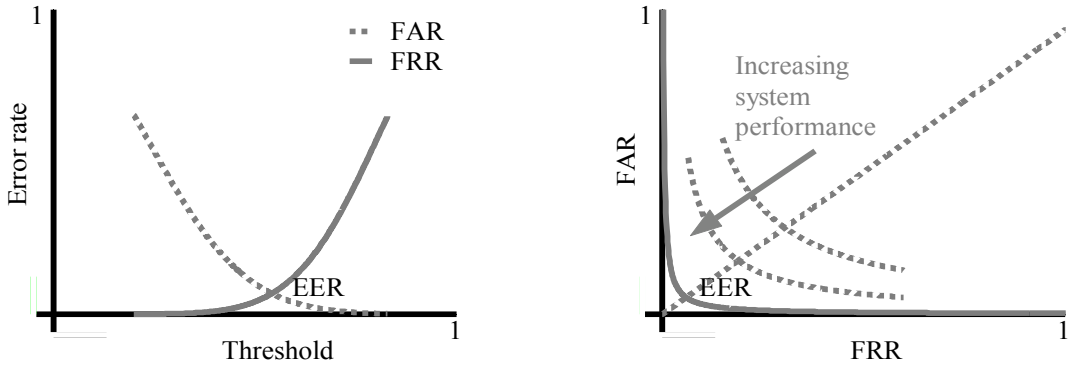


Figure 1.6*: Quality performance measures. (a) FAR, FRR and EER. (b) ROC graph.*

Another widely used quality performance measure is the area bounded by a so-called *ROC (receiver operating characteristic) graph* and its two axes. An ROC graph is obtained by

plotting the FAR against the FRR. Figure 1.6(b) shows a theoretical ROC graph. From this graph it is clear that it is not possible to minimise the FAR and the FRR at the same time, however, a compromise can be reached where the FAR equals the FRR, that is the EER. The closer the ROC graph is to the origin, the better the quality of the system.

1.2.7 Off-line and on-line signature verification

Diverse applications inspired researchers to investigate the feasibility of two distinct categories of automatic signature verification systems – those concerned with the verification of static signature images and those concerned with the verification of signatures that were captured dynamically, using a special pen and digitising tablet. These systems are referred to as off-line and on-line systems respectively.

In *off-line* systems, a signature is digitised using a hand-held or flat-bed scanner and only the completed writing is stored as an image. These images are referred to as *static* signatures. Off-line systems are of interest in scenarios where only hard copies of signatures are available, for example where a large number of documents need to be authenticated.

In the *on-line* case a special pen is used on an electronic surface such as a digitiser combined with a liquid crystal display. Apart from the two-dimensional coordinates of successive points of the writing, pen pressure, as well as the angle and direction of the pen, are captured dynamically and then stored as a function of time. The stored data is referred to as a *dynamic* signature and also contains information on pen velocity and acceleration. On-line systems are of interest for "point-of-sale" and security applications.

Since on-line signatures also contain dynamic information, they are difficult to forge. Off-line systems also have to account for background noise and variations in stroke-width. It therefore comes as no surprise that off-line signature verification systems are much less reliable than on-line systems.

1.2.8 Forgeries

A signature verification system typically focuses on the detection of one or more category of forged signatures. A *skilled* forgery is produced when the forger has unrestricted access to one or more samples of the writer's actual signature (see figure 1.7 (b)). A *casual* forgery or *simple* forgery (see figure 1.7 (c)) is produced when the forger is familiar with the writer's name, but does not have access to a sample of the actual signature – stylistic differences are therefore prevalent. A *random* forgery or *zero-effort* forgery (see figure 1.7 (d)) can be any random scribble or a signature of another writer, and may even include the forger's own signature. The genuine signatures and high quality forgeries for other writers are usually considered to be forgeries of this type. It is not compulsory in South Africa for a person's signature to be readable. Casual forgeries are therefore not necessarily of a better quality than random forgeries. In figure 1.1 we therefore see real-life examples of a skilled and random forgery.

Skilled forgeries can be subdivided into *amateur* and *professional* forgeries. A *profes-*

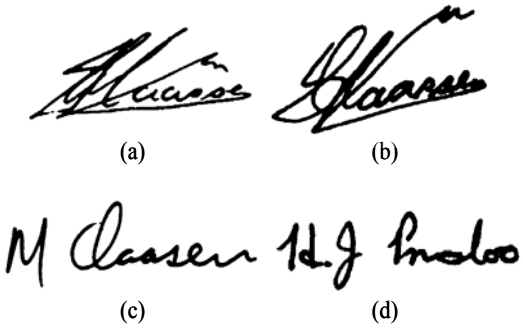


Figure 1.7*: Example of a (a) genuine signature, (b) skilled forgery, (c) casual forgery and (d) random forgery for the writer "M. Claasen".*

sional forgery is produced by an individual who has professional expertise in handwriting analysis. They are able to circumvent obvious problems and exploit their knowledge to produce high quality, spacial forgeries (see figure 1.8 (b)).

In the context of on-line verification, amateur forgeries can be subdivided into "home-improved" and "over-the-shoulder" forgeries (see Dolfing et al. (1998a)). The category of *home-improved forgeries* contains forgeries that are produced when the forger has a paper copy of a genuine signature and has ample opportunity to practice the signature at home. Here the imitation is based only on the static image of the original signature (see figure 1.8 (c)). The category of *over-the-shoulder* forgeries contains forgeries that are produced immediately after the forger has witnessed a genuine signature being produced. The forger therefore learns not only the spatial image, but also the dynamic properties of the signature by observing the signing process (see figure 1.8 (d)). The different types of forgeries are summarised in figure 1.9. The systems developed in this dissertation target the boxed forgeries. Note that these systems do not *specialise* in the detection of random forgeries. We give the reasons for this in section 7.2. We do however evaluate the ability of these systems to detect random forgeries. The results for this experiment are discussed in section 8.7.

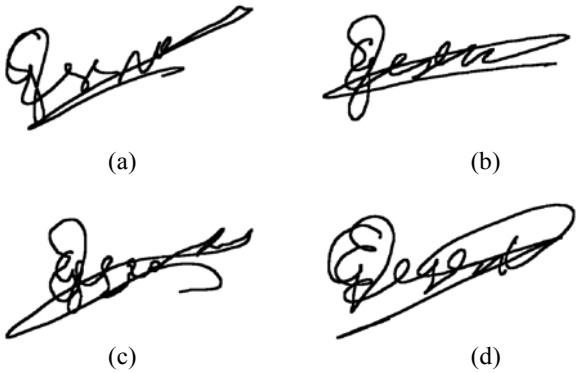


Figure 1.8*: Example of a (a) genuine signature, (b) professional forgery, (c) home-improved forgery and (d) over-the-shoulder forgery.*

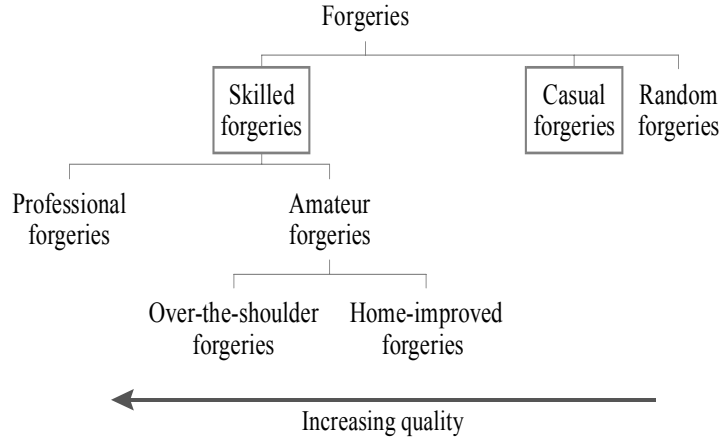


Figure 1.9: *Types of forgeries.*

1.3 Literature synopsis

A great deal of work has been done in the area of off-line signature verification over the past two decades.

Surveys. A recent article by Guo, Doermann, and Rosenfeld (2001) includes an extensive overview of previous work. Numerous methods and approaches are summarised in a number of survey articles. The state of the art from 1993 to 2000 is discussed in a paper by Plamondon and Shihari (2000). The period from 1989 to 1993 is covered by Leclerc and Plamondon (1994) and the period before 1989 by Plamondon and Lorette (1989). Another survey was published by Sabourin, Plamondon, and Lorette (1992b). A review of on-line signature verification by Gupta and McCabe (1998) also includes a summary of some earlier work on the off-line case.

Global and local features. Earlier work on off-line signature verification deals primarily with casual and random forgeries. Many researchers therefore found it sufficient to consider only the *global features* of a signature. Global features describe an entire signature image and include the Hough transform (Kaewkongka et al. (1999)), horizontal and vertical projections (Fang et al. (2003)) and smoothness features (Fang et al. (2001)).

As signature databases became larger and researchers moved toward more difficult skilled forgery detection tasks, we saw a progression not only to more elaborate classifiers, but also to the increased use of *local features* and matching techniques. Local features are extracted at stroke and sub-stroke level and include unballistic motion and tremor information in stroke segments (Guo et al. (2001)), stroke "elements" (Fang et al. (2003)), local shape descriptors (Sabourin et al. (1997c)), and pressure and slant features (Quek et al. (2002)).

Pattern recognition techniques. Various pattern recognition *techniques* have been exploited to authenticate handwritten signatures. These techniques include template matching techniques (Deng et al. (1999); Fang et al. (2003); Guo et al. (2001)), SDCs (Fang et al. (2001, 2002); Mizukami et al. (2002); Sabourin et al. (1997c)), NNs (Baltzakis et al. (2001); Kaewkongka et al. (1999); Quek et al. (2002)), HMMs (El-Yacoubi et al. (2000); Justino et al. (2001)) and structural pattern recognition techniques.

A detailed discussion of existing off-line signature verification systems is presented in chapter 2.

1.4 Objectives of this study

The purpose of this research is to develop an off-line signature verification system that is

- *novel* (in the sense that the approach we use is fundamentally different from the approaches used in existing systems, so as to make the approach used in this dissertation complimentary to other approaches),
- *accurate* and *efficient* (so as to make the adoption of the system developed in this dissertation, independent from other systems, economically viable).

1.5 Overview of this work

In this dissertation we focus on off-line signature verification. We are therefore not concerned with the verification of dynamic signatures, nor with the recognition of signatures. We developed *two* systems, a *DTW-based system* and an *HMM-based system*. A précis of the HMM-based system can be found in a paper by Coetzer, Herbst, and Du Preez (2004). Throughout this dissertation we often refer to "the DTW-based system" and "the HMM-based system", instead of "the DTW-based system developed in this dissertation" and "the HMM-based system developed in this dissertation", respectively.

1.5.1 System design

Feature extraction. The systems developed in this dissertation use similar feature extraction techniques. The bulk of the image processing and feature extraction involves the calculation of the *discrete Radon transform (DRT)* of each signature image. The DRT is obtained by calculating projections of each signature at different angles. The DRT is very similar to the *Hough transform* (Kaewkongka et al. (1999)). We show in chapter 4 that the DRT is a very stable feature extraction method. After some further image processing (normalisation), each of these projections constitutes a feature vector in an observation sequence. These features are classified as global features, since they are not extracted at stroke or sub-stroke level.

Signature modelling. The systems developed in this dissertation use two very different approaches to model a specific writer's signature. In the case of the DTW-based system, each writer's signature is modelled by an observation sequence, that represents the writer's most representative training signature. This observation sequence acts as a template for the writer's signature. In the case of the HMM-based system, each writer's signature is modelled by an HMM of which the states are organised in a ring.

Matching. The distance between a test signature and a model for the claimed writer's signature is obtained as follows. The DTW-based system developed in this dissertation

matches the observation sequence for a test signature with the observation sequence for the reference (template) signature of the claimed writer, by first aligning these observation sequences in an optimal way. This alignment is necessary to achieve rotation invariance and is discussed in more detail in chapter 5. The average of the DTW-based distances between the *aligned* observations is then calculated. The HMM-based system developed in this dissertation matches the feature set (observation sequence) for a test signature with an HMM of the claimed writer's signature, through Viterbi alignment. A distance measure is obtained by calculating a negative loglikelihood.

Verification. When a claim is made that a test signature belongs to a specific writer, the extracted observation sequence is first matched with a model of the writer's signature, so that a distance measure is obtained. This distance measure is then normalised in order to compensate for the variation in the writer's signature. The variation in the writer's signature is estimated by matching all of the writer's training signatures with the writer's signature model. In this way several distance measures are obtained. Statistics of these distance measures are then used to estimate the variation in the writer's training set. A *global* threshold, that is a threshold which is the same for all writers, can therefore be used. Test signatures, for which the normalised distance measure is less than this threshold are accepted − the others are rejected. The verifiers are constructed in such a way that they are geared towards the detection of only skilled and casual forgeries (see section 1.2.8). We therefore do not consider random forgeries. We give the reasons for this in section 8.7. A schematic representation of the systems developed in this dissertation is given in figure 1.10.

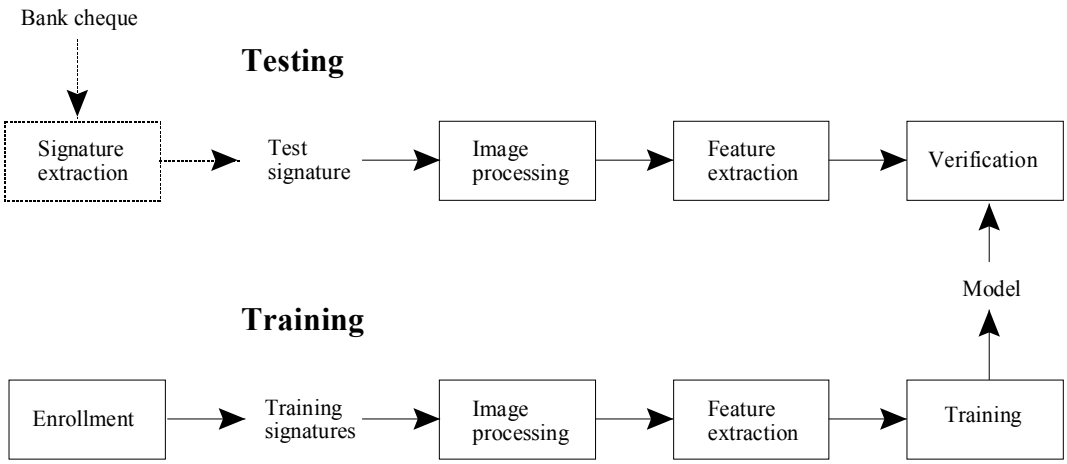


Figure 1.10: *A schematic representation of the systems developed in this dissertation.*

1.5.2 Data

Experiments are conducted on two different data sets. We call these data sets the Stellenbosch data set and Dolfing's data set respectively.

The Stellenbosch data set. We first test the systems developed in this dissertation on an independent database of 924 off-line signatures that we collected from 22 writers. We consider thirty genuine signatures, six skilled forgeries and six casual forgeries for each writer. For each writer, ten genuine signatures are used for training and twenty for testing. No genuine signatures are used for validation purposes.

Dolfing's data set. Since it makes sense to compare the results for the systems developed in this dissertation to those of another algorithm on the *same* database of signatures, and since off-line signature databases are not freely available, we also test the HMM-based system on a set of signatures that was originally captured on-line. Hans Dolfing was kind enough to make this database available to us. Philips Research enabled the data collection. Dolfing's data set contains 4800 signatures from 51 writers. Before we test the HMM-based system on Dolfing's signatures, we first use the pen-tip coordinates, which is part of the dynamic signature data, to create static signature images (see section 8.2.2). We then compare these results to the results of one of Dolfing's on-line algorithms. This algorithm uses an HMM and only considers the spacial coordinates of each writing. We consider thirty genuine signatures for each writer, an average of 58.8 amateur forgeries per writer, and an average of 5.3 professional forgeries per writer. For each writer, fifteen genuine signatures are used for training and fifteen for testing. No genuine signatures are used for validation purposes.

The above data sets are discussed in more detail in section 8.2.

1.5.3 Results

DTW-based system.

The Stellenbosch data set. When only skilled forgeries are considered, an EER of approximately 18% is achieved. When only casual forgeries are considered, the DTW-based algorithm achieves an EER of approximately 4% (see figure 8.2).

HMM-based system.

The Stellenbosch data set. When only skilled forgeries are considered, an EER of approximately 18% is achieved. When only casual forgeries are considered, the HMM-based algorithm achieves an EER of 4.5% (see figure 8.3).

Dolfing's data set. When only amateur forgeries are considered, the HMM-based algorithm achieves an EER of 12.2% (see figure 8.4). Dolfing's on-line algorithm (see Dolfing (1998b), p. 160) achieves an EER of 13.3%.

Discussion. When tested on the Stellenbosch data set, the DTW-based and HMM-based systems achieve similar results. This is quite surprising, since the HMM-based system uses a more sophisticated model to represent a writer's signature. However, the HMM-based system is computationally more *efficient* than the DTW-based system, which makes the HMM-based system more suited for commercial implementation (see section 9.4). When tested on Dolfing's data set, the HMM-based system does a little better than Dolfing's on-line algorithm. This is despite the fact that Dolfing's algorithm also considers the *sequence* in which the spatial coordinates were produced. We also

demonstrate in section 8.6 that the HMM-based system performs better than a typical human being.

1.6 Contribution of this work

- **A novel HMM-based system.**

  The results for a wide range of off-line signature verification systems, which focus on casual and/or skilled forgery detection, are well documented (see chapter 2). Although neither of the systems developed in this dissertation outperforms all of these existing systems, the HMM-based system utilises a unique combination of a feature extraction algorithm and a pattern recognition technique.

  The HMM-based system employs a feature extraction algorithm, that is based on the calculation of the DRT (see section 4.2), and then represents these features with a ring-structured HMM (see section 6.4.1). Although a few other systems either use similar feature extraction algorithms (Baltzakis et al. (2001), Deng et al. (1999), El-Yacoubi et al. (2000), Fang et al. (2001), Fang et al. (2002), Mizukami et al. (2002) and Sabourin et al. (1997c)), or use similar pattern recognition techniques (Baltzakis et al. (2001), Deng et al. (1999), Fang et al. (2001), Fang et al. (2002), Guo et al. (2001), Mizukami et al. (2002), Quek et al. (2002) and Sabourin et al. (1997c)), *none of these systems uses a DRT/HMM combination.* Since all of these systems differ fundamentally from the HMM-based system, it is very likely that a combination of any of these systems and the HMM-based system will result in a superior merged system. This will make their approach complementary to the HMM-based approach used in this dissertation.

- **A robust DRT representation, enabling rotation, shift and scale invariance.**

  The feature extraction techniques for both the HMM-based and DTW-based systems developed in this dissertation are based on the calculation of the DRT of a signature image. The DRT is a matrix, where each column represents a projection or shadow of the original image at a certain angle. The algorithm that is used to calculate the DRT is discussed in section 4.2.

  When a sufficient number of projections are calculated at equally distributed angles between 0^{o} and 180^{o}, the original signature image can be reconstructed from the projection data. The projections therefore contain the same information as the original signature image. However, *we also consider the projections calculated at equally distributed angles between* 180^{o} *and* 360^{o}. Since these projections are simply reflections of the projections already calculated, no additional calculations are necessary.

  We now briefly discuss the advantages of basing the feature extraction algorithm used in this dissertation on the calculation of an (augmented) DRT. These advantages are discussed in more detail in section 4.4.

Simulated time-evolution. Each signature is a static image and contains no dynamic information. Since the feature vectors are obtained by calculating projections at different angles, simulated time-evolution is created from one feature vector to the next, where the angle is the dynamic variable. This enables the HMM-based system to construct an HMM for each signature.

Rotation invariance. Since the augmented DRT is periodic, with a period of 360^{o}, we are able to achieve rotation invariance in an elegant, yet robust way. The HMM-based system, for example, represents each writer's signature with an HMM of which the states are organised in a ring. The HMM is constructed in such a way that it is equally likely to enter the HMM at any state. This guarantees rotation invariance (see section 6.4.1). For the DTW-based system, rotation invariance is only achieved after some further processing (see section 4.3.2).

Shift and scale invariance. The projection data is processed so that each processed projection represents an observation in an observation sequence. The projections are normalised in such a way that each observation sequence is a shift and scale invariant representation of the corresponding signature image. The calculation of the DRT again enables us to achieve shift and scale invariance in an elegant, yet robust way. The zero-valued components of each projection are decimated (removed), so that the corresponding feature vector is constructed from the remaining components only. Each decimated projection is then expanded or shrunk to the required dimension through linear interpolation. The intensity of each feature vector is normalised by dividing it by the variance of the intensity of the entire set of feature vectors.

- **Rotation invariance.**

  The HMM-based system represents each writer's signature with an HMM of which the *states are organised in a ring.* This structure is similar to that of a traditional left-to-right model, but a transition from the last state to the first state is allowed. Since each HMM is constructed in such a way that it is equally likely to enter the model at any state, the periodic nature of the augmented DRT (which contains projections calculated at equally distributed angles between 0^{o} to 360^{o}), guarantees that each observation sequence is a rotation invariant representation of the signature in question. We elaborate on this in sections 4.4 and 6.4.1. Rotation invariance is therefore achieved in a robust and efficient way.

1.7 Outline of the dissertation

Chapter 2: *Literature study* discusses some recent work on off-line signature verification and gives some historical perspectives.
Chapter 3: *Simple distance classifiers* discusses the theory of SDCs and explains how to deal with the problem of data scarcity.
Chapter 4: *Image processing and feature extraction* presents the DRT as a stable feature extraction method and discusses some of the other image processing algorithms that are used during the feature extraction phase.

Chapter 5: *DTW-based signature modelling* explains how the DTW-based system developed in this dissertation constructs a model for each writer's signature.
Chapter 6: *Signature modelling using HMMs* discusses the HMM-based off-line signature verification system developed in this dissertation.
Chapter 7: *Verification* discusses the verification protocol and threshold selection strategy for both of the systems developed in this dissertation.
Chapter 8: *Experiments* discusses the data sets used in this dissertation and the experimental setup. The results for both of the systems developed in this dissertation are presented. We also compare the performance of these systems to the performance of existing systems, and to the performance of human beings.
Chapter 9: *Computational requirements* discusses the computational requirements for the systems developed in this dissertation.
Chapter 10: *Outstanding issues and conclusion* touches on some outstanding issues and discusses the prospects for continuing this research in the future.

Chapter 2

Literature Study

2.1 Introduction

In this chapter we discuss some existing off-line signature verification systems. We categorise these systems according to the pattern recognition technique used (see figure 1.2). We discuss template matching techniques (section 2.2), SDCs (section 2.3), Bayesian classifiers (section 2.4), NNs (section 2.5), structural techniques (section 2.6), support vector machines (SVMs) (section 2.7) and HMMs (section 2.8). Hybrid systems, that is systems that use more than one pattern recognition technique, are classified according to the principal technique. Although recent systems are generally more relevant to this work, we also discuss a few older systems in order to provide a historical perspective. In section 8.5 we compare these systems to ours.

Although the term "classifier" (e.g. SDC) is frequently used in this chapter, we remind the reader that, in the context of off-line signature verification, a verifier (i.e. a classifier with only two classes) is often implied (see section 1.2.4).

When discussing a system we follow a few general guidelines. For each system we discuss

- the feature extraction method used, specifically whether local or global features are used,
- the strategy used to obtain a model for each writer's signature,
- the technique used to match a test signature with a model,
- the verification strategy used,
- the database(s) used to evaluate the system, specifically the type of forgeries that are targeted and finally,
- the experimental results.

2.2 Template matching techniques

Template matching is one of the earliest and simplest approaches to pattern recognition. A pattern class is represented by a template or prototype pattern. Such a prototype pattern can either be a curve or an image.

DTW is the most popular template matching technique for off-line signature verification. Although this is one of the older and simpler approaches to pattern recognition, many recent systems still use this technology. We discuss DTW in more detail in chapter 5 and in appendix A.

Wilkinson and Goodman (1990) use DTW to detect casual forgeries. Based on the assumption that the properties of curvature, total length and slant angle are constant among different sample signatures, each signature is represented by a slope histogram. They use a database of 500 genuine signatures and 306 casual forgeries from nine individuals, sampled over a period of 18 months. The authors report an EER of approximately 7%.

Shapiro and Bakalov (1993) use DTW to compare projections of signature images at different angles, on the basis that the signature image can be retrieved from these projections.

Nouboud and Plamondon (1994) use DTW for curve comparison, where the curve is obtained from the envelope of the signature.

M. Yoshimura and I. Yoshimura (1997) apply a DTW technique to the projection profile of the frequency of black pixels on the x-axis for Japanese signature verification. The average EER is approximately 12.9%.

Deng et al. (1999) developed a system that uses a closed contour tracing algorithm to represent the edges of each signature with several closed contours. The curvature data of the traced closed contours are decomposed into multiresolutional signals using wavelet transforms. The zero-crossings corresponding to the curvature data are extracted as features for matching. A statistical measurement is devised to decide systematically which closed contours and their associated frequency data are most stable and discriminating. Based on these data, the optimal threshold value which controls the accuracy of the feature extraction process is calculated. Matching is done through DTW. Experiments are conducted independently on two data sets. The first data set consists of only English signatures, while the second data set consists of only Chinese signatures. For each experiment twenty-five writers are used with ten training signatures, ten genuine test signatures, ten skilled forgeries and ten casual forgeries per writer. For the English data set an FRR of 5.6% and FARs of 21.2% (skilled forgeries) and 0% (casual forgeries) is reported. For the Chinese data set an FRR of 6.0% and FARs of 13.5% (skilled forgeries) and 0% (casual forgeries) is reported.

Guo, Doermann, and Rosenfeld (2001) approach the off-line problem by establishing a local correspondence between a model and a submitted signature. The submitted signature is segmented into consecutive stroke segments that are matched to the stroke segments of the model. The cost of the match is determined by comparing a set of geometric properties of the corresponding sub-strokes and computing a weighted sum of the property value differences. The least invariant features of the least invariant sub-strokes are given

the largest weights, thus emphasising features that are highly writer dependent. Using the local correspondence between the model and a submitted signature, the writer dependant information embedded at the sub-stroke level is examined and unballistic motion and tremor information in each stroke segment are examined. Matching is done through DTW. A database with ten writers is used with five training signatures, five genuine test signatures, twenty skilled forgeries and ten casual forgeries per writer. The skilled forgeries consist of simulated (50%) and traced (50%) forgeries. The same genuine signatures that are used for training are also used for testing. An FRR of 6% and an FAR of 11.5% is obtained when only skilled forgeries are considered, while an FRR of 2% and an FAR of 3.3% is obtained when only casual forgeries are considered.

Fang et al. (2003) propose two methods for the detection of skilled forgeries. These methods are evaluated on a database of 1320 genuine signatures from 55 writers and 1320 forgeries from 12 forgers. In determining the FRR, the leave-one-out method was adopted to maximise the use of the available genuine signatures. The signatures are therefore authenticated in an iterative way. During each iteration one genuine signature is selected for testing (left out of the training set), while the other genuine signatures are used for training.

Fang's *first method* calculates one dimensional projection profiles for each signature in both the horizontal and vertical directions. These profiles are then optimally matched with reference profiles using DTW. This method differs from previous methods in the sense that the distance between the warped projection profiles is not used in the decision. Instead, the positional distortion of each point of the sample profile, when warped onto a reference profile, is incorporated into a distance measure. A Mahalanobis distance is used instead of a simple Euclidean distance. The leave-one-out covariance (LOOC) method is adopted for this purpose, but the unreliable off-diagonal elements of the covariance matrices are set to zero. When binary and gray-scale signatures are considered, the best AERs for this method are 20.8% and 18.1% respectively.

Fang's *second method* matches the individual stroke segments of a two-dimensional test signature directly with those of a template signature, using a two-dimensional elastic matching algorithm. The objective of this algorithm is to achieve maximum similarity between the "elements" of a test signature and the "elements" of a reference signature, while minimising the deformation of these signatures. A gradient descent procedure is used for this purpose. Elements are short straight lines that approximate the skeleton of a signature. A Mahalanobis distance, with the same restrictions as for the first method, is used. An AER of 23.4% is achieved for this method.

2.3 Simple distance classifiers

An SDC usually represents each pattern class with a Gaussian PDF, where each PDF is uniquely defined by the mean vector and covariance matrix of the feature vectors that belong to the particular class. When the full covariance matrix is estimated for each class, classification is based on the Mahalanobis distance. When only the mean vector is estimated for each class, classification is based on the Euclidean distance. SDCs are

discussed in more detail in chapter 3.

We briefly discuss three off-line signature verification systems that were developed by research teams lead by Robert Sabourin from 1993 to 1997. These systems primarily use SDCs and target random forgeries. A database of 800 genuine signatures from 20 writers is used for each of these systems.

The *first system* (Sabourin, Cheriet, and Genest (1993)) uses an extended-shadow-code representation. Two experiments, that use a k nearest neighbours classifier (with vote) and a minimum distance classifier respectively, are conducted on the above-mentioned database. An AER of 0.01% is obtained for the first experiment when $k = 1$. When 10 training signatures are used for each writer, an AER of 0.77% is obtained for the second experiment.

The *second system* (Sabourin, Drouhard, and Wah (1997b)) uses a shape matrix representation and a minimum distance classifier. A best AER of 0.84% is reported.

The *third system* (Sabourin, Genest, and Prêteux (1997c)) uses granulometric size distributions for the definition of local shape descriptors in an attempt to characterise the amount of signal activity exciting each retina on the focus of an superimposed grid. The system then uses a nearest neighbour and threshold based classifier. AERs of 0.02% and 1.0% are reported for the respective classifiers.

Fang et al. (2001) developed a system that is based on the assumption that the cursive segments of forged signatures are generally less smooth than those of genuine ones. Two approaches are proposed to extract the smoothness feature: a crossing method and a fractal dimension method. The smoothness feature is then combined with global shape features. Verification is based on a SDC. An iterative leave-one-out method is used for training and for testing genuine test signatures. A database with 55 writers is used with 24 training signatures and 24 skilled forgeries per writer. An FRR of 18.1% and an FAR of 16.4% is obtained.

Fang et al. (2002) also developed a system that uses an elastic matching method to generate additional samples. A set of peripheral features, which is useful in describing both the internal and the external structures of signatures, is employed to represent a signature in the verification process. Verification is based on a Mahalanobis distance classifier. An iterative leave-one-out method is used for training and for testing genuine test signatures. The database used in Fang's previous paper, is again used here. The additional samples generated by this method reduce the AER from 15.6% to 11.4%.

Mizukami et al. (2002) propose a system based on a displacement extraction method. The optimum displacement functions are extracted for any pair of signatures using minimisation of a functional. The functional is defined as the sum of the squared Euclidean distance between two signatures and a penalty term that requires smoothness of the displacement function. A coarse-to-fine search method is applied to prevent the calculation from stopping at local minima. Based on the obtained displacement function, the dissimilarity between the submitted signature and the corresponding authentic one is measured. A database with twenty writers is used with ten training signatures, ten genuine test signatures and ten skilled forgeries per writer. An AER of 24.9% is obtained.

2.4 Bayesian classifiers

For a tutorial on Bayesian networks, the reader is referred to an article by Heckerman (1999).

Xiao and Leedham (2002) investigate the feasibility of using a modified Bayesian network for off-line signature verification. For each signature, a top and bottom profile are calculated. It is assumed that the signature strokes are represented by black pixels. The top profile is calculated as follows: a minimum rectangular bounding box, which embraces the signature image, is used to define the signature area. This box is then scanned vertically line by line, from top to bottom and from left to right until a black pixel is encountered. The scan line (up to the encountered black pixel) is then filled with black pixels. The bottom profile is obtained in a similar way. The length from the beginning of the scan line to the encountered black pixel is called the run length of the profile at that point. These profiles are then subdivided into smaller components at the positions where the run lengths of two adjacent points change significantly. For each component certain attributes are extracted. These attributes include the length of the component, the run length of each scan line in the component, etc. They use a small database that consists of the signatures of eight writers. Between 10 and 20 signatures are collected for each writer and 60% of these signatures are used for training. An FRR of 20% and an FAR of 14% is reported. Although the test set contains some skilled forgeries, this paper is not clear on the quality of the other forgeries.

2.5 Neural networks

An NN is a massively parallel computing system that consists of a large number of simple processors with many interconnections. The main characteristic of an NN is that it has the ability to learn complex non-linear input-output relationships, use sequential training procedures, and adapt itself to the data. An NN model attempts to use organisational principles in a network of weighted directed graphs, in which the nodes are artificial neurons (perceptrons) and the directed edges (with weights) are connections between neuron outputs and neuron inputs.

NNs have been extensively used in off-line signature verification over the last two decades. Most of these studies use conventional approaches, like multilayer perceptrons (Mighell et al. (1989); Barua (1992); Sabourin et al. (1992a); Bajaj et al. (1997); Dehghan et al. (1997); Huang et al. (1997)), cooperative architecture neocognition (Cardot et al. (1994); Foltyniewicz et al. (1996)) and adaptive resonance theory (ART) networks (Forte et al. (1996)).

Mighell, Wilkinson, and Goodman (1989) use an NN that learns through backpropagation to detect random forgeries. They use a training set that consists of 10 genuine signatures and 10 forgeries and a test set that consists of 70 genuine signatures and 56 forgeries. All of the genuine signatures belong to the same writer. They achieve an EER of 2%.

Sabourin and Drouhard (1992a) use an NN, with the PDF of the stroke directions serving

as a global characteristics vector. Their database consists of 800 signatures from 20 writers. During training, the genuine signatures of the other writers are treated as random forgeries. An FRR of 1.75% and an FAR of 9% is achieved.

Cardot, Revenu, Victorri, and Revillet (1994) use a global approach to eliminate random forgeries. They use the envelope and geometric parameters (mean stroke direction, moments of inertia and scale) of a signature. Their database consists of 6000 signatures. An FRR of 5% and an FAR of 2% is achieved.

Kaewkongka, Chamnongthai, and Thipakorn (1999) use the Hough transform (general Radon transform) to extract the parameterised Hough space from a signature skeleton as a unique characteristic feature of a signature. A backpropagation NN is used to evaluate the performance of the method. The system is tested with 70 signatures from different writers and a recognition rate of 95.24% is achieved.

Baltzakis and Papamarkos (2001) developed an NN-based system for the detection of random forgeries. Their system uses global features, grid features (pixel densities) and texture features (co-occurrence matrices) to represent each signature. For each one of these feature sets, a special two stage perceptron OCON (one-class-one-network) classification structure is implemented. In the first stage, the classifier combines the decision results of the NNs and the Euclidean distance obtained using the three feature sets. The results of the first stage classifier feed a second stage radial basis function (RBF) NN structure, which makes the final decision. A database is used which contains the signatures of 115 writers, with between 15 and 20 genuine signatures per writer. Of all the signatures in the database, 1500 are used for training purposes. The database contains 500 genuine test signatures and 57 000 random forgeries. An average FRR and FAR of 3% and 9.8% respectively is obtained.

Quek and Zhou (2002) investigate the feasibility of using a pseudo-outer product based fuzzy NN (POPFNN-TVR) for skilled forgery detection. They use global baseline features (that is the vertical and horizontal position in the signature image which corresponds to the peak in the frequency histogram of the vertical and horizontal projection of the binary image respectively), pressure features (that correspond to high pressure regions in the signature) and slant features (which are found by examining the neighbours of each pixel of the thinned signature). They then conduct two types of experiments. The first group of experiments use genuine signatures and forgeries as training data (unrealistic), while the second group of experiments use only genuine signatures as training data (realistic). These experiments are conducted on the signatures of 15 different writers, that is five writers from three different ethnic groups. For each writer, five genuine signatures and five skilled forgeries are submitted. When genuine signatures and forgeries are used as training data, the average of the individual EERs is 22.4%. They claim that comparable results are obtained when only genuine signatures are used as training data.

2.6 Structural techniques

Structural pattern recognition techniques adopt a hierarchical perspective where a pattern is viewed as being composed of simple subpatterns which are themselves built from

yet simpler subpatterns. The most elementary subpatterns to be recognised are called *primitives* and the given complex pattern is represented in terms of the interrelationships between these primitives. In structural pattern recognition, a formal analogy is drawn between the structure of patterns and the syntax of a language. The patterns are viewed as sentences belonging to a language, primitives are viewed as the alphabet of the language, and the sentences are generated according to a grammar. Thus, a large collection of complex patterns can be described by a small number of primitives and grammatical *rules*. The grammar for each pattern class must be inferred from the available training samples.

Structural methods are not particularly popular for the purpose of off-line signature verification and we briefly discuss three systems that are based on this approach. Other structural approaches are discussed in a survey article by Sabourin (1997a).

Bastos, Bortolozzi, Sabourin, and Kaestner (1997) developed a structural approach for detecting random forgeries. The writing trace of each signature is subdivided into conic sections, like straight lines, ellipses and hyperbolae. They use a database of 120 signatures from six writers, that is twenty signatures per writer. Similarity indices, that vary between 86.3% and 97.3%, are reported for the individual writers.

Ismail and Gad (2000) explore the use of fuzzy concepts for the verification of Arabic signatures. They use local features to form a primary feature set, including central line features, corner line features, central circle features, corner curve features and critical point features. These features produce signature snap shots taken from different angles and are selected in such a way that they give high importance to pixel positions and are not sensitive to noise. Instead of using a sharp threshold, a set of fuzzy rules is used to make a decision with a degree of certainty. They use a data set that contains the signatures of 22 writers, with six training signatures, four genuine test signatures and five skilled forgeries per writer. An average of 98% overall verification confidence is achieved.

Huang and Yan (2002) use a signature database that contains the signatures of 53 writers. For each writer, 24 genuine signatures, of which eight are used for training and 16 for testing, as well as 144 skilled forgeries are submitted. The forgeries are either simulated or traced. Statistical models, which are based on the pixel distribution and structural layout of the signatures, are used for an initial classification. During the initial classification the system rejects 2.2% of the genuine signatures, accepts 3.6% of the forgeries and is undecided on 32.7% of the signatures. For these "questionable" signatures, that is 32.7% of the data set, a structural feature verification algorithm is evoked. This algorithm compares the detailed structural correlation between the input and reference signatures. The system rejects 31.2% of the questionable genuine signatures and accepts 23.2% of the questionable forgeries. This implies that, for the combined classifier, an FRR of 6.3% and an FAR of 8.2% is achieved.

2.7 Support vector machines

SVMs are machine learning algorithms for binary classification based on recent advances in statistical learning theory. They were originally developed by Vapinik (1998). The

input data is mapped onto a high dimensional feature space. A linear classifier, that maximises the separation between the classes, is then constructed. SVMs therefore generalises well to unseen data. Learning requires only information about the relative distances of the training patterns, so it can be performed for arbitrary distance metrics (called kernels) that may be specific to the application domain. These generalised SVMs are called kernel machines. SVMs are therefore well-suited for verification purposes. However, since the theory of SVMs was developed quite recently, there are very few published SVM-based off-line signature verification systems.

In a recent paper Justino, Bortolozzi, and Sabourin (2004) compare SVM-based and HMM-based classifiers under two specific conditions, that is the number of samples used for training and the use of different types of forgeries. Under both of these conditions the SVM-based classifier performs better. They use a set of "graphometric" (static and pseudo-dynamic) features which are obtained trough a grid-segmentation scheme. The pixel density and center of mass for each cell are used as static features. The stroke curvature (angle of curvature of the largest stroke in each cell) and slant (predominant slant in each cell) are used as pseudo-dynamic features. They do not report any specific AERs or EERs.

2.8 Hidden Markov models

HMMs are used to model a sequence of observations and their relationship to each other, and is a *stochastic* approach to pattern recognition. We discuss HMMs in more detail in chapter 6 and in appendix B.

Although HMMs have been extensively used in handwriting recognition in the last decade, their use in off-line signature verification has been limited (Rigoll and Kosmala (1998)). In this section we discuss some recent work on off-line signature verification, that involves the use of HMMs.

El-Yacoubi, Justino, Sabourin, and Bortolozzi (2000) use HMMs and the cross-validation principle for random forgery detection. A grid is superimposed on each signature image, segmenting it into local square cells. From each cell, the pixel density is computed, so that each pixel density represents a local feature. Each signature image is therefore represented by a sequence of feature vectors, where each feature vector represents the pixel densities associated with a column of cells. The cross-validation principle involves the use of a subset (validation set) of each writer's training set for validation purposes. Since this system only attempts to detect random forgeries, subsets of other writers' training sets are used for impostor validation. Two experiments are conducted on two independent data sets. These data sets contain the signatures of 40 and 60 writers respectively. Both experiments use 20 genuine signatures (from each writer) for training and 10 for validation. Both experiments use the forgeries of the first experiment for impostor validation. Each test signature is analyzed under several resolutions and the majority-vote rule is used to make a decision. AERs of 0.46% and 0.91% are reported for the respective data sets.

Justino, Bortolozzi, and Sabourin (2001) use a discrete observation HMM to detect ran-

dom, casual and skilled forgeries. A grid segmentation scheme is used to extract three features: a pixel density feature, a pixel distribution feature (extended-shadow-code) and an axial slant feature. A cross-validation procedure is used to dynamically define the optimal number of states for each model (writer). Two data sets are used. The first data set contains the signatures of 40 writers with 40 genuine signatures per writer. This data set is used to determine the optimal codebook size for detecting random forgeries. This optimised system is then used to detect random, casual and skilled forgeries in a second data set. The second data set contains the signatures of 60 writers with 40 training signatures, 10 genuine test signatures, 10 casual forgeries and 10 skilled forgeries per writer. An FRR of 2.83% and an FAR of 1.44%, 2.50% and 22.67% is reported for random, casual and skilled forgeries respectively.

2.9 Summary

There is no standard international database of off-line signatures. It is therefore very difficult to directly compare the results reported in this chapter. However, based on their results, it seems that the systems developed by Guo et al. (2001) and Huang et al. (2002) are the most effective in detecting skilled forgeries. The system developed by El-Yacoubi et al. (2000) seems to be the most effective in detecting random forgeries. Recent work on off-line signature verification is summarised in tables 2.1 and 2.2.

We present the results for the DTW-based and HMM-based systems developed in this dissertation in section 8.4 and compare these results to those of existing systems − those systems discussed in this chapter − in section 8.5.

(A)uthors (R)epresentation (D)escription (V)erification	Database (number of writers, signatures per writer) *E: Experiment; Ω: Writers; TR: Training set; GS, SF, CF, RF: Genuine test signatures, skilled, casual, random forgeries; I: FRR; II: FAR; I+II: EER or AER*							Error rates (percentages)						
	E	**Ω**	**TR**	**GS**	**SF**	**CF**	**RF**	**GS**	**SF**		**CF**		**RF**	
								I	**II**	**I+II**	**II**	**I+II**	**II**	**I+II**
(A) El-Yacoubi et al. (2000) **(R)** Superimposed grid **(D)** Pixel density **(V)** HMM	1 2	40 60	30 30	10 10			1560 2360	0.75 1.17					0.18 0.64	0.46 0.91
• Both experiments use 20 genuine signatures for training and 10 for validation. • The forgeries of the first experiment is used for impostor validation. • Each signature is analysed under several resolutions before making a decision (majority vote rule).														
(A) Ismail et al. (2000) **(R)** Superimposed curves **(D)** Intersection points **(V)** Structural: fuzzy rules	1	22	6	4	5			Average of 98% overall verification confidence						
• This work was done on Arabic signatures.														
(A) Baltzakis et al. (2001) **(R)** Global, grid, texture **(D)** Geometric, pixel density **(V)** NN	1	115	15 to 20					3					9.8	
• Size of entire training set: 1500. Total number of genuine test signatures: 500. Total number of random forgeries: 57 000.														
(A) Fang et al. (2001) **(R)** Global shape, strokes **(D)** Geometry, smoothness **(V)** SDC	1	55	23	1	24			18.1	16.4	17.3				
• An iterative leave-one-out method is used for training and for testing genuine signatures.														
(A) Guo et al. (2001) **(R)** Stroke segments **(D)** Geometric, gray-levels **(V)** DTW	1 2	10 10	5 5	5 5		10 20		2 6		11.5	3.3 8.8	2.7		
• Skilled forgeries: 50% simulated, 50% traced. The same genuine signatures are used for training and testing.														
(A) Justino et al. (2001) **(R)** Superimposed grid **(D)** Pixel distribution, slant **(V)** HMM	1	60	40	10	10	10	590	2.8	22.7	12.8	2.5	2.7	1.4	2.1
• Another data set (1600 signatures) is first used to determine the optimal codebook size for detecting random forgeries. • This optimised system is then used to detect random, casual and skilled forgeries in a second data set.														

Table 2.1: *Off-line signature verification: summary of recent work (2000-2001).*

(A)uthors (R)epresentation (D)escription (V)erification	Database (number of writers, signatures per writer) *E: Experiment; Ω: Writers; TR: Training set; GS, SF, CF, RF: Genuine test signatures, skilled, casual, random forgeries; I: FRR; II: FAR; I+II: EER or AER*							Error rates (percentages)						
	E	**Ω**	**TR**	**GS**	**SF**	**CF**	**RF**	**GS**	**SF**		**CF**		**RF**	
								I	**II**	**I+II**	**II**	**I+II**	**II**	**I+II**
(A) Fang et al. (2002) **(R)** Signature structure **(D)** Peripheral features **(V)** Mahalanobis distance	1 2	55 55	23 23	1 1	24 24			14.7	16.5	15.6 11.4				
• Both algorithms use an iterative leave-one-out method for training and for testing genuine signatures. • The second algorithm generates 506 additional training signatures for each writer using an elastic matching method.														
(A) Huang et al. (2002) **(R)** Structural layout **(D)** Structural correlation **(V)** Statistical, structural	1	53	8	16	144			6.3	8.2					
• Signatures classified as "questionable" by a statistical classifier are either accepted or rejected by a structural classifier.														
(A) Mizukami et al. (2002) **(R)** Signature coordinates **(D)** Displacement function **(V)** Euclidean distance	1	20	10	10	10					24.9				
(A) Quek et al. (2002) **(R)** Global, pressure, slant **(D)** Histograms, geometric **(V)** NN	1 2	15 15	10 5	5 5	5 5					22.4 Similar				
• The first experiment uses genuine signatures and forgeries for training. The second experiment uses only genuine signatures.														
(A) Xiao et al. (2002) **(R)** Profiles, components **(D)** Component geometry **(V)** Bayesian classifier	1	8	10-20					20	14	17				
• Although the test set contains some skilled forgeries, this paper is not clear on the quality of the other forgeries.														
(A) Fang et al. (2003) **(R)** Profiles, "elements" **(D)** Geometric **(V)** Template matching	1 2 3	55 55 55	23 23 23	1 1 1	24 24 24					20.8 18.1 23.4				
• The first two experiments consider projection profiles and are conducted on binary and gray-scale signatures respectively. • The third experiment considers "elements", that is short straight lines that approximate the skeleton of a signature. • All of these algorithms use an iterative leave-one-out method for training and testing genuine signatures.														

Table 2.2: *Off-line signature verification: summary of recent work (2002-2003).*

Chapter 3

Simple Distance Classifiers

3.1 Introduction

In order to classify a test pattern, a minimum distance classifier represents (models) each pattern class with a *PDF* and then calculates the distance between the test pattern and each of these PDFs. The pattern is then assigned to the class for which the distance between the pattern and the corresponding PDF is the smallest.

Two types of minimum distance classifiers are particularly relevant to this work, that is SDCs and HMMs. An SDC considers each pattern to be a *single* observation, that is a single d-dimensional feature vector. An HMM, on the other hand, considers each pattern to be a sequence of feature vectors, that is an observation sequence, and also models the relationship of these observations to each other. HMMs are discussed in chapter 6.

We use the following notation for a single d-dimensional feature vector,

$$\mathbf{x} = [x_1, x_2, \ldots, x_d]', \tag{3.1}$$

where $'$ denotes the transpose.

For an SDC, each pattern class $\omega_j,\ j = 1, \ldots, \Omega$ is typically represented by a Gaussian PDF in a d-dimensional feature space,

$$f(\mathbf{x}|\omega_j) = \frac{1}{(2\pi)^{d/2}|\mathbf{\Sigma}_j|^{1/2}}\, e^{-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu}_j)'\,\mathbf{\Sigma}_j^{-1}\,(\mathbf{x}-\boldsymbol{\mu}_j)}, \tag{3.2}$$

where Ω represents the number of classes.

Each PDF $f(\mathbf{x}|\omega_j)$ is uniquely defined by a *mean vector* $\boldsymbol{\mu}_j$ and *covariance matrix* $\mathbf{\Sigma}_j$. The mean vector and covariance matrix are estimated from a set of sample patterns, that is a training set, for the particular class. The mean vector for pattern class ω_j is given by

$$\boldsymbol{\mu}_j = \frac{1}{N_j}\sum_{k=1}^{N_j}\mathbf{x}_k \tag{3.3}$$

and the covariance matrix by

$$\boldsymbol{\Sigma}_j = \frac{1}{N_j - 1} \sum_{k=1}^{N_j} (\mathbf{x}_k - \boldsymbol{\mu}_j)(\mathbf{x}_k - \boldsymbol{\mu}_j)', \tag{3.4}$$

where $\mathbf{x}_k \in \omega_j$ and N_j denotes the number of sample vectors for class ω_j.

The covariance matrix for pattern class ω_j can be written as follows in matrix notation,

$$\boldsymbol{\Sigma}_j = \begin{pmatrix} \sigma_1^2 & \sigma_{12} & \sigma_{13} & \cdots & \sigma_{1d} \\ \sigma_{21} & \sigma_2^2 & \sigma_{23} & \cdots & \sigma_{2d} \\ \sigma_{31} & \sigma_{32} & \sigma_3^2 & \cdots & \sigma_{3d} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \sigma_{d1} & \sigma_{d2} & \sigma_{d3} & \cdots & \sigma_d^2 \end{pmatrix}_j . \tag{3.5}$$

The diagonal component σ_i^2 represents the variance of the ith feature, while the off-diagonal component $\sigma_{ij},\ i \neq j$ represents the correlation between the ith and jth features. Note that, in order to train a PDF, $d(d+3)/2$ parameters have to be estimated, that is $d(d+1)/2$ parameters for the covariance matrix and an additional d parameters for the mean vector. When only a few sample patterns are available, one may opt to estimate only a certain number of these parameters, while the other parameters are kept fixed (see section 3.4).

The test pattern $\mathbf{x}_{\text{Test}}$ is assigned to pattern class ω_i if

$$f(\mathbf{x}_{\text{Test}}|\omega_i) > f(\mathbf{x}_{\text{Test}}|\omega_j),\ j = 1, 2, \ldots, \Omega;\ j \neq i, \tag{3.6}$$

or when

$$D(\mathbf{x}_{\text{Test}}, f(\mathbf{x}|\omega_i)) < D(\mathbf{x}_{\text{Test}}, f(\mathbf{x}|\omega_j)),\ j = 1, 2, \ldots, \Omega;\ j \neq i, \tag{3.7}$$

where D denotes a *distance* between the test pattern and the particular PDF.

The *decision boundary* that separates class ω_i from class ω_j is given by those points in the feature space $\mathbf{x} \in \mathbb{R}^d$ for which

$$D(\mathbf{x}, f(\mathbf{x}|\omega_i)) = D(\mathbf{x}, f(\mathbf{x}|\omega_j)). \tag{3.8}$$

We discuss a number of distance measures that can be used to depict the dissimilarity between a test pattern and a particular PDF. The nature and accuracy of the particular distance measure depends on how many parameters of the corresponding PDF are trained. We first focus on the *Euclidean* and *Mahalanobis* distances and the proceed to discuss other distance measures.

We shall illustrate the concepts discussed in this chapter using a simple example. We consider a classifier that attempts to classify a person as either a rugby player (class ω_1) or a hockey player (class ω_2) based on only two measurements, that is the person's body mass in kilograms (feature x_1) and the person's height in centimetres (feature x_2). For this example we have that $\Omega = d = 2$. There are one hundred sample patterns for each class, that is $N_1 = N_2 = 100$. We refer to this example as the *rugby/hockey example*. A hypothetical data distribution for the two classes is shown in figure 3.1

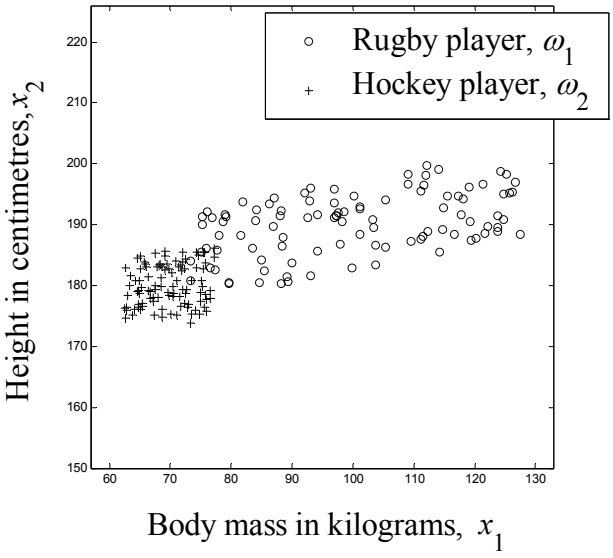


Figure 3.1: *Rugby/hockey example: hypothetical data distribution.*

3.2 The Euclidean distance

When we use the Euclidean distance to depict the dissimilarity between a test pattern and a PDF, we invariably assume that the covariance matrix (3.4) is not estimated. This implies that only the mean vector (3.3) is used to model the pattern class. This mean vector can always be estimated, even when there is only one sample available. The Euclidean distance between a test vector $\mathbf{x}_{\text{Test}}$ and the pattern class ω_j is given by

$$D_{\text{Eucl}}(\mathbf{x}_{\text{Test}}, \omega_j) = \sqrt{(\mathbf{x}_{\text{Test}} - \boldsymbol{\mu}_j)'(\mathbf{x}_{\text{Test}} - \boldsymbol{\mu}_j)}. \tag{3.9}$$

The test vector $\mathbf{x}_{\text{Test}}$ is assigned to pattern class ω_i if

$$D_{\text{Eucl}}(\mathbf{x}_{\text{Test}}, \omega_i) < D_{\text{Eucl}}(\mathbf{x}_{\text{Test}}, \omega_j),\ j = 1, 2, \ldots, \Omega;\ j \neq i, \tag{3.10}$$

and the decision boundary that separates class ω_i from class ω_j is given by those points in the feature space $\mathbf{x} \in \mathbb{R}^d$ for which

$$D_{\text{Eucl}}(\mathbf{x}, \omega_i) = D_{\text{Eucl}}(\mathbf{x}, \omega_j). \tag{3.11}$$

It is easy to show that the surface given by (3.11) is simply the perpendicular bisector of the line segment joining $\boldsymbol{\mu}_i$ and $\boldsymbol{\mu}_j$. For $d = 2$ and $d = 3$ the perpendicular bisector is a line and a plane respectively.

We recall that, when the Euclidean distance is used to depict the dissimilarity between a test pattern and a PDF, the covariance matrix is not trained. This implies that for each PDF only d parameters (that is the dimension of the corresponding mean vector) are

trained. This also implies that each pattern class is represented by a rigid PDF, which only describes the position of the data in the feature space, but not the variability of the data. This is often accomplished by assuming that

$$\mathbf{\Sigma}_j^{(\text{Eucl})} = I/2,\ j = 1, \ldots, \Omega, \tag{3.12}$$

where I denotes the identity matrix. Given this assumption, (3.2) simplifies to

$$f_{\text{Eucl}}(\mathbf{x}|\omega_j) = \frac{1}{\pi^{d/2}}\, e^{-(\mathbf{x}-\boldsymbol{\mu}_j)'(\mathbf{x}-\boldsymbol{\mu}_j)}. \tag{3.13}$$

This implies that, as an alternative to (3.11), the decision boundary that separates class ω_i from class ω_j is also given by those points in the feature space $\mathbf{x} \in \mathbb{R}^d$ for which

$$f_{\text{Eucl}}(\mathbf{x}|\omega_i) = f_{\text{Eucl}}(\mathbf{x}|\omega_j). \tag{3.14}$$

Note that when we take the negative of the natural logarithm on both sides of (3.13), we obtain the following,

$$-\ln(f_{\text{Eucl}}(\mathbf{x}|\omega_j)) = D^2_{\text{Eucl}}(\mathbf{x}, \omega_j) + (d/2)\ln\pi. \tag{3.15}$$

This implies that the squared Euclidean distance $D^2_{\text{Eucl}}(\mathbf{x}, \omega_j)$ differs from the negative loglikelihood $-\ln(f_{\text{Eucl}}(\mathbf{x}|\omega_j))$ by only the constant term $(d/2)\ln\pi$. The negative loglikelihood is often used to match a test pattern with an HMM (see chapter 6).

We now return to the rugby/hockey example. Figure 3.2 illustrates how an SDC, that is based on the Euclidean distance, works. The bullets (large dots) represent the mean vectors for the two pattern classes. The solid lines represent a superimposed contour plot for the respective PDFs. This contour plot uses a gray-scale colourmap with lower heights indicated with a lighter shade of gray. For convenience, one of these contours is represented by a thicker line. This line does not necessarily represent one standard deviation. The decision boundary (dashed line) represents those points in the feature space where two contours of similar height (one from each PDF) intersect. It is clear that this decision boundary coincides with the perpendicular bisector of the line segment (dotted line) that joins the mean vectors.

The mean vectors are given by

$$\boldsymbol{\mu}_1 = \begin{pmatrix} 99.8 \\ 190.1 \end{pmatrix}, \quad \boldsymbol{\mu}_2 = \begin{pmatrix} 69.6 \\ 180.0 \end{pmatrix} \tag{3.16}$$

and the covariance matrices for both classes are equal to $0.5I$.

From figure 3.2 it is clear that several patterns that belong to pattern class ω_1 and no patterns that belong to pattern class ω_2 are misclassified. It is also clear that the main reason for the high number of misclassifications is the fact that the above model does not consider the fact that the samples for pattern class ω_1 vary more than those for pattern class ω_2. The orientation (rotation) of the data is also not taken into account.

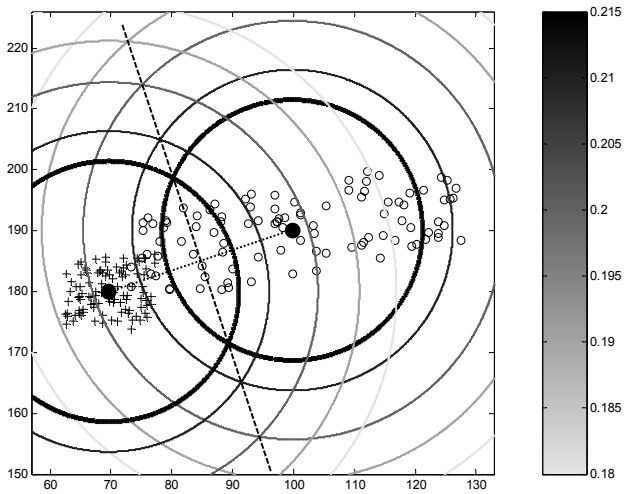


Figure 3.2: *Rugby/hockey example: an SDC that is based on the Euclidean distance.*

3.3 The Mahalanobis distance

In the previous section we discussed an SDC that is based on the Euclidean distance, that is an SDC that uses only the mean vector for each training set to model the corresponding pattern class. When, in addition to the mean vector, the covariance matrix is also estimated (trained), the corresponding PDF is given by (3.2). We refer to this matrix as a *full* covariance matrix, since *all* of its components are estimated (trained).

When both the mean vector and the full covariance matrix are estimated, classification is based on the so-called Mahalanobis distance. We therefore use the following notation for the PDF that represents pattern class ω_j,

$$f_{\text{Mah}}(\mathbf{x}|\omega_j) = f(\mathbf{x}|\omega_j), \tag{3.17}$$

where $f(\mathbf{x}|\omega_j)$ is given by (3.2)

Note that for each PDF $d(d+3)/2$ parameters (that is $d(d+1)/2$ parameters for the corresponding covariance matrix and an additional d parameters for the corresponding mean vector) are trained. Since we now also take the variability and rotation of the data into account, the distribution of the data can be modelled much more accurately. This implies that better decision boundaries can be obtained, which will result in a better classification.

The decision boundary that separates classes ω_i and ω_j is given by those points in the feature space $\mathbf{x} \in \mathbb{R}^d$ for which

$$f_{\text{Mah}}(\mathbf{x}|\omega_i) = f_{\text{Mah}}(\mathbf{x}|\omega_j), \tag{3.18}$$

or alternatively, for which

$$D_{\text{Mah}}(\mathbf{x}, f_{\text{Mah}}(\mathbf{x}|\omega_i)) = D_{\text{Mah}}(\mathbf{x}, f_{\text{Mah}}(\mathbf{x}|\omega_j)). \tag{3.19}$$

This implies that when we choose two or more test vectors (points in the feature space) in such a way that the height (likelihood) of a PDF is the same at all of these points, the Mahalanobis distance from these points to the PDF will also be the same.

The test vector $\mathbf{x}_{\text{Test}}$ is therefore assigned to pattern class ω_i if

$$f_{\text{Mah}}(\mathbf{x}_{\text{Test}}|\omega_i) > f_{\text{Mah}}(\mathbf{x}_{\text{Test}}|\omega_j),\ j = 1, 2, \ldots, \Omega;\ j \neq i, \tag{3.20}$$

or if

$$D_{\text{Mah}}(\mathbf{x}_{\text{Test}}, f_{\text{Mah}}(\mathbf{x}|\omega_i)) < D_{\text{Mah}}(\mathbf{x}_{\text{Test}}, f_{\text{Mah}}(\mathbf{x}|\omega_j)),\ j = 1, 2, \ldots, \Omega;\ j \neq i. \tag{3.21}$$

We now return to the rugby/hockey example. Figure 3.3 illustrates how an SDC, that is based on the Mahalanobis distance, works. The bullets (large dots) again represent the mean vectors for the two pattern classes. The solid lines represent a superimposed contour plot for the respective PDFs. This contour plot uses a gray-scale colourmap with lower heights indicated with a lighter shade of gray. For convenience, one of these contours is represented by a thicker line. This line does not necessarily represent one standard deviation. A specific contour (associated with a specific PDF) represents points in the feature space for which the Mahalanobis distance to the PDF is the same. The decision boundary represents those points in the feature space where two contours of similar height (one from each PDF) intersect. In order to get some idea of what the decision boundary looks like, a few of these intersection points are indicated with asterisks.

The covariance matrices for the respective classes are given by

$$\boldsymbol{\Sigma}_1^{(\text{Full})} = \begin{pmatrix} 270.1 & 36.6 \\ 36.6 & 24.0 \end{pmatrix}, \quad \boldsymbol{\Sigma}_2^{(\text{Full})} = \begin{pmatrix} 17.8 & 2.1 \\ 2.1 & 11.1 \end{pmatrix}. \tag{3.22}$$

The Euclidean distance, that was used in the previous section, did not result in a particularly good classification (see figure 3.2). From figure 3.3 it is clear that a better separation is obtained when the Mahalanobis distance is used.

3.4 Other distance measures and the curse of dimensionality

When the number of samples that is available for one or more of the pattern classes is much less than the dimension of the feature space, it is not feasible to use the Mahalanobis distance for classification purposes. In order to use equation (3.2), each covariance matrix $\boldsymbol{\Sigma}_j$ needs to be inverted. When only a few training samples are available, the estimation of the covariance matrices is unreliable and in some cases one or more of the covariance

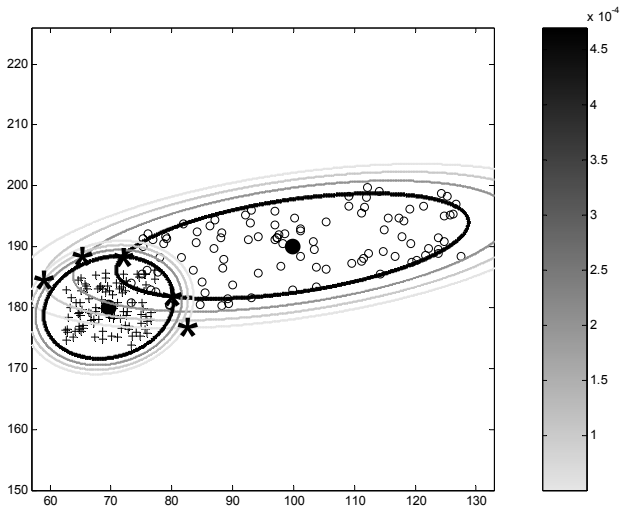


Figure 3.3: *Rugby/hockey example: an SDC that is based on the Mahalanobis distance.*

matrices may even be singular. This dilemma is commonly referred to as the *"curse of dimensionality"*.

One way to address this problem is to use the Euclidean distance (section 3.2) instead of the Mahalanobis distance (section 3.3). When the Euclidean distance is used, only the mean vectors for the respective PDFs need to be estimated. This is not the ultimate solution though, since this approach does not address the variability of the data in any way.

When the variability of the data is important, as is the case for the rugby/hockey example, this approach invariably leads to bad classification results. In these situations therefore, alternative distance measures have to be considered. These distance measures are derived from using covariance matrices for which only a certain number of parameters are trained, while the other parameters are kept fixed.

It is also possible to deal with the "curse of dimensionality" (to some extent) by using a suitable dimension reduction technique (see section 3.4.2).

3.4.1 Rotation invariant ellipsoidal density functions

One way to obtain better estimates for the covariance matrices from limited data, is to assume that the different features are uncorrelated. We therefore assume that the data for each pattern class is aligned with the principal axes of the feature space. Rotational variations of the data with respect to the principal axes are therefore ignored.

This can be accomplished by simply setting the off-diagonal components of each covari-

ance matrix equal to zero. This implies that the covariance matrix for pattern class ω_j is now defined as follows,

$$\mathbf{\Sigma}_j^{(\text{Diag})} = \begin{pmatrix} \sigma_1^2 & 0 & 0 & \cdots & 0 \\ 0 & \sigma_2^2 & 0 & \cdots & 0 \\ 0 & 0 & \sigma_3^2 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & \sigma_d^2 \end{pmatrix}_j . \tag{3.23}$$

We refer to this matrix as a *diagonal* covariance matrix. We therefore assume that the components of the feature vectors, that is the individual features (measurements), are uncorrelated. Note that for each PDF, $2d$ parameters (that is d parameters for the diagonal components of the corresponding covariance matrix and an additional d parameters for the corresponding mean vector) are trained.

We now return to the rugby/hockey example. Figure 3.4 illustrates how an SDC, that is based on the assumption that the different features are uncorrelated, works. The figure again shows the sample distribution of the data from the respective pattern classes (plusses and circles), the mean vector for each class (bullets), a superimposed contour plot of the respective PDFs (solid lines) and the decision boundary (asterisks).

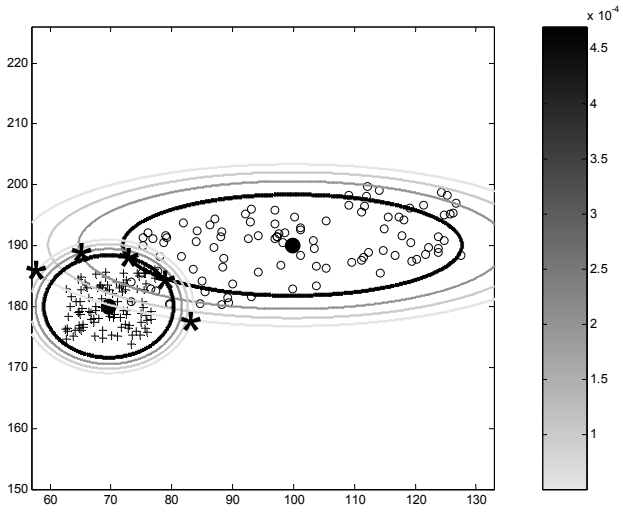


Figure 3.4*: Rugby/hockey example: an SDC that is based on the assumption that the different features are uncorrelated.*

The covariance matrices for the respective pattern classes are now given by

$$\mathbf{\Sigma}_1^{(\text{Diag})} = \begin{pmatrix} 270.1 & 0 \\ 0 & 24.0 \end{pmatrix}, \quad \mathbf{\Sigma}_2^{(\text{Diag})} = \begin{pmatrix} 17.8 & 0 \\ 0 & 11.1 \end{pmatrix}. \tag{3.24}$$

From figure 3.4 it is clear that the contours of the respective PDFs are ellipses, that are aligned with the principal axes of the feature space, and that the rotational variations of the data are ignored.

We summarise the simple distance measures discussed in this chapter in table 3.1.

PDF	Distance measure	Number of parameters trained
Fixed	Euclidean	d
Ellipsoid (aligned)		$2d$
Ellipsoid (rotated)	Mahalanobis	$d(d+3)/2$

Table 3.1: *Summary of simple distance measures.*

3.4.2 Dimension reduction

In order to better deal with the problems associated with high dimensional feature vectors, a dimension reduction technique is sometimes applied to the feature vectors. This can be accomplished by applying the KL-transform (Karhunen-Loève transform)[1] to each feature vector. This transform decorrelates the features so that the off-diagonal components of the covariance matrix for the transformed feature vectors are minimised. This transform also rotates the feature vectors so that the data is aligned with the principal axes. This enables one to represent the data more accurately with a rotation invariant ellipsoidal density function. It is also possible to discard the least significant transformed features, that is those features that vary the least, therefore reducing the dimension of the feature space. For a detailed discussion of the KL-transform and dimension reduction techniques, the reader is referred to Gonzalez and Woods (2002), p. 675.

3.5 Concluding remarks

We included a chapter on SDCs, in order to place certain aspects of the DTW-based and HMM-based off-line signature verification systems developed in this dissertation into perspective. The DTW-based and HMM-based systems are discussed in chapters 5 and 6 respectively.

In chapter 5 we note that the Euclidean distance is a special case of the more general *DTW-based distance*. A "bandwidth" is selected so as to restrict the search for the DTW algorithm developed in this dissertation. The Euclidean distance between two vectors is obtained when this bandwidth is restricted to zero.

[1]This transform is sometimes referred to as the Hotelling transform. This is also called principal component analysis or linear discriminant analysis.

In chapter 6 we point out that the HMM-based system developed in this dissertation models each writer's signature with a specialised first order, continuous HMM with a ring structure. Due to the high dimension of the feature vectors, and the small number of available sample signatures, each state is represented by an Euclidean PDF.

In the next chapter we discuss the feature extraction process for both of these systems.

Chapter 4

Image Processing and Feature Extraction

4.1 Introduction

In this chapter we explain how the DTW-based and HMM-based systems developed in this dissertation extract a sequence of feature vectors from a raw signature image. The feature extraction techniques for these two systems are very similar, and only differ in the sense that the DTW-based system aligns each extracted observation sequence with a reference sequence, in order to ensure rotation invariance. This alignment is not required for the HMM-based system. For the DTW-based system, each reference sequence acts as a template for the corresponding writer's signature.

The feature extraction techniques for both of the systems developed in this dissertation are based on the calculation of the *DRT*. Image processing is also required to remove speckle noise and to normalise each signature, that is to ensure that each observation sequence is a translation, rotation and scale invariant representation of the corresponding signature image. The calculation of the DRT requires the bulk of the floating point operations during the feature extraction process.

In section 4.2 we give a brief description of the DRT. In section 4.3.1 we discuss the *basic* feature extraction algorithm. This algorithm is utilised by both the DTW-based and HMM-based systems developed in this dissertation. In section 4.3.2 we explain how the DTW-based system aligns an extracted observation sequence with a reference sequence. Finally, in section 4.4, we accentuate the *advantages* of basing the feature extraction technique on the calculation of the DRT.

4.2 The discrete Radon transform

The DRT is obtained when projections or shadows of an image (or any matrix) are calculated at equally distributed angles between 0^{O} and 180^{O}. The DRT is also a matrix, where each column represents a projection or shadow of the original image at a certain angle. We now discuss the algorithm that is used to calculate the DRT.

Algorithm. Assume that an image consists of Ψ pixels in total, and that the intensity of the ith pixel is denoted by $I_i,\ i = 1, \ldots, \Psi$. The DRT is calculated using N_φ non-overlapping beams per angle and N_Θ angles in total. The cumulative intensity of the pixels that lie within the jth beam is denoted by $R_j,\ j = 1, \ldots, N_\varphi N_\Theta$. This is called the jth beam-sum. In discrete form the Radon transform can therefore be expressed as follows

$$R_j = \sum_{i=1}^{\Psi} w_{ij} I_i, \quad j = 1, 2, \ldots, N_\varphi N_\Theta, \tag{4.1}$$

where w_{ij} indicates the contribution of the ith pixel to the jth beam-sum (see figure 4.1).

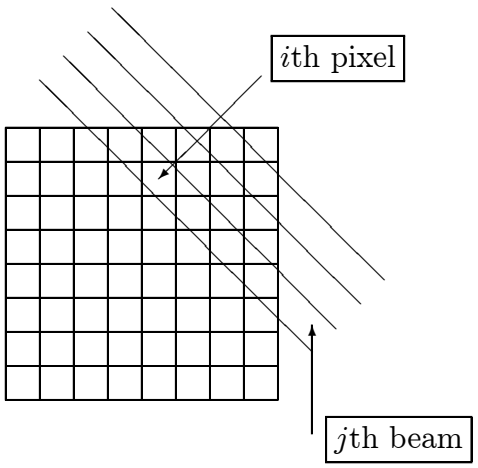


Figure 4.1: *Discrete model for the Radon transform with $w_{ij} \approx 0.9$. This implies that the jth beam overlaps approximately 90% of the ith pixel.*

The value of w_{ij} is found through two-dimensional interpolation. Each projection therefore contains the beam-sums that are calculated at a given angle. The accuracy of the DRT is determined by N_Θ (the number of angles), N_φ (the number of beams per angle), and the accuracy of the interpolation method. A typical signature and its DRT are shown in figure 4.2. It is important to note that the background in figure 4.2 (a) is mapped to zero and the pen strokes to one. This differs from the usual representation of white = 1 and black = 0. The colourmap used in figure 4.2 (b) is therefore also the inverse of the usual representation. This colourmap is used throughout the dissertation.

Note that the continuous form of the Radon transform can be inverted through analytical means. The DRT therefore contains the same information[1] as the original image and

[1]The digital inversion of the DRT has important applications in computerised tomography (Kak and

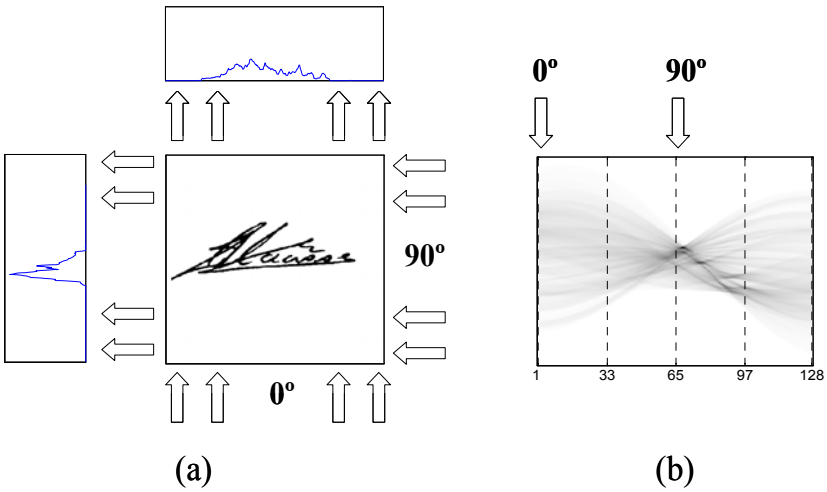


Figure 4.2: *A signature and its DRT . (a) A signature and its projections calculated at angles of* 0^{O} *and* 90^{O}*. (b) The DRT displayed as a gray-scale image. This image has* $N_\Theta = 128$ *columns, where each column represents a projection.*

can be efficiently calculated with an algorithm by Bracewell (1995). The theory and implementation of the DRT are discussed in detail in Peter Toft's Ph.D. thesis (Toft (1996)).

4.3 Feature extraction

The DTW-based and HMM-based systems developed in this dissertation use the same feature extraction algorithm to obtain an *initial observation sequence* from a raw signature image. This *basic feature extraction algorithm* is discussed in section 4.3.1.

For the HMM-based system, each writer's signature is represented by an HMM. These HMMs are constructed in such a way that each initial observation sequence already constitutes a rotation invariant representation of the corresponding signature. No further processing is required. We elaborate on this in chapter 6.

The DTW-based system, on the other hand, represents each writer's signature with a single (reference) observation sequence, which acts as a template. Each initial observation sequence is aligned with the writer's reference sequence, so as to ensure that the aligned sequence is a rotation invariant representation of the corresponding signature image. Each aligned sequence constitutes a *final observation sequence*. The *alignment* process is discussed in section 4.3.2. The feature extraction techniques for the HMM-based and DTW-based systems are compared in figure 4.3.

Slaney (1988)). The convolution back-projection algorithm is commonly used to reconstruct images from projections (CAT-scans).

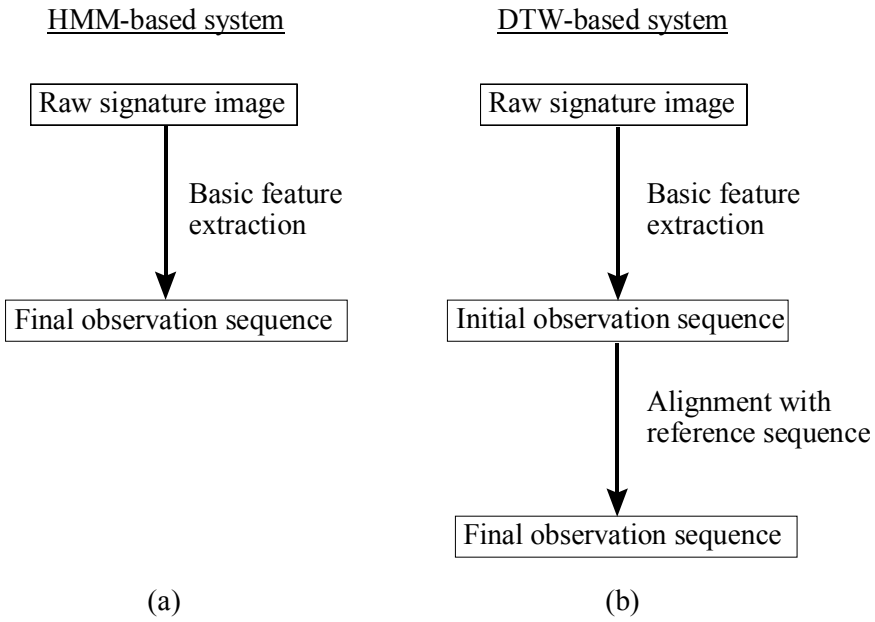


Figure 4.3*: Comparison of the feature extraction techniques for the (a) HMM-based and (b) DTW-based systems developed in this dissertation.*

4.3.1 Basic feature extraction algorithm

As we mentioned in section 1.5.2, we test the systems developed in this dissertation on two different data sets. Both of these data sets contain static signature images. We captured the signatures for the "Stellenbosch data set" with a standard flat-bed scanner. Each of these signatures was produced on a blank sheet of paper within a specified bounding box. However, the signatures for "Dolfing's data set" were originally captured on-line. We transformed the dynamic data for each signature into an image, using only the spatial coordinates of the writing. The image acquisition algorithms for these two data sets are discussed in more detail in section 8.2. We therefore assume in this section that a binary image has already been acquired for each signature.

The basic feature extraction algorithm is illustrated in figure 4.4. On average, a signature image has a width of 400 to 600 pixels and a height of 200 to 400 pixels (see figure 4.4 (a)). The image dimensions are not normalised and median filtering is applied to remove speckle noise. Subsequently the DRT of the signature image is calculated, using the algorithm discussed in section 4.2.

Each column of the DRT represents a projection or shadow of the signature at a certain angle (see figure 4.4 (b) and (d)). After these projections are processed and normalised, they represent an initial set of feature vectors (initial observation sequence) for the signature in question.

The systems developed in this dissertation calculate the DRT at N_Θ angles. These angles are equally distributed between 0^{o} and 180^{o}. The dimension of each projection is subsequently altered from N_φ to d. This is done by first decimating (removing) all the zero-valued components from each projection. These decimated vectors are then shrunk

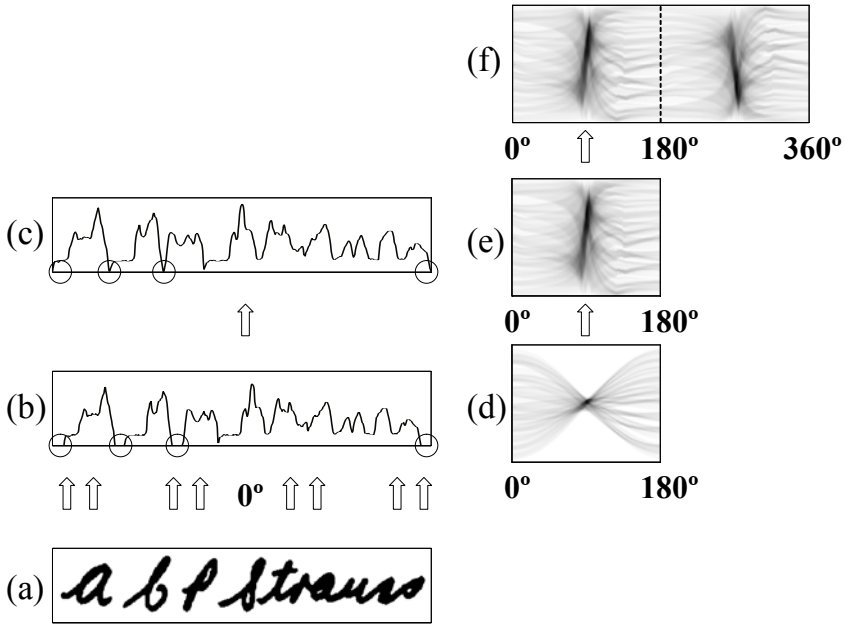


Figure 4.4: *The basic feature extraction algorithm. (a) A signature image. (b) The projection of (a), calculated at an angle of* 0^{O}*. The zero-valued components are marked with circles. (c) The projection in (b), after the zero-valued components were removed. (d) The DRT of the image in (a). The first column represents the projection in (b). (e) The DRT in (d), after the zero-valued components of each projection were removed. The first column represents the projection in (c). (f) The initial observation sequence. Each column represents a feature vector.*

or expanded to a length of d through linear[2] interpolation (see figure 4.4 (c) and (e)). Although almost all the information in the original signature image is contained in the projections at angles that range from 0^{O} to 180^{O}, the projections at angles that range from 180^{O} to 360^{O} are also included in the observation sequence. These additional projections are added to the sequence, in order to ensure rotation invariance. Since these projections are simply reflections of the projections already calculated, no additional calculations are necessary (see figure 4.4 (f)). An observation sequence therefore consists of $T = 2N_{\Theta}$ feature vectors, that is $\mathbf{X}_1^T = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_T\}$.

Each vector is subsequently normalised by the variance of the intensity of the entire set of T feature vectors. Each signature pattern is therefore represented by an observation sequence that consists of T observations, where each observation is a feature vector of dimension d.

[2]The use of more accurate interpolation schemes does not improve the performance of the systems developed in this dissertation.

4.3.2 DTW-based system: observation sequence alignment

As we explained earlier, the DTW-based system developed in this dissertation represents each writer's signature with a reference observation sequence, which acts as a template. Consequently rotation invariance can only be achieved when each initial observation sequence is properly aligned with the claimed writer's reference sequence, before they are compared. This alignment is not necessary for the HMM-based system.

Two methods can be used to achieve this alignment. The *"linear method"* linearly aligns the observation sequences by shifting all the initial observations the appropriate number of times towards the left or towards the right. Alternatively, the *"non-linear method"* (DTW-based method) can be used to align the observation sequences in a non-linear way.

The alignment is achieved in two steps. During the *first step*, a reference observation sequence is obtained for each writer. During the *second step*, the initial observation sequence for each input signature − this can be a training signature or a test signature − is aligned with the claimed writer's reference sequence. We now discuss these steps in more detail.

Step 1: Obtaining a reference sequence. For each writer, one reference sequence is selected from the initial observation sequences in the writer's training set. This sequence can be selected at random or the most representative training sequence can be used. One way to obtain the most representative training sequence is to compare every training sequence with every other training sequence. The sequence that differs the least from the other training sequences is selected.

Two training sequences can be compared in one of the following ways. When we assume that the training signatures are already more or less normalised with respect to rotational variations − this is a realistic assumption, since all the training signatures are typically obtained during one enrollment session − the *corresponding* observations for the respective signatures are compared using *DTW-based feature vector alignment*. We present this algorithm in section 5.3. The observations are therefore not aligned first. The distance between the two signatures is simply the average of the distances between the corresponding observations. When we assume that the training signatures are not normalised with respect to rotational variations, the respective observation sequences are aligned first and then compared by matching the *aligned* observations. This type of alignment is also used when a test sequence is matched with a reference sequence.

Step 2: Observation sequence alignment.

Linear method. The linear method linearly aligns the initial observation sequence for an input signature with a reference sequence. One way to achieve this, is to first compare the *corresponding* observations using a DTW-based feature vector alignment algorithm (section 5.3). The average of the DTW-based distances between these observations is then obtained.

Every input observation is subsequently shifted one place towards the right. The first observation replaces the second one, the second observation replaces the third one, etc. The periodic nature of the DRT, when calculated through 360°, enables us to replace the first observation with the last one. The shifted observation sequence is now compared

with the most representative sequence and the average of the DTW-based distances is again obtained. This process is repeated until the initial observation sequence is shifted a total of T times, where T is the length of each observation sequence.

The number of shifts is optimal when it results in an alignment, for which the distance between the observation sequences is a minimum. The distance between these optimally aligned sequences represents the distance between the signatures in question. When this approach is used to linearly align two observation sequences, $4{N_\Theta}^2$ DTW-based distances have to be calculated. It is however possible to linearly align two observation sequences in a more efficient way − this algorithm is discussed in the next few paragraphs. The optimal linear alignment of a test sequence with a reference sequence, using this *"inefficient" linear approach*, is illustrated in figure 4.5.

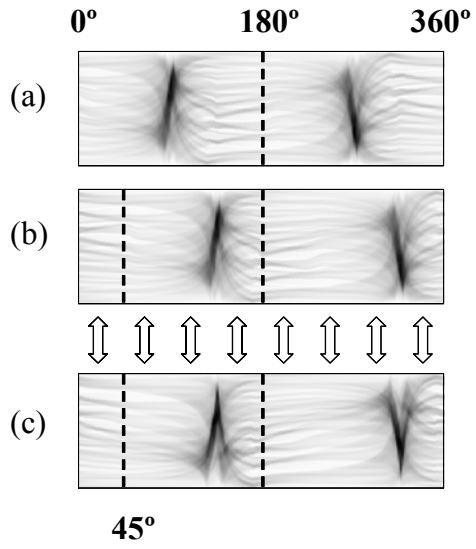


Figure 4.5*: Inefficient linear method for observation sequence alignment. (a) The initial observation sequence for an input signature, displayed as a gray-scale image. This sequence consists of* 256 *observations. (b) The observation sequence in (a) after it was shifted* 45^{O}*, that is* 32 *observations, to the right. The sequence in (a) is shifted in such a way that its individual observations are optimally aligned with those of the reference sequence. (c) The reference sequence, displayed as a gray-scale image. In order to ensure rotation invariance, this type of alignment is necessary for the DTW-based system developed in this dissertation.*

We now present a more efficient algorithm for the linear alignment of two observation sequences. When this algorithm is used, it is possible to provide for *any* possible rotation, that is rotations of up to 180° (in a clockwise (CW) or counter-clockwise (CCW) direction) with respect to a reference signature, by calculating only $2{N_\Theta}^2$ DTW-based distances.

We illustrate the *"efficient" linear approach* in figure 4.6. The grid in the top-right section consists of $4{N_\Theta}^2$ nodes, where each node relates an observation (component) of a reference sequence with the corresponding observation (component) of a test sequence. When the test sequence (on the vertical axis, labeled 0°) and the reference sequence (on

the horizontal axis) are identical, the optimal alignment is represented by the solid line on the diagonal. The observation sequences are represented by rectangular blocks. The two shades of gray provide a position of reference on each sequence. One half of the observations is represented by a dark-gray block, while the other half is represented by a light-gray block. We recall from section 4.2, that the observations represented by the light-gray block are simply reflections of the observations represented by the dark-gray block.

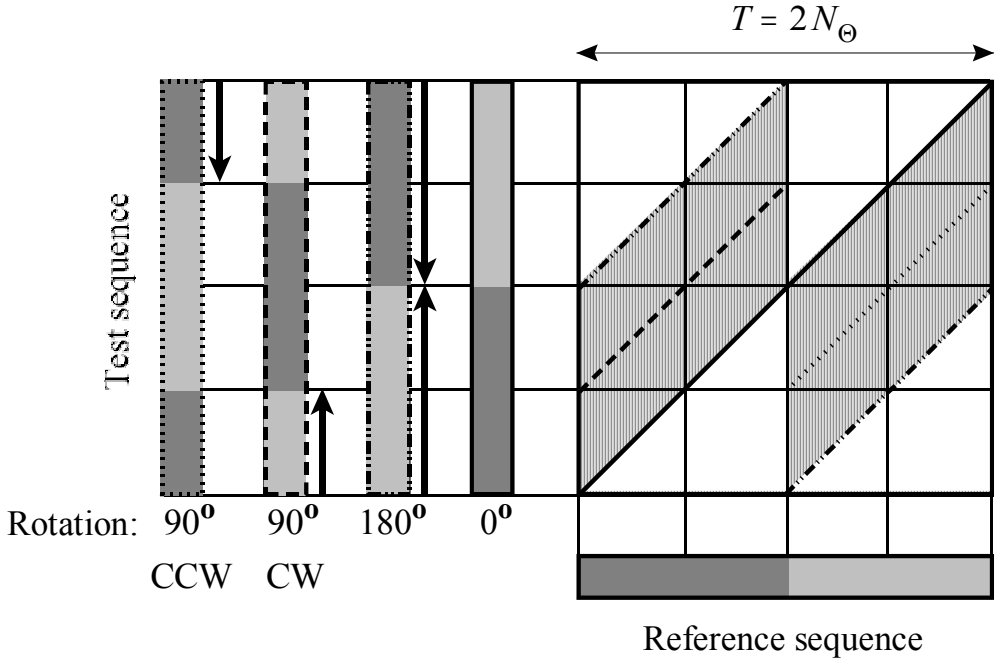


Figure 4.6*: Efficient linear method for observation sequence alignment. One half of the observations is represented by a dark-gray block, while the other half is represented by a light-gray block. The observations represented by the light-gray block are simply reflections of the observations represented by the dark-gray block. In order to provide for any possible rotation, it is only necessary to consider the nodes within the two checkered regions (parallelograms). In order to provide for rotations through an angle of up to* 90°*, it is only necessary to consider half of the nodes within the checkered regions, that is the nodes bounded by the parallelogram with the dashed line at the top and the solid line at the bottom, and the nodes bounded by the parallelogram with the solid line at the top and the dotted line at the bottom.*

Suppose that a test signature is rotated through an angle of 180° (in a CW or CCW direction) with respect to a reference signature. This is the worst-case scenario. Since an observation sequence is obtained via the calculation of the DRT, a rotation through an angle of 180° is equivalent to a shift of N_Θ observations. Since the DRT has a period of 360° ($2N_\Theta$ observations), a shift of N_Θ observations (in any direction) will result in the sequence labeled 180°. The arrows indicate the possible shifts. The dash-double-dotted lines represent two possible optimal alignments for this scenario. It is important to note that, since the observations represented by the light-gray block are simply reflections of the observations represented by the dark-gray block, only those observations aligned

along *one* of the two dash-double-dotted lines need to be compared. Therefore, in order to provide for *any* possible rotation, it is only necessary to consider the nodes within the two checkered regions (parallelograms). These nodes represent half the total number of nodes in the grid, that is $2{N_\Theta}^2$ nodes.

When all the signatures are produced on a reference line (for example on a cheque), it is reasonable to assume that they will only be rotated up to a certain angle (in a CW or CCW direction) with respect to a reference signature. Should one deem it sufficient to provide for only small rotational variations, it is possible to further reduce the computational requirements. For example, in order to provide for rotations through an angle of up to 90° (in a CW or CCW direction) with respect to a reference signature, one only needs to calculate ${N_\Theta}^2$ DTW-based distances.

We again use figure 4.6 to illustrate this. Suppose that a test signature is rotated through an angle of 90° (in a CW or CCW direction) with respect to a reference signature, and that this is the maximum rotation we want to provide for. A rotation through an angle of 90° is equivalent to a shift of $N_\Theta/2$ observations. Let us assume that a rotation of 90° in a CW direction results in a shift of $N_\Theta/2$ observations in an upwards direction (the sequence labeled 90° CW), and that a rotation of 90° in a CCW direction results in a shift of $N_\Theta/2$ observations in a downwards direction (the sequence labeled 90° CCW). The arrows indicate these shifts. The directions of these shifts (upwards or downwards) depend on whether the angle is incremented in a CW or CCW direction, when a DRT is calculated. The optimal alignments for these two scenarios are represented by the dashed and dotted lines, respectively. It is important to note that, since the observations represented by the light-gray block are simply reflections of the observations represented by the dark-gray block, *only* the dark-gray observations are compared along the dashed line, and *only* the light-gray observations are compared along the dotted line. Therefore, in order to provide for rotations through an angle of up to 90°, it is only necessary to consider half of the nodes within the checkered regions, that is the nodes bounded by the parallelogram with the dashed line at the top and the solid line at the bottom, and the nodes bounded by the parallelogram with the solid line at the top and the dotted line at the bottom. These nodes represent a quarter of the total number of nodes in the grid, that is ${N_\Theta}^2$ nodes.

In general, when we deem it sufficient to only provide for rotations through an angle of up to θ_r degrees (in a CW or CCW direction) with respect to a reference signature, the number of nodes that need to be considered, that is the number of DTW-based distances (between observations) that need to be calculated, is given by

$$\frac{\theta_r {N_\Theta}^2}{90}. \tag{4.2}$$

Non-linear method. The non-linear method uses a DTW algorithm to align observation sequences in a non-linear way. This algorithm is similar to the DTW algorithm of section 5.3, which is used for feature vector alignment, and implements the latter algorithm in an iterative way. Since we present the DTW algorithms used in this dissertation in chapter 5, we reserve a detailed discussion of this approach for section 5.4. Since this algorithm aligns observation sequences in a non-linear way, the "final observation sequence" (see figure 4.3 (b)) will always have a length greater than or equal to T (see

section 5.4). As is the case for the linear method, it is also possible to significantly reduce the computational requirements for the non-linear method, when one deems it sufficient to provide for only small rotational variations. We discuss these aspects in more detail in sections 5.4 and 9.3.

The HMM-based system developed in this dissertation is more efficient than the DTW-based system (see section 9.4) and achieves rotation invariance without observation sequence alignment. This is one of the reasons why the HMM-based system is more suited for commercial implementation.

4.4 Advantages of the discrete Radon transform

We conclude this chapter by emphasising some of the advantages of basing the feature extraction techniques used by the DTW-based and HMM-based systems developed in this dissertation on the calculation of the DRT of a signature image.

Although the DRT is not a shift invariant representation of a signature image, *shift and scale invariance* is ensured by the subsequent image processing.

Each signature is a static image and contains no dynamic information. Since the feature vectors are obtained by calculating projections at different angles, *simulated time-evolution* is created from one feature vector to the next, where the angle is the dynamic variable. This enables the HMM-based system to construct an HMM for each signature.

Since the DRT is calculated at angles that range from 0^{O} to 360^{O}, instead of 0^{O} to 180^{O}, both of the systems developed in this dissertation are able to represent each signature image with a set of feature vectors that is *rotation invariant*. These advantages are now discussed in more detail.

Shift invariance. Although the DRT is not a shift invariant representation of a signature image, shift invariance is ensured by the subsequent image processing. The zero-valued components of each projection are decimated (removed) and the corresponding feature vector is constructed from the remaining components only.

Rotation invariance. The DRT is calculated at angles that range from 0^{O} to 360^{O}. We explain in chapter 6 that the HMM-based system developed in this dissertation represents each observation sequence with an HMM of which the states are organised in a ring. This ensures that each set of feature vectors is rotation invariant. The periodic nature of the DRT also enables the DTW-based system to optimally align each initial observation sequence with the appropriate reference sequence, so that rotation invariance is guaranteed.

As an alternative to calculating the DRT, one can also consider the coordinates of the black pixels within a signature image to be a set of two-dimensional sample vectors and use the KL-transform (see section 3.4.2) to align these "stroke pixels" with the two principal axes. In this way rotation invariance is also achieved. Nel, Du Preez, and Herbst (2005) explain that this procedure is simple, but not reliable. Problems are encountered with signatures that do not have a clear "direction", that is where the signature is almost

round and the two principal values are approximately equal. These type of problems are not encountered when the DRT is calculated.

Scale invariance. For each projection, scale invariance has to be achieved in the direction perpendicular to the direction in which the signature image is scanned (see section 4.2), that is perpendicular to the beams, and in the direction parallel to the beams. Scale invariance perpendicular to the beams is ensured by shrinking or expanding each decimated projection to the required dimension. Scale invariance parallel to the beams is achieved by normalising the intensity of each feature vector. This is achieved by dividing each feature vector by the variance of the intensity of the entire set of feature vectors.

4.5 Concluding remarks

In this chapter we explained how the DRT can be used to extract a sequence of feature vectors from a raw signature image. The DRT is a robust feature extraction method and enables us to normalise each signature's representation (with respect to variations in scale, rotational orientation, and position) in a relatively simple way. The DRT also simulates time-evolution, from one feature vector (in an observation sequence) to the next, and therefore enables us to construct an HMM for each writer's signature.

We present the DTW-based and HMM-based approaches to signature modelling (that are used in this dissertation) in the next two chapters, that is chapter 5 and 6, respectively. The protocol for classifying an input (test) signature as authentic or fraudulent, is discussed in chapter 7.

Chapter 5

DTW-Based Signature Modelling

5.1 Introduction

In this chapter we focus on the DTW-based off-line signature verification system developed in this dissertation. In section 5.2 we give a brief overview of DTW. Although many DTW algorithms exist, we discuss only *two* algorithms that are especially suited for this work. In section 5.3 we discuss a *DTW-based feature vector alignment algorithm*. We illustrate this algorithm with a simple example. In section 5.4 we discuss a *DTW-based observation sequence alignment algorithm*. In section 5.5 we explain how a DTW-based signature model is constructed and trained. The DTW-based and HMM-based systems use similar verification protocols and a discussion of the verification strategy is reserved for chapter 7. The HMM-based approach to off-line signature verification is presented in chapter 6.

5.2 Overview

The primary objective of DTW is to non-linearly align one or more feature vectors (or observation sequences) before they are compared (matched). This alignment is often necessary in order to ensure that the appropriate features (or observations) of the respective feature vectors (or observation sequences) are compared.

Since DTW (like HMMs and SDCs) is a feature-based approach to pattern recognition, it is sometimes categorised as a statistical pattern recognition technique. However, unlike HMMs, DTW is not a "stochastic" approach to pattern recognition and simply involves the matching of a test pattern with one or more templates (reference patterns). It is therefore better categorised as a *template matching* technique. Other template matching techniques include object matching via correlation coefficients, etc.

Dynamic programming. DTW and HMMs are heavily based on *dynamic programming* techniques. Dynamic programming is a broad mathematical concept for analysing

processes involving optimal decisions. HMMs are based on a stochastic approach to pattern recognition and generalise DTW techniques. For those unfamiliar with dynamic programming, we present some key concepts in appendix A. A more comprehensive tutorial on dynamic programming can be found in a book by Deller, Proakis, and Hansen (1999).

5.3 DTW-based feature vector alignment

5.3.1 Background and motivation

We recall from chapter 4 that the bulk of the image processing and feature extraction (for both of the systems developed in this dissertation) involves the calculation of the DRT of each signature image. The DRT is obtained by calculating projections of each signature at different angles. Each of these projections are then subjected to some further processing, so that each of these processed projections represents a feature vector.

A *DTW-based feature vector alignment algorithm* is used to obtain a *DTW-based distance* between two feature vectors. The DTW-based system developed in this dissertation uses this distance to

- optimally align two observation sequences during the feature extraction process and
- calculate the distance between two signatures.

We recall from chapter 4 that, in order to ensure that each observation sequence is a rotation invariant representation of the corresponding signature image, observation sequence alignment is necessary for the DTW-based system. As we explained in section 4.3.2, the optimal alignment of two observation sequences can be achieved in a linear or a non-linear (DTW-based) way. We now briefly explain how both of these methods use DTW-based feature vector alignment.

The *linear method* iteratively shifts the observation sequences with respect to each other. During each iteration, the DTW-based distances between the corresponding observations (feature vectors) are calculated. The alignment is optimal when the average DTW-based distance between the corresponding observations is a minimum. The distance between two signatures is simply the average of the DTW-based distances between the optimally aligned feature vectors. We explained in section 4.3.2 that it is possible to achieve this alignment more efficiently.

The *nonlinear method* uses the DTW algorithm of section 5.4 to align two observation sequences in a non-linear way. This algorithm uses a grid (similar to the one in figure 4.6), where each node relates an observation (component) of a reference sequence with the corresponding observation (component) of a test sequence. For each node, DTW-based feature vector alignment is used to calculate the DTW-based distance between the related observations. The optimal path through the grid is then obtained. This path is optimal in the sense that, for all the legitimate paths through the grid, the average of the DTW-based

distances along the optimal path is a minimum. The distance between two signatures is the average of the DTW-based distances along the optimal path. The DTW-based system developed in this dissertation uses this distance to

- construct a model for each writer's signature (section 5.5) and
- classify an input (test) signature as authentic or fraudulent (chapter 7).

Now suppose that we want to match a test vector, $\mathbf{x}_{\text{Test}}$, with a reference vector, $\mathbf{x}_{\text{Ref}}$. Let $\mathbf{x}_{\text{Test}}(i)$ and $\mathbf{x}_{\text{Ref}}(i)$ denote the ith components (features) of the test and reference vectors respectively. The simplest way to match these vectors is to calculate the Euclidean distance between them, that is

$$D_{\text{Eucl}}(\mathbf{x}_{\text{Test}}, \mathbf{x}_{\text{Ref}}) = \sqrt{(\mathbf{x}_{\text{Test}} - \mathbf{x}_{\text{Ref}})'(\mathbf{x}_{\text{Test}} - \mathbf{x}_{\text{Ref}})} \quad (5.1)$$

$$= \sqrt{[\mathbf{x}_{\text{Test}}(1) - \mathbf{x}_{\text{Ref}}(1)]^2 + \cdots + [\mathbf{x}_{\text{Test}}(d) - \mathbf{x}_{\text{Ref}}(d)]^2}. \quad (5.2)$$

Note that when the Euclidean distance (5.2) is used, the *corresponding* features are compared. This is often not desirable, especially for the type of features that the systems developed in this dissertation utilise.

For these two systems, $\mathbf{x}_{\text{Test}}$ and $\mathbf{x}_{\text{Ref}}$ represent corresponding (processed) projections for a test and reference signature, respectively (see chapter 4). When these feature vectors are represented graphically (with the vector indices on the one axis and the feature values on the other), the graphs typically resemble mountains with "peaks" and "valleys". Two such graphs (which represent a reference and test vector, respectively) are shown in the bottom section and left section of figure 5.1. The optimal path, that maps the features of the respective vectors, is plotted on the grid in the top-right section of figure 5.1. The reference vector is therefore plotted horizontally on the system of axes below the grid, while the test vector is plotted vertically on the system of axes to the left of the grid. The algorithm that calculates this path is discussed in section 5.3.2. All the feature vectors utilised by the algorithms developed in this dissertation have the *same* dimension, d. Therefore, for the purposes of this discussion, $\mathbf{x}_{\text{Test}}$ and $\mathbf{x}_{\text{Ref}}$ will always have the same length (as indicated in figure 5.1). In general though, the dimensions of $\mathbf{x}_{\text{Test}}$ and $\mathbf{x}_{\text{Ref}}$ may differ (see figure A.1).

Even when the test and reference signatures are produced by the same writer, the signatures will invariably differ – this results in different projections.

Suppose, for example, that the test and reference graphs (vectors) have prominent peaks at more or less the same point. This is the case for the vectors in figure 5.1. Although these peaks occur at different points (that is at i_k and j_k), it is more sensible to compare the local maxima (that is $\mathbf{x}_{\text{Ref}}(i_k)$ and $\mathbf{x}_{\text{Test}}(j_k)$), than it is to compare the "altitudes" at the same point (that is $\mathbf{x}_{\text{Ref}}(i_k)$ and $\mathbf{x}_{\text{Test}}(i_k)$).

In order to achieve this, we first have to non-linearly align the components of the two feature vectors, before they are compared. There is a DTW algorithm that is especially suited for this purpose. This algorithm is discussed in the next section.

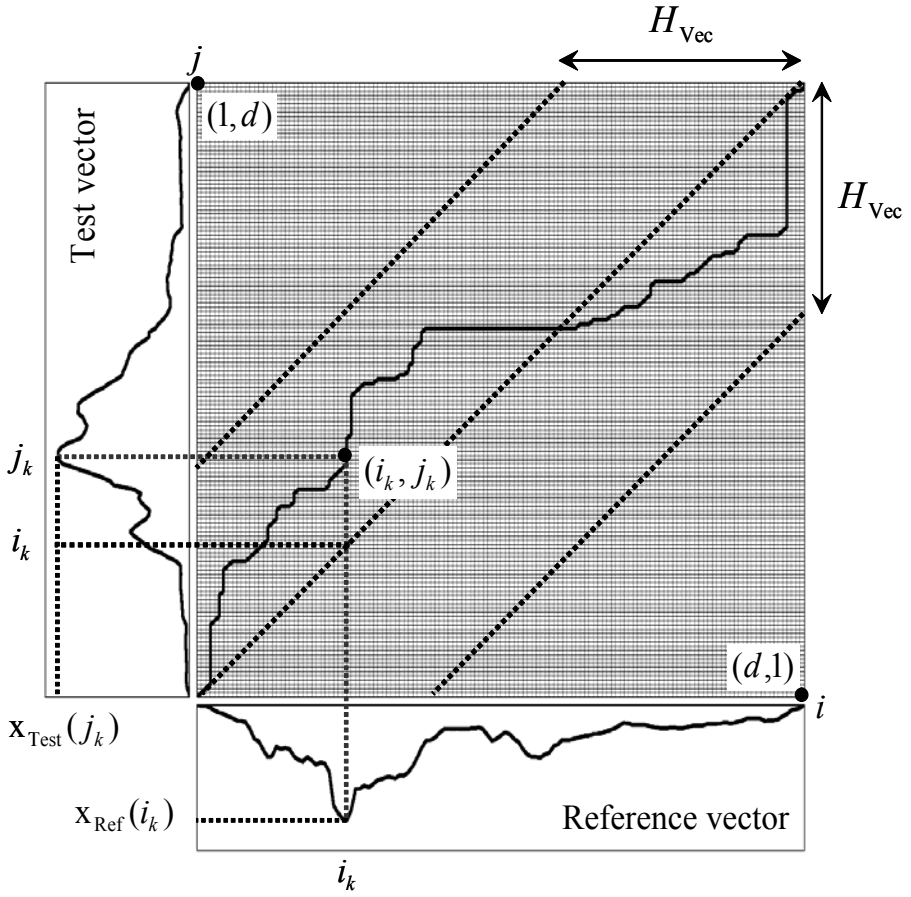


Figure 5.1: *Illustration of the DTW algorithm that is used to calculate the distance between two feature vectors. The reference vector (plotted horizontally on the system of axes below the grid) and the test vector (plotted vertically on the system of axes to the left of the grid) represent processed projections obtained via the DRT. The optimal path, that maps the features of the respective vectors, is plotted on the grid in the top-right section of the figure. All the feature vectors utilised by the algorithms developed in this dissertation have the same dimension, d.*

5.3.2 Algorithm

In order to match a test vector, $\mathbf{x}_{\text{Test}}$, and a reference vector, $\mathbf{x}_{\text{Ref}}$, in such a way that the distances between the prominent features are considered, we first construct a grid like the one in figure 5.1. Each node in the grid relates a specific component of the reference vector with a specific component of the test vector. Node (i, j), for example, relates the ith component of the reference vector, that is $\mathbf{x}_{\text{Ref}}(i)$, with the jth component of the test vector, that is $\mathbf{x}_{\text{Test}}(j)$.

For each node, the distance between the related test and reference components is calculated as follows,

$$D_{\text{Node}}(i, j) = \left(\mathbf{x}_{\text{Ref}}(i) - \mathbf{x}_{\text{Test}}(j)\right)^2 . \tag{5.3}$$

We therefore assign *node-based costs* to each path. Transition costs are not considered.

The difference between node-based costs, transition costs and combined costs is explained in appendix A.

The objective is to find a *complete path*

$$(i_0, j_0)(i_1, j_1)(i_2, j_2)\dots(i_{K-1}, j_{K-1})(i_K, j_K) \tag{5.4}$$

through the grid for which the *sum* of the node-based costs

$$D^{(\text{Compl})}_{\text{Node}}(\mathbf{x}_{\text{Test}}, \mathbf{x}_{\text{Ref}}) = \sum_{k=0}^{K} D_{\text{Node}}(i_k, j_k) \tag{5.5}$$

is a *minimum*, subject to the following constraints,

$$(i_0, j_0) = (1, 1),\ (i_K, j_K) = (d, d), \tag{5.6}$$
$$i_k \geq i_{k-1}, \quad \text{for} \quad k = 1, \dots, K, \tag{5.7}$$
$$j_k \geq j_{k-1}, \quad \text{for} \quad k = 1, \dots, K, \tag{5.8}$$
$$|j_k - i_k| \leq H_{\text{Vec}}, \quad \text{for} \quad k = 0, \dots, K, \tag{5.9}$$

where H_{Vec} is a constant that is less than or equal to d.

Constraint (5.6) ensures that the optimal path is a complete path. This implies that the first components (and the last components) of the respective vectors are always mapped onto each other. Note that the original node is (1,1), and that the terminal node is (d, d), where d is the dimension of the feature space.

Constraints (5.7) and (5.8) ensure that the optimal path is monotonically increasing. This implies that each node, (i, j), that forms part of a legitimate path, can only have one of three possible *preceding nodes*, that is $(i-1, j-1)$, $(i, j-1)$, or $(i-1, j)$.

Constraint (5.9) ensures that the optimal path does not deviate too much from the diagonal $i = j$. It does not make sense to relate features for which the vector indices differ to much. This also limits the computational cost. The bandwidth H_{Vec} is chosen by trail-and-error. This choice is often influenced by the dimension of the feature space, the computational requirements and the nature of the problem.

Let $D^{(\text{Part})}_{\text{Node}}(i, j)$ denote the distance associated with the *partial optimal path* that terminates at (i, j). The optimal preceding node for (i, j) is denoted by $\leftarrow (i, j)$. A preceding node is deemed optimal when the partial optimal path that passes through it and terminates at (i, j), is associated with a minimum cost.

We now discuss the DTW algorithm that calculates the *complete optimal path* and the distance (cost) associated with it.

Initialisation. The algorithm is initiated by setting

$$D^{(\text{Part})}_{\text{Node}}(1, 1) = D_{\text{Node}}(1, 1). \tag{5.10}$$

Recursion. The other legitimate nodes (that is the nodes that are within the allotted bandwidth around the diagonal) are then considered in a left-to-right, bottom-to-top fashion. For each of these nodes, $\leftarrow (i, j)$ and $D^{(\text{Part})}_{\text{Node}}(i, j)$, are calculated as follows,

$$\leftarrow (i, j) = \text{argmin}\left\{D^{(\text{Part})}_{\text{Node}}(i-1, j-1), D^{(\text{Part})}_{\text{Node}}(i, j-1), D^{(\text{Part})}_{\text{Node}}(i-1, j)\right\}, \tag{5.11}$$

$$D_{\text{Node}}^{(\text{Part})}(i,j) = D_{\text{Node}}(i,j) + D_{\text{Node}}^{(\text{Part})}(\leftarrow (i,j)). \tag{5.12}$$

Note that when $D_{\text{Node}}^{(\text{Part})}(\cdot,\cdot)$ is the same for two or three of the preceding nodes, the order of preference is as follows: $(i-1,j-1)$, then $(i,j-1)$, then $(i-1,j)$.

Backtracking. The complete optimal path is found through backtracking. The last node, (d,d), is connected with $\leftarrow (d,d)$, which is connected with $\leftarrow (\leftarrow (d,d))$, etc. The process is repeated until the second node on the complete optimal path, that is (i_2,j_2), is connected with the first node, that is $(1,1)$. The complete optimal path (thick line) is shown in figure 5.1.

Termination. The distance associated with the complete optimal path is simply the partial optimal path that terminates at (d,d), that is

$$D_{\text{Node}}^{(\text{Compl})}(\mathbf{x}_{\text{Test}}, \mathbf{x}_{\text{Ref}}) = D_{\text{Node}}^{(\text{Part})}(d,d). \tag{5.13}$$

The distance associated with the complete optimal path represents the distance between the warped (mapped) feature vectors. We refer to this distance as the *DTW-based distance* between the two vectors.

Note that the Euclidean distance ($H_{\text{Vec}} = 0$) is a special case for the DTW-based distance. In this case the complete optimal path will be on the diagonal, $i = j$.

5.3.3 Example

We now illustrate the principles discussed in the previous sections with a simple example. Suppose that we want to obtain the distance between the following two vectors,

$$\mathbf{x}_{\text{Test}} = [\,0 \quad 1 \quad 2 \quad 1 \quad 1 \quad 2 \quad 3 \quad 0\,], \tag{5.14}$$

$$\mathbf{x}_{\text{Ref}} = [\,0 \quad 1 \quad 2 \quad 3 \quad 2 \quad 1 \quad 2 \quad 0\,], \tag{5.15}$$

using the DTW algorithm of the previous section.

We first use (5.3) to calculate the distance between the related test and reference components, for example,

$$D_{\text{Node}}(4,2) = (\mathbf{x}_{\text{Ref}}(4) - \mathbf{x}_{\text{Test}}(2))^2 = (3-1)^2 = 4. \tag{5.16}$$

The distance associated with each node is shown in figure 5.2.

Note that we restrict the search by choosing $H_{\text{Vec}} = 4$. This implies that only 52 out of a possible 64 nodes are considered. The dimension of the feature space is $d = 8$.

We use (5.10) to initiate the algorithm,

$$D_{\text{Node}}^{(\text{Part})}(1,1) = D_{\text{Node}}(1,1) = (0-0)^2 = 0. \tag{5.17}$$

The other legitimate nodes are now considered in a left-to-right, bottom-to-top fashion. We therefore first consider node $(2,1)$. Note that the only possible preceding node for $(2,1)$ is the node to its left, that is $(1,1)$. Therefore, according to (5.11), we have that

$$\leftarrow (2,1) = (1,1). \tag{5.18}$$

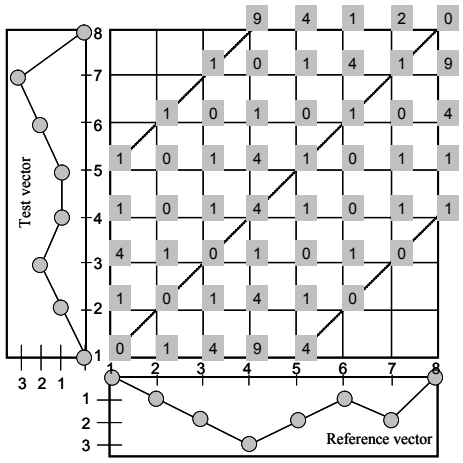


Figure 5.2: *Grid with distances associated with each node.*

According to (5.12) the distance associated with the *partial optimal path* that terminates at $(2, 1)$ is

$$D^{(\text{Part})}_{\text{Node}}(2,1) = D_{\text{Node}}(2,1) + D^{(\text{Part})}_{\text{Node}}(\leftarrow (2,1)) = 1 + 0 = 1. \tag{5.19}$$

This process is now repeated for the other nodes. The distances associated with the partial optimal paths that terminate at each node are shown in figure 5.3. The preceding nodes are indicated with arrows.

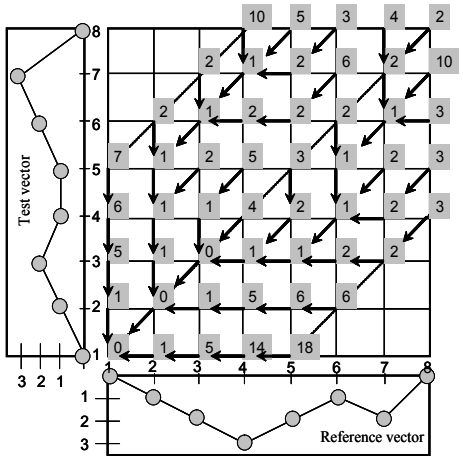


Figure 5.3: *Distances associated with the partial optimal paths that terminate at each node. The preceding nodes are indicated with arrows.*

Take $(6, 7)$ for example. The distances associated with the partial optimal paths that terminate at its three preceding nodes are as follows

$$D^{(\text{Part})}_{\text{Node}}(5,6) = D^{(\text{Part})}_{\text{Node}}(6,6) = D^{(\text{Part})}_{\text{Node}}(5,7) = 2. \tag{5.20}$$

Since the diagonal node has precedence, we have that

$$\leftarrow (6,7) = (5,6), \tag{5.21}$$

$$D_{\text{Node}}^{(\text{Part})}(6,7) = D_{\text{Node}}(6,7) + D_{\text{Node}}^{(\text{Part})}(\leftarrow (6,7)) = 2 + 4 = 6. \tag{5.22}$$

The complete optimal path is found through backtracking,

$$\leftarrow (8,8) = (7,7),\ \leftarrow (7,7) = (7,6),\ \leftarrow (7,6) = (6,5), \ldots, \leftarrow (2,2) = (1,1). \tag{5.23}$$

The complete optimal path is therefore given by

$$(1,1),(2,2),(3,3),(4,3),(5,3),(6,4),(6,5),(7,6),(7,7),(8,8), \tag{5.24}$$

which implies that $K = 9$, where K is the number of transitions in the optimal path.

According to (5.13) the distance associated with the complete optimal path is

$$D_{\text{Node}}^{(\text{Compl})}(\mathbf{x}_{\text{Test}}, \mathbf{x}_{\text{Ref}}) = D_{\text{Node}}^{(\text{Part})}(8,8) = 2. \tag{5.25}$$

Note that, when the bandwidth is restricted to $H_{\text{Vec}} = 0$, the optimal path is on the diagonal, $i = j$. In this case, the DTW-based distance is simply the Euclidean distance between the two vectors, which is 7. Figure 5.4 shows the complete optimal path (thick line) for this example.

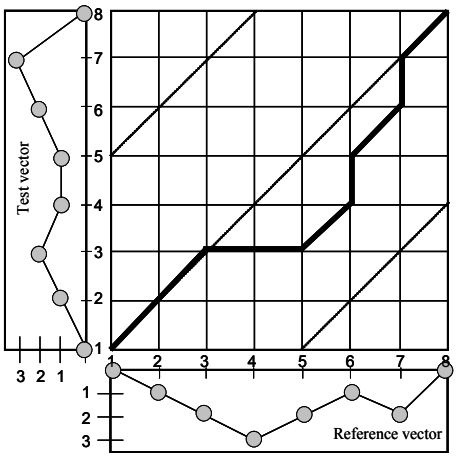


Figure 5.4*: Complete optimal path.*

5.4 DTW-based observation sequence alignment

In this section we discuss a DTW algorithm that non-linearly aligns two *observation sequences* and also calculates the DTW-based distance between them. This algorithm is very similar to the DTW-based feature vector alignment algorithm discussed in section 5.3.2 and uses the latter algorithm in an iterative way.

A grid similar to the one in figure 5.1 is constructed, except that the test and reference vectors are replaced by test and reference observation sequences, respectively. Each node in the grid therefore relates an observation (component) of a reference sequence

with the corresponding observation (component) of a test sequence. Node (i, j), for example, relates the ith observation of the reference sequence, that is $\mathbf{X}_{\text{Ref}}(i)$, with the jth observation of the test sequence, that is $\mathbf{X}_{\text{Test}}(j)$.

For each node, DTW-based feature vector alignment (section 5.3.2) is used to calculate the DTW-based distance between the related test and reference vectors,

$$D_{\text{NODE}}(i, j) = D_{\text{Node}}^{(\text{Compl})}(\mathbf{X}_{\text{Ref}}(i), \mathbf{X}_{\text{Test}}(j)), \tag{5.26}$$

where $D_{\text{Node}}^{(\text{Compl})}(\cdot, \cdot)$ is given by (5.5). Note that we use the notation $D_{\text{NODE}}(\cdot, \cdot)$ in (5.26), in order to prevent confusion with the notation $D_{\text{Node}}(\cdot, \cdot)$ of (5.3).

The objective is to find a path (see equation 5.4)) through the grid for which the *average* of the node-based costs

$$D(\mathbf{X}_{\text{Test}}, \mathbf{X}_{\text{Ref}}) = \frac{1}{K+1} \sum_{k=0}^{K} D_{\text{NODE}}(i_k, j_k) \tag{5.27}$$

is a minimum, subject to the following constraints,

$$(i_0, j_0) = (1, k) \text{ or } (i_0, j_0) = (k, 1), \text{ for } k = 1, \ldots, T, \tag{5.28}$$
$$(i_K, j_K) = (T, k) \text{ or } (i_K, j_K) = (k, T), \text{ for } k = 1, \ldots, T, \tag{5.29}$$
$$i_k \geq i_{k-1}, \text{ for } k = 1, \ldots, K, \tag{5.30}$$
$$j_k \geq j_{k-1}, \text{ for } k = 1, \ldots, K, \tag{5.31}$$
$$|j_k - i_k| \leq H_{\text{Seq}}, \text{ for } k = 0, \ldots, K, \tag{5.32}$$

where the bandwidth H_{Seq} is less than or equal to T.

Note that the optimal path is not necessarily a complete path, but constraints (5.28) and (5.29) require that one or both of the coordinates of the initial node have to be unity, and that one or both of the coordinates of the terminating node have to be equal to T, where $T = 2N_\Theta$ is the length of an observation sequence. The other constraints are similar to those used for feature vector alignment (section 5.3). The algorithm for finding the optimal path through the grid is also equivalent to the algorithm of section 5.3.2.

One can provide for *any* possible rotation, that is rotations of up to 180° (in a CW or CCW direction) with respect to a reference signature, by restricting the search to a bandwidth of $H_{\text{Seq}}^{180^\circ} = N_\Theta$.

We illustrate this in figure 5.5. When the test sequence (on the vertical axis, labeled 0°) and the reference sequence (on the horizontal axis) are identical, the optimal path (solid line) lies on the diagonal. The observation sequences are represented by rectangular blocks. The two shades of gray provide a position of reference on each sequence. One half of the observations are represented by a dark-gray block, while the other half are represented by a light-gray block. We recall from section 4.2, that the observations represented by the light-gray block are simply reflections of the observations represented by the dark-gray block.

Suppose that a test signature is rotated through an angle of 180° (in a CW or CCW direction) with respect to a reference signature. This is the worst-case scenario. Since

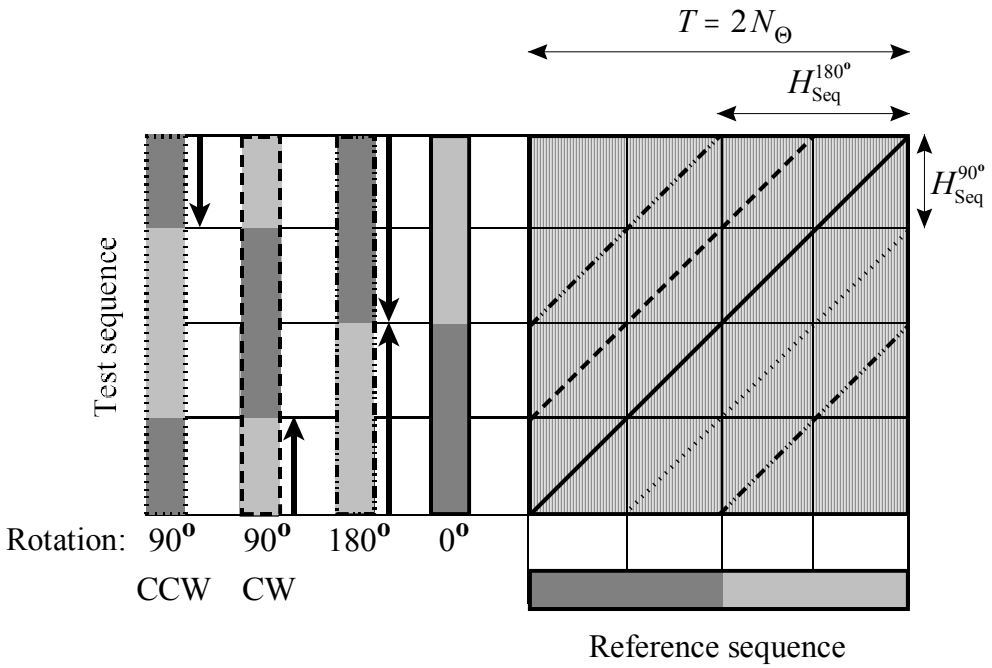


Figure 5.5*: Non-linear observation sequence alignment. Illustration of the DTW algorithm that is used to calculate the distance between two observation sequences. One half of the observations is represented by a dark-gray block, while the other half is represented by a light-gray block. The observations represented by the light-gray block are reflections of the observations represented by the dark-gray block. In order to provide for any possible rotation, it is only necessary to consider the nodes between the dash-double-dotted lines. In order to provide for rotations through an angle of up to* 90°*, it is only necessary to consider the nodes between the dashed line and the dotted line.*

an observation sequence is obtained via the calculation of the DRT, a rotation through an angle of 180° is equivalent to a shift of N_Θ observations. Since the DRT has a period of 360° ($2N_\Theta$ observations), a shift of N_Θ observations (in any direction) will result in the sequence labeled 180°. The arrows indicate the possible shifts. The dash-double-dotted lines represent two possible optimal paths for this scenario. Note that, since the observations represented by the light-gray block are reflections of the observations represented by the dark-gray block, only those observations aligned along *one* of the two dash-double-dotted lines need to be compared. It is therefore only necessary to consider the nodes between these lines. With the above restriction, the non-linear (DTW-based) approach requires on the order of 3/2 times more floating point operations (flops) than the efficient linear method of section 4.3.2.

Should one deem it sufficient to provide for only small rotational variations – this is also the case for the linear method of section 4.3.2 – it is possible to further reduce the computational requirements. One may, for example, only provide for rotations through an angle of up to 90° (in a CW or CCW direction) with respect to a reference signature, by restricting the search to a bandwidth of $H_{\text{Seq}}^{90^\circ} = N_\Theta/2$.

We again use figure 5.5 to illustrate this. Suppose that a test signature is rotated through

an angle of 90° (in a CW or CCW direction) with respect to a reference signature, and that this is the maximum rotation we want to provide for. A rotation through an angle of 90° is equivalent to a shift of $N_\Theta/2$ observations. Let us assume that a rotation of 90° in a CW direction results in a shift of $N_\Theta/2$ observations in an upwards direction (the sequence labeled 90° CW), and that a rotation of 90° in a CCW direction results in a shift of $N_\Theta/2$ observations in a downwards direction (the sequence labeled 90° CCW). The arrows indicate these shifts. The direction of these shifts (upwards or downwards) depends on whether the angle is incremented in a CW or CCW direction, when a DRT is calculated.

The optimal paths for these two scenarios are represented by the dashed and dotted lines, respectively. Note that, since the observations represented by the light-gray block are reflections of the observations represented by the dark-gray block, *only* those observations aligned along the dashed line (dark-gray observations and half of the light-gray observations), or *only* those observations aligned along the dotted line (light-gray observations and half of the dark-gray observations), are compared. It is therefore only necessary to consider the nodes between these lines. With the above restriction, the non-linear (DTW-based) approach requires on the order of 7/4 times more flops than the efficient linear method of section 4.3.2.

In general, when we deem it sufficient to only provide for rotations through an angle of up to $\theta_r \in (0°, 180°]$ degrees (in a CW or CCW direction) with respect to a reference signature, we should restrict the search to a bandwidth of

$$H_{\text{Seq}}^{\theta_r} = \frac{N_\Theta\, \theta_r}{180°}. \tag{5.33}$$

We give a detailed analysis of the computational requirements for the DTW-based and HMM-based systems developed in this dissertation in chapter 9.

5.5 Signature modelling

We explained in section 4.3.1 how the DTW-based system developed in this dissertation extracts an initial observation sequence from a raw signature image. The technique described in section 4.3.2 is then used to select a single reference sequence from each writer's training set. Each initial observation sequence is then optimally aligned with the corresponding writer's reference sequence, so that rotation invariance is guaranteed. The selected reference sequence acts as a template for the writer's signature. When we assume that there are Ω writers in the database, there are Ω corresponding reference sequences, that is

$$\{\mathbf{X}_{\text{Ref}}^{(1)}, \mathbf{X}_{\text{Ref}}^{(2)}, \ldots, \mathbf{X}_{\text{Ref}}^{(\Omega)}\}. \tag{5.34}$$

Assume that for each writer, the individual observations for all the training sequences have *already* been optimally aligned with the observations for the corresponding reference (template) sequence. This alignment is part of the feature extraction process and can be achieved through the linear method of section 4.3.2, or the non-linear (DTW-based) method of section 5.4.

When we take the linear approach, the distance, D, between the ith training sequence for writer ω (denoted by $\mathbf{X}_i^{(\omega)}$) and the reference sequence (denoted by $\mathbf{X}_{\text{Ref}}^{(\omega)}$), is the *average* of the DTW-based distances between the *aligned* observations. The distance between two individual observations is the *sum* of the node-based costs along the complete optimal path. This implies that

$$D(\mathbf{X}_i^{(\omega)}, \mathbf{X}_{\text{Ref}}^{(\omega)}) = \frac{1}{T} \sum_{k=1}^{T} D_{\text{Node}}^{(\text{Compl})}(\mathbf{X}_i^{(\omega)}(k), \mathbf{X}_{\text{Ref}}^{(\omega)}(k)), \tag{5.35}$$

where $\mathbf{X}(k)$ represents the kth observation of sequence $\mathbf{X}$ and $D_{\text{Node}}^{(\text{Compl})}(\cdot,\cdot)$ is given by (5.5).

When we take the non-linear (DTW-based) approach, equations (5.26) and (5.27) are used to calculate the distance, D, between the two observation sequences, so that

$$D(\mathbf{X}_i^{(\omega)}, \mathbf{X}_{\text{Ref}}^{(\omega)}) = \frac{1}{K+1} \sum_{k=0}^{K} D_{\text{Node}}^{(\text{Compl})}(\mathbf{X}_i^{(\omega)}(i_k), \mathbf{X}_{\text{Ref}}^{(\omega)}(j_k)), \tag{5.36}$$

where (i_k, j_k) is the optimal path.

We now define the following statistics for the signature of writer ω,

$$\mu_\omega = \frac{1}{N_\omega - 1} \sum_{\substack{i=1,\\ i \neq \text{Ref}}}^{N_\omega} D(\mathbf{X}_i^{(\omega)}, \mathbf{X}_{\text{Ref}}^{(\omega)}), \tag{5.37}$$

$$\sigma_\omega^2 = \frac{1}{N_\omega - 2} \sum_{\substack{i=1,\\ i \neq \text{Ref}}}^{N_\omega} (D(\mathbf{X}_i^{(\omega)}, \mathbf{X}_{\text{Ref}}^{(\omega)})) - \mu_\omega)^2, \tag{5.38}$$

where N_ω is the number of samples in the training set.

These statistics ((5.37) and (5.38)) are used to obtain a *threshold distance*, which is subsequently used to authenticate an input (test) signature (section 7.3). The mean provides a reference distance. The standard deviation σ_ω (5.38) measures the *variability* of the writer's signature. Note that the mean μ_ω (5.37) represents the average DTW-based distance between the individual training sequences and the corresponding reference sequence, which implies that the mean μ_ω (5.37), in conjunction with providing a reference distance, *also* measures the variability of the writer's signature. We discuss these aspects in more detail in chapter 7.

The *trained model* for the signature of writer ω therefore consists of a reference (template) sequence $\mathbf{X}_{\text{Ref}}^{(\omega)}$, in conjunction with one or both of the statistics defined in (5.37) and (5.38).

5.6 Concluding remarks

In this chapter we explained how the DTW-based system developed in this dissertation constructs and trains a template-based model for a specific writer's signature. Although

this is a relatively simple model, the results for this DTW-based system compare well with the results for the (more sophisticated) HMM-based system developed in this dissertation (see section 8.4). However, the computational cost (for verifying a test signature) is significantly higher when the DTW-based system is used. This is especially true when we allow for the possibility that a test signature is rotated through an angle of up to 180° (in a CW or CCW direction) with respect to a reference signature (see section 9.4).

The HMM-based system developed in this dissertation is discussed in the next chapter, while the protocol for classifying an input (test) signature as authentic or fraudulent, is discussed in chapter 7.

Chapter 6

Signature Modelling Using HMMs

6.1 Introduction

In this chapter we focus on the HMM-based off-line signature verification system developed in this dissertation. We give a brief overview of HMMs in section 6.2 and present the notation in section 6.3. For those readers that are not familiar with HMMs, we give an introduction to key HMM concepts in appendix B. More comprehensive tutorials on HMMs can be found in a paper by Rabiner (1989) and the book by Deller, Proakis, and Hansen (1999). A précis of the HMM-based system developed in this dissertation can be found in a paper by Coetzer, Herbst, and Du Preez (2004).

The HMM-based system developed in this dissertation uses a continuous, first order HMM to represent each writer's signature. We discuss the very specific structure (topology) of these HMMs in section 6.4.1 and explain how they are initialised and trained in sections 6.4.2 and 6.4.3 respectively. The HMM-based and DTW-based systems use similar verification protocols and a discussion of the verification strategy is reserved for chapter 7.

6.2 Overview

As we explained in chapter 3, an SDC models each entity (pattern) with a single feature vector, that is a single observation. This approach is often sufficient for simple recognition problems. However, more complex problems often demand the use of a more sophisticated classifier, that is a classifier that models each entity (pattern) with a *sequence* of observations.

Unlike DTW, HMMs are based on a "stochastic" approach to pattern recognition and generalise the DTW techniques discussed in chapter 5. A pattern recognition system, which is based on HMMs, typically uses an HMM to represent each pattern class. Each of these HMMs is used to model an observation sequence, as well as the relationship

between the individual observations. HMMs are therefore constructed in such a way that time-evolution is assumed from one observation in the sequence to the next.

Since speech signals and dynamic (on-line) signatures also contain temporal information, it is possible to extract a continuous observation sequence from these signals in a very intuitive way. For this reason HMMs are especially well-suited for modelling these type of signals. This is not the case for static (off-line) signatures. Consequently, feature vectors have to be extracted from off-line signatures in such a way that time-evolution is *simulated* from one observation to the next.

In chapter 2 we discussed two recent HMM-based off-line signature verification systems which both achieve time-evolution by using a superimposed grid on a signature image. El-Yacoubi et al. (2000) use such a grid to segment a signature image into local square cells. From each cell, the pixel density is computed, so that each pixel density represents a local feature. Each signature image is therefore represented by a sequence of feature vectors, where each feature vector represents the pixel densities associated with a column of cells. Justino et al. (2001) also use a grid segmentation scheme to extract three features, that is a pixel density feature, a pixel distribution feature (extended-shadow-code) and an axial slant feature.

The HMM-based system developed in this dissertation simulates time-evolution from one observation to the next by calculating the DRT of each signature image during the feature extraction process (section 4.2). Before we discuss the HMM-based signature model (section 6.4), we first present the notation in the following section.

6.3 Notation

We use the following notation for a sequence of T continuous observations,

$$\mathbf{X}_1^T = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_T\}, \tag{6.1}$$

where $\mathbf{x}_i,\ i = 1, 2, \ldots T$ denotes the ith feature vector in the sequence.

We use the following notation for a continuous, first order HMM λ:

- We denote the N individual states as

$$\mathbf{S} = \{s_1, s_2, \ldots, s_N\}, \tag{6.2}$$

  and the state at time t as q_t.

- The initial state distribution is denoted by $\boldsymbol{\pi} = \{\pi_i\}$, where

$$\pi_i = P(q_1 = s_i),\ i = 1, \ldots N. \tag{6.3}$$

- The state transition probability distribution is denoted by $\mathbf{A} = \{a_{i,j}\}$, where

$$a_{i,j} = P(q_{t+1} = s_j | q_t = s_i),\ i = 1, \ldots N,\ j = 1, \ldots, N. \tag{6.4}$$

- The PDF, which quantifies the similarity between a feature vector $\mathbf{x}$ and the state s_j, is denoted by
$$f(\mathbf{x}|s_j, \lambda),\ j = 1, \ldots, N. \tag{6.5}$$
- The similarity between an observation sequence $\mathbf{X}$ and a model λ is denoted by
$$f(\mathbf{X}|\lambda). \tag{6.6}$$

6.4 Signature modelling

The HMM-based system developed in this dissertation simulates time-evolution from one observation in an observation sequence to the next by calculating the DRT of a raw signature image (section 4.2). The feature vectors are therefore obtained by calculating projections of a signature at different angles, after which they are subjected to some further processing. The angle is therefore the dynamic variable. This enables us to construct an HMM for each signature.

6.4.1 HMM topology

The HMM-based system developed in this dissertation represents each writer's signature with an HMM of which the states are organised in a *ring* (see figure 6.1). This model is equivalent to the popular left-to-right model (see appendix B, figure B.10), but a transition from the last state to the first state is allowed. Since the HMM is constructed in such a way that it is equally likely to enter the model at any state, and the feature vectors are obtained from all the projections, that is, the projections calculated at angles ranging from 0^{o} to 360^{o}, the ring topology of the HMM guarantees that the signatures are rotation invariant. Each state in the HMM represents one or more feature vectors that occupy similar positions in a d-dimensional feature space. This implies that the HMM groups certain projections (columns of the DRT) together. It is important to note that this segmentation process only takes place after some *further* image processing (see section 4.3.1) has been conducted on the original projections.

The state transition probabilities and the parameters for the PDFs, that represent the individual states, are estimated during training.

Note that, when a test sequence is matched with a trained ring-structured HMM, it is possible that the HMM is exited *before* the entire ring has been traversed. In other words, it is possible that the HMM is exited before the state immediately preceding the emitting state, which was initially visited, is encountered. See section B.4 for the definitions of emitting and non-emitting states. It is also possible that the ring is traversed more than once.

In order to ensure that the *entire* ring is traversed, and only once, one may for example use N HMMs, where each HMM has N states and left-to-right topology. Figure 6.2 illustrates this configuration for $N = 6$. Note that the first HMM, that is λ_1, does not have a transition between s_6 (the last emitting state) and s_1 (the first emitting state).

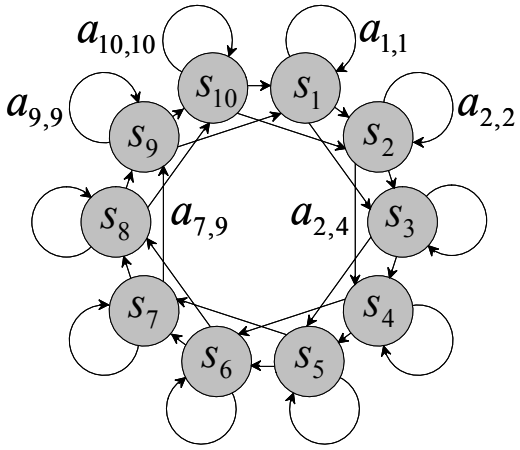


Figure 6.1: *An example of an HMM with a ring topology. This model has ten states with one state skip. One state skip is equivalent to two allotted forward links.*

The second HMM, that is λ_2, does not have a transition between s_1 (the first emitting state) and s_2 (the second emitting state). Corresponding states within these HMMs, for example s_2 in λ_1 and s_2 in λ_2, share a common probability density function. The initial and terminal (non-emitting) states are denoted by s_0 and s_7, respectively. The probability to make a transition from the initial (non-emitting) state to s_1 in λ_1, or to s_2 in λ_2, or to s_3 in λ_3, etc., is the same, that is $\pi_1 = \pi_2 = \pi_3 = \ldots = \pi_6 = 1/6$. When a transition is made from the initial (non-emitting) state to s_1 in λ_1, it is only possible to reach the terminal (non-emitting) state from s_6. Similarly, when a transition is made from the initial (non-emitting) state to s_2 in λ_2, it is only possible to reach the terminal (non-emitting) state from s_1, etc. In this way it is still equally likely to enter the model at any state, but it is guaranteed that the entire ring will be traversed.

However, when the above model is used with N HMMs and N states per HMM, we have N^2 states and $3N^2$ transitions, in contrast to the original N states and $3N$ transitions. This enlarges the computational requirements considerably. We therefore did not implement this model. The HMM-based system developed in this dissertation utilises a single ring-structured HMM, like the one shown in figure 6.1.

6.4.2 Initial estimates

The HMM-based system developed in this dissertation uses *uniform estimates* for the initial state and state transition probabilities. The PDFs, which represent the individual states, are estimated by first assigning an *equal* number of observations to each state. The average of the observations within each state is then calculated.

6.4.3 Training

Each model is trained using the Viterbi reestimation technique. The dissimilarity between an observation sequence $\mathbf{X}$ and a model λ can therefore be calculated as follows (see

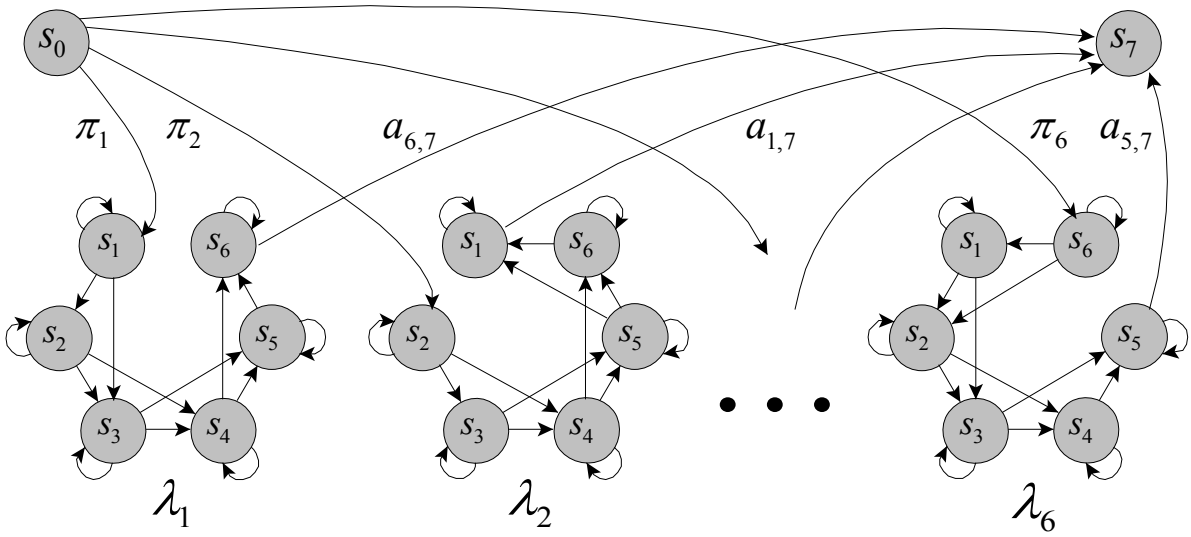


Figure 6.2*: An example of a model that consists of six individual HMMs. Each of these individual HMMs has six states with one state skip and a left-to-right topology. The initial (non-emitting) state is denoted by s_0 and the terminal (non-emitting) state by s_7. With this configuration it is still equally likely to enter the model at any state, but it is guaranteed that the entire ring will be traversed.*

Rabiner (1989)),

$$D(\mathbf{X}, \lambda) = -\ln(f(\mathbf{X}|\lambda)). \tag{6.7}$$

In real-world scenarios, each writer can only submit a small number of training samples when he or she is enrolled into the system. Since the algorithm uses feature vectors with a high dimension, the estimated covariance matrix of the PDF for each state is not reliable and may even be singular. A Mahalanobis distance measure can therefore not be found. Consequently these covariance matrices are not estimated and are initially set to $0.5I$, where I is the identity matrix. Only the mean vectors are estimated, which implies that the dissimilarity values are based on an Euclidean distance measure.

We assume that training signatures, genuine test signatures and forgeries are available for only a limited number of writers, that is for those writers in the database used. No forgeries are used in the training process, since the system aims to detect only skilled and casual forgeries − we give the reasons for this in section 8.7 − and these type of forgeries were not available when the system was implemented. The genuine test signatures and the forgeries are used to determine the error rates for the HMM-based system (chapter 8). Assuming that there are Ω writers in the database, the training signatures for each writer are used to construct an HMM, resulting in Ω models, that is

$$\{\lambda_1, \lambda_2, \ldots, \lambda_\Omega\}. \tag{6.8}$$

When the training set for writer ω is denoted by

$$\{\mathbf{X}_1^{(\omega)}, \mathbf{X}_2^{(\omega)}, \ldots, \mathbf{X}_{N_\omega}^{(\omega)}\}, \tag{6.9}$$

where N_ω is the number of samples in the training set, the dissimilarity between every training sample and the model is used to determine the following statistics for the writer's signature,

$$\mu_\omega = \frac{1}{N_\omega} \sum_{i=1}^{N_\omega} D(\mathbf{X}_i^{(\omega)}, \lambda_\omega), \tag{6.10}$$

$$\sigma_\omega^2 = \frac{1}{N_\omega - 1} \sum_{i=1}^{N_\omega} (D(\mathbf{X}_i^{(\omega)}, \lambda_\omega) - \mu_\omega)^2. \tag{6.11}$$

These statistics ((6.10) and (6.11)) are used to obtain a *threshold distance*, which is subsequently used to authenticate an input (test) signature (section 7.4). The mean provides a reference distance. The standard deviation σ_ω (6.11) measures the *variability* of the writer's signature. Note that the mean μ_ω (6.10) represents the average dissimilarity between the observation sequences in the training set and the HMM, λ_ω. Also note that λ_ω was trained with these observation sequences. This implies that the mean μ_ω (6.10), in conjunction with providing a reference distance, *also* measures the variability of the writer's signature. We discuss these aspects in more detail in the next chapter.

The *trained model* for the signature of writer ω therefore consists of the trained HMM, λ_ω, in conjunction with one or both of the statistics defined in (6.10) and (6.11).

6.5 Concluding remarks

In this chapter we explained how the HMM-based system developed in this dissertation constructs and trains an HMM for a specific writer's signature. The HMM is constructed in such a way that it constitutes a rotation invariant representation of the signature. In contrast to the DTW-based system developed in this dissertation, no observation sequence alignment is therefore necessary. This results in a lower computational cost (for verifying a test signature). The HMM-based system is therefore a more viable option for commercial implementation (see section 9.4).

The protocol for classifying an input (test) signature as authentic or fraudulent, is discussed in the next chapter.

Chapter 7

Verification

7.1 Introduction

In this chapter we explain how the systems developed in this dissertation classify an input (test) signature as authentic or fraudulent. We give a brief overview of this process in section 7.2. Although the DTW-based and HMM-based systems developed in this dissertation use different signature models (chapters 5 and 6), their verification protocols are very similar. We discuss the verification protocols for the DTW-based and HMM-based systems in sections 7.3 and 7.4, respectively.

7.2 Overview

When a system aims to detect only random forgeries, subsets of other writers' training sets can be used to model "typical" forgeries. This is called "impostor validation" and can be achieved through strategies like test normalisation (Auckenthaler et al. (2000)). These techniques enable one to construct verifiers that detect random forgeries very accurately (El-Yacoubi et al. (2000); Sabourin et al. (1997c)).

Since the systems developed in this dissertation aim to detect only skilled and casual forgeries, and since models for these forgeries are generally unobtainable, we are not able to utilise any of the above-mentioned impostor validation techniques. We also do not use any subset of genuine signatures for validation purposes.

When a claim is made that a test pattern was produced by a specific writer, the systems first match this test pattern with a trained model of the writer's signature. We presented these models in chapters 5 and 6. Statistics for the claimed writer's signature are used to obtain a threshold value. The dissimilarity between the test pattern and the model is then calculated. When this dissimilarity value is greater than the threshold, the test pattern is rejected as fraudulent, otherwise it is accepted as authentic.

7.3 DTW-based signature verification

We first discuss the verification strategy employed by the DTW-based system developed in this dissertation. When a claim is made that a test pattern (denoted by $\mathbf{X}_{\text{Test}}^{(\omega)}$) belongs to writer ω, the test pattern is first matched with the reference (template) pattern for writer ω (denoted by $\mathbf{X}_{\text{Ref}}^{(\omega)}$), so that a distance $D(\mathbf{X}_{\text{Test}}^{(\omega)}, \mathbf{X}_{\text{Ref}}^{(\omega)})$ is obtained.

When the *linear* approach of section 4.3.2 is used to align the observation sequences, the above distance is given by

$$D(\mathbf{X}_{\text{Test}}^{(\omega)}, \mathbf{X}_{\text{Ref}}^{(\omega)}) = \frac{1}{T} \sum_{k=1}^{T} D_{\text{Node}}^{(\text{Compl})}(\mathbf{X}_{\text{Test}}^{(\omega)}(k), \mathbf{X}_{\text{Ref}}^{(\omega)}(k)), \tag{7.1}$$

where $\mathbf{X}(k)$ represents the kth observation of sequence $\mathbf{X}$ and $D_{\text{Node}}^{(\text{Compl})}(\cdot, \cdot)$ is given by (5.5).

When the *non-linear* (DTW-based) approach of section 5.4 is used, the distance is given by

$$D(\mathbf{X}_{\text{Test}}^{(\omega)}, \mathbf{X}_{\text{Ref}}^{(\omega)}) = \frac{1}{K+1} \sum_{k=0}^{K} D_{\text{Node}}^{(\text{Compl})}(\mathbf{X}_{\text{Test}}^{(\omega)}(i_k), \mathbf{X}_{\text{Ref}}^{(\omega)}(j_k)), \tag{7.2}$$

where (i_k, j_k) is the optimal path.

We now normalise $D(\mathbf{X}_{\text{Test}}^{(\omega)}, \mathbf{X}_{\text{Ref}}^{(\omega)})$ in order to obtain a *global* threshold for *all* writers. Since equations (7.1) and (7.2) are based on the *Euclidean* distance between aligned feature vectors, we use the *mean* distance (5.37) to normalise $D(\mathbf{X}_{\text{Test}}^{(\omega)}, \mathbf{X}_{\text{Ref}}^{(\omega)})$ in the following way,

$$D_{\text{Eucl}}(\mathbf{X}_{\text{Test}}^{(\omega)}, \mathbf{X}_{\text{Ref}}^{(\omega)}) = \frac{D(\mathbf{X}_{\text{Test}}^{(\omega)}, \mathbf{X}_{\text{Ref}}^{(\omega)}) - \mu_\omega}{\mu_\omega}. \tag{7.3}$$

We justify this normalisation strategy in the next section.

It is important to note that the mean distance μ_ω (5.37), not only provides a *reference* distance, but *also* measures the *variability* of the writer's signature. The latter statement is valid, since the more the training signatures vary, the larger the average DTW-based distance between the training sequences and the reference (template) sequence will be. This results in a larger μ_ω.

A *sliding* threshold τ, where $\tau \in [0, \infty)$, is used to determine the error rates (see section 8.4.1) for the test patterns . When $D_{\text{Eucl}}(\mathbf{X}_{\text{Test}}^{(\omega)}, \mathbf{X}_{\text{Ref}}^{(\omega)}) < \tau$, that is when

$$D(\mathbf{X}_{\text{Test}}^{(\omega)}, \mathbf{X}_{\text{Ref}}^{(\omega)}) < \mu_\omega(1 + \tau), \tag{7.4}$$

the claim is accepted, otherwise the claim is rejected. When $\tau = 0$, all the test patterns for which $D(\mathbf{X}_{\text{Test}}^{(\omega)}, \mathbf{X}_{\text{Ref}}^{(\omega)}) \geq \mu_\omega$ are rejected. This almost always results in an FRR of 100% and an FAR of 0%. When $\tau \to \infty$, all the test patterns for which $D(\mathbf{X}_{\text{Test}}^{(\omega)}, \mathbf{X}_{\text{Ref}}^{(\omega)})$ is finite, are accepted. This always results in an FRR of 0% and an FAR of 100%.

7.4 HMM-based signature verification

We now discuss the verification strategy employed by the HMM-based system developed in this dissertation. When a claim is made that a test pattern $\mathbf{X}_{\text{Test}}^{(\omega)}$ belongs to writer ω, the pattern is first matched with the HMM λ_ω through Viterbi alignment. This match is quantified by $f(\mathbf{X}_{\text{Test}}^{(\omega)}|\lambda_\omega)$. The dissimilarity between the test pattern and the model is then calculated as follows (see Rabiner (1989)),

$$D(\mathbf{X}_{\text{Test}}^{(\omega)}, \lambda_\omega) = -\ln(f(\mathbf{X}_{\text{Test}}^{(\omega)}|\lambda_\omega)). \tag{7.5}$$

In order to use a *global* threshold for all writers, Dolfing (1998) suggests that every dissimilarity value in (7.5) is normalised, using the statistics of the claimed writer's signature, that is (6.10) and (6.11),

$$D_{\text{Mah}}(\mathbf{X}_{\text{Test}}^{(\omega)}, \lambda_\omega) = \frac{D(\mathbf{X}_{\text{Test}}^{(\omega)}, \lambda_\omega) - \mu_\omega}{\sigma_\omega}, \tag{7.6}$$

where $D_{\text{Mah}}(\mathbf{X}_{\text{Test}}^{(\omega)}, \lambda_\omega)$ denotes the normalised dissimilarity between the test pattern and the HMM of the claimed writer's signature. This normalisation assumes that the dissimilarity value in (7.5) is based on a *Mahalanobis* distance measure.

When only the mean vectors are estimated though, the dissimilarity value in (7.5) is based on an *Euclidean* distance measure. When this is the case, we found that significantly better results are obtained when the standard deviation of the dissimilarities of the training set, that is σ_ω in (7.6), is replaced by the mean μ_ω, that is

$$D_{\text{Eucl}}(\mathbf{X}_{\text{Test}}^{(\omega)}, \lambda_\omega) = \frac{D(\mathbf{X}_{\text{Test}}^{(\omega)}, \lambda_\omega) - \mu_\omega}{\mu_\omega}. \tag{7.7}$$

We used similar arguments for the normalisation strategy (equation (7.3)) in section 7.3.

Note that the mean μ_ω (6.10) provides a *reference* dissimilarity value. Also note that *both* σ_ω (6.11) *and* μ_ω (6.10) measure the *variability* of the writer's signature. This statement is valid, since the more the training signatures vary, the larger the average dissimilarity between the training sequences and the *trained* HMM will be. This results in a larger μ_ω. The *same* training set, that is the training set that we use in this section to obtain an appropriate threshold value, was also used to train the HMM (section 6.4.3). The HMM-based system therefore does not utilise a separate validation set. The performance of the HMM-based system does not improve when half of the training signatures from the Stellenbosch data set are used for validation purposes and the remaining signatures are used for training.

A *sliding* threshold τ, where $\tau \in [0, \infty)$, is again used here. When $D_{\text{Eucl}}(\mathbf{X}_{\text{Test}}^{(\omega)}, \lambda_\omega) < \tau$, that is when

$$D(\mathbf{X}_{\text{Test}}^{(\omega)}, \lambda_\omega) < \mu_\omega(1 + \tau), \tag{7.8}$$

the claim is accepted, otherwise the claim is rejected.

7.5 Concluding remarks

In this chapter we showed that the DTW-based and HMM-based systems developed in this dissertation use very similar verification protocols. The DTW-based system matches a test sequence with a template (reference) sequence, while the HMM-based system matches a test sequence with an appropriate HMM, using the Viterbi algorithm. In this way a dissimilarity value (distance measure) is obtained. In order to obtain a global threshold for all writers, each dissimilarity value is normalised using only the mean distance between the respective training patterns and the specific model. A sliding threshold is then used to obtain the error rates.

In the next chapter we discuss the two data sets, that are used in this dissertation, in some detail and present some experimental results.

Chapter 8

Experiments

8.1 Introduction

We presented the DTW-based and HMM-based systems developed in this dissertation in chapters 4 to 7. In this chapter we implement these systems on two independent data sets. We present these data sets in section 8.2 and give the experimental setup in section 8.3. We present the results in section 8.4 and compare these results to those of existing systems in section 8.5. In section 8.6 we compare the performance of the HMM-based system to that of a human being. Finally, in section 8.7, we investigate the performance of both of these systems when only random forgeries are considered.

8.2 Data

Experiments are conducted on two different data sets. The first data set, which we shall call the Stellenbosch data set, consists of signature images that were captured off-line on designated areas on blank sheets of paper. The second data set, which we shall call Dolfing's data set, consists of dynamic signatures that were originally captured for Hans Dolfing's Ph.D. thesis (Dolfing (1998b)). We use the pen-tip coordinates to convert each dynamic signature into an "ideal" signature image, that is, a signature that contains no background noise and has a uniform stroke width.

8.2.1 The Stellenbosch data set

The first data set contains 924 signatures from twenty-two writers. Ten training signatures were obtained from each writer during an initial enrollment session. Thirty-two test signatures, made up by twenty genuine signatures, six skilled forgeries and six casual forgeries, were subsequently obtained over a period of two weeks. The twenty genuine test signatures consist of two sets of ten signatures each. These signatures were supplied

by the same writers one week and two weeks after the enrollment session. The forgeries were obtained from six forgers. The casual forgeries were obtained first. Only the name of the writer was supplied to the forgers and they did not have access to the writer's signatures. The skilled forgeries were then obtained from the same group of forgers. They were provided with several samples of each writer's genuine signature and were allowed ample opportunity to practice. Each forger submitted one casual forgery and one skilled forgery for each writer. The writers were instructed to produce each signature within an appropriate rectangular region on a sheet of white paper. The signatures were then digitised with a flat-bed scanner at a resolution of 300 dots per inch. The genuine signatures were produced with different pens and the forgeries were produced with the same pens that were used for producing the genuine signatures. These signatures are free of excessive noise, smears and scratches.

8.2.2 Dolfing's data set

The second data set contains 4800 signatures from fifty-one writers and differs from the Stellenbosch data set in the sense that the signatures were originally captured on-line for Hans Dolfing's Ph.D. thesis (Dolfing (1998b)). Each of these signatures contains static *and* dynamic information captured at 160 samples per second. Each of these sample points contains information on pen-tip position, pen pressure, and pen tilt. Static signature images are constructed from this data using only the pen-tip position, that is the x and y coordinates, for those sample points for which the pen pressure is non-zero (see figure 8.1 (a)).

These signature images are therefore "ideal" in the sense that they contain virtually no background noise. This acquisition method also ensures a uniform stroke-width within each signature and throughout the data set. One hundred interpolatory points are inserted between each two successive coordinate pairs. Linear interpolation is used for this purpose and only those coordinate pairs that form part of the same "pen-down" segment are connected in this way (see figure 8.1 (b)). These coordinates are then rescaled in such a way that the range of the coordinate with the greater range is normalised to roughly 480, while the spatial aspect ratio of the entire signature is maintained. An image that consists of only zeros and of which the larger dimension is 512 is subsequently constructed. The normalised coordinates are then translated and rounded to the nearest integer so that the superimposed coordinates are roughly in the middle of the image. The pixel coordinates which coincide with these superimposed coordinates are then set to one. The resulting signature image has a stroke width of one (see figure 8.1 (c)). In order to obtain signatures with a stroke-width of five, each signature is dilated using a square morphological mask (structuring element) of dimension five (see figure 8.1 (d)).

Dolfing's data set contains four types of forgeries: random forgeries, over-the-shoulder forgeries, home-improved forgeries and professional forgeries (see section 1.2.8). A summary of the two data sets is given in table 8.1. It is important to note that, for each writer in these data sets, all the genuine signatures and all the high quality forgeries for all the *other* writers (in the particular data set) are considered to be random forgeries. This explains the fact that there are 19 404 and 240 000 random forgeries available for

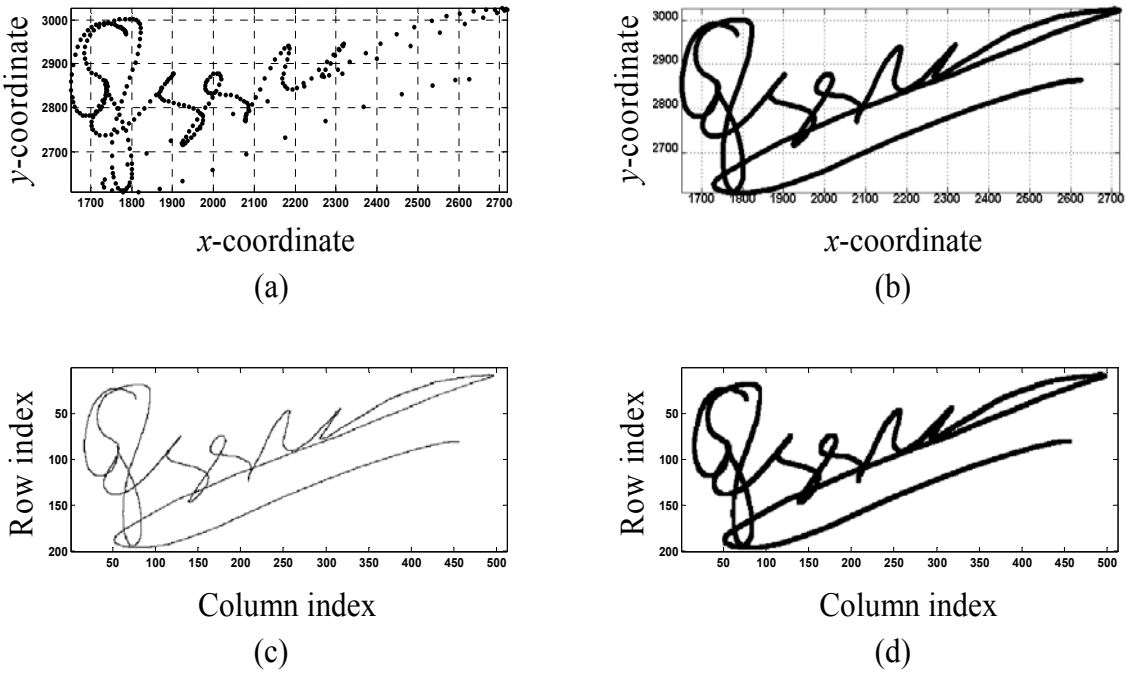


Figure 8.1: *Conversion of dynamic signature data into a static signature image. (a) The "pen-down" coordinates of a signature that was captured on-line. (b) One hundred interpolatory points are inserted between each two successive coordinate pairs. (c) A signature image with a stroke-width of one. (d) A signature image with a stroke-width of five.*

the Stellenbosch data set and Dolfing's data set respectively.

8.3 Experimental setup

The Stellenbosch data set. We consider thirty genuine signatures, six skilled forgeries and six casual forgeries for each writer. For each writer, ten genuine signatures are used for training and twenty for testing. No genuine signatures are used for validation purposes.

Dolfing's data set. We consider thirty genuine signatures for each writer, an average of 58.8 amateur forgeries per writer, and an average of 5.3 professional forgeries per writer. For each writer, fifteen genuine signatures are used for training and fifteen for testing. No genuine signatures are used for validation purposes.

Data set	Data acquisition method	Number of writers	Training signatures per writer	Number of genuine test signatures
Stellenbosch	Off-line	22	10	440
Dolfing	On-line	51	15	765

Data set	Number of forgeries			
	Skilled		Casual	Random
Stellenbosch	132		132	19 404
	Professional	Amateur		
Dolfing	270	3 000	–	240 000

Table 8.1: *Summary of the Stellenbosch data set and Dolfing's data set. These data sets are used to evaluate the performance of the DTW-based and HMM-based off-line signature verification systems developed in this dissertation.*

8.4 Results

8.4.1 DTW-based system

The DTW-based system developed in this dissertation is computationally less efficient than the HMM-based system (see section 9) and we present these results to place the performance of the HMM-based system (section 8.4.2) into perspective. We only implement the DTW-based system on the Stellenbosch data set.

Figure 8.2 shows the FRR and FAR as functions of the threshold parameter $\tau \in [0, 3]$, when $d = 512$, $N_\Theta = 128$ and $H_{\text{Vec}} = 40$. The *linear* approach of section 4.3.2 was used to achieve observation sequence alignment. The FRR, the FAR for a test set that contains only skilled forgeries, and the FAR for a test set that contains only casual forgeries are plotted on the same system of axes. When, for example, a threshold of $\tau = 0.45$ is selected, equation (7.4) implies that all the test patterns for which $D(\mathbf{X}^{(\omega)}_{\text{Test}}, \mathbf{X}^{(\omega)}_{\text{Ref}}) \geq 1.45\mu_\omega$ are rejected − the other patterns are accepted. When only skilled forgeries are considered, this threshold selection will ensure an EER of approximately 18%. When only casual forgeries are considered, this algorithm achieves an EER of 4%.

Similar results are obtained when the *non-linear* (DTW-based) approach of section 5.4 is used to achieve observation sequence alignment. This implies that the optimal alignment between two observation sequences is more or less linear. This is represented by an almost straight path (parallel to the diagonal) on the search grid, that is a path similar to the paths shown in figure 5.5. When the search is restricted to a bandwidth of $H_{\text{Seq}} < N_\Theta$, the performance of the DTW-based system does not deteriorate significantly. We expected this, since most of the signatures in the Stellenbosch data set are rotated in more or less the same way. This is the result of the fact that the writers were instructed to produce each signature within an appropriate rectangular region on a white sheet of paper. The performance of the DTW-based system also deteriorates when d, N_Θ or H_{Vec} is decreased.

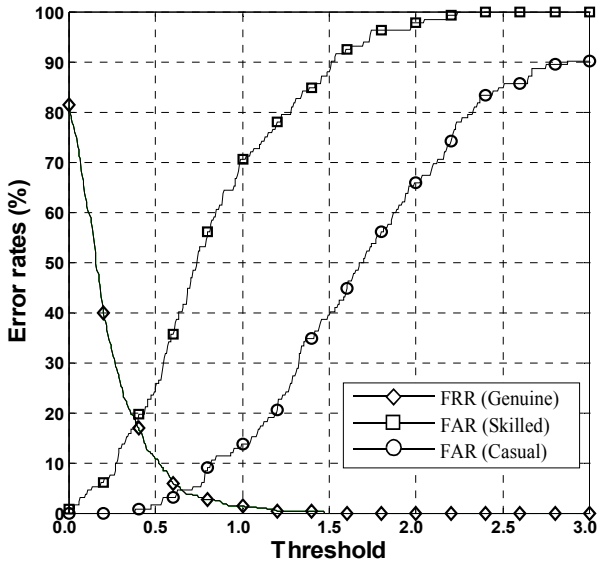


Figure 8.2: *DTW-based system implemented on the Stellenbosch data set. Graphs for the FRR and the FAR, when* $d = 512$, $N_\Theta = 128$ *and* $H_{\text{Vec}} = 40$. *The linear approach of section 4.3.2 was used to achieve observation sequence alignment. When only skilled forgeries and only casual forgeries were considered, EERs of approximately* 18% *and* 4%, *respectively were obtained.*

8.4.2 HMM-based system

The Stellenbosch data set. Let ℓ denote the number of allotted forward links in the HMM. This is equivalent to $\ell - 1$ state skips (see figure 6.1). Figure 8.3 shows the FRR and FAR as functions of the threshold parameter $\tau \in [-0.1, 1]$, when $d = 512$, $N_\Theta = 128$, $N = 64$, and $\ell = 1$. The number of feature vectors ($T = 2N_\Theta = 256$) is therefore four times the number of HMM states ($N = 64$).[1] The FRR, the FAR for a test set that contains only skilled forgeries, and the FAR for a test set that contains only casual forgeries are plotted on the same system of axes. When, for example, a threshold of $\tau = 0.16$ is selected, equation (7.8) implies that all the test patterns for which $D(\mathbf{X}^{(\omega)}_{\text{Test}}, \lambda_\omega) \geq 1.16\mu_\omega$ are rejected − the other patterns are accepted. When only skilled forgeries are considered, this threshold selection will ensure an EER of approximately 18%. When only casual forgeries are considered, this algorithm achieves an EER of 4.5%.

Tables 8.2 and 8.3 tabulate the EER, as well as a local FRR and FAR, for various values of d, N_Θ, N, and ℓ. It is clear that when the dimension of the feature vectors is decreased from $d = 512$ to $d = 256$ or even to $d = 128$, the performance of the system is not significantly compromised. The performance of the HMM-based system is generally enhanced when the number of feature vectors, that is $T = 2N_\Theta$, or the number of states in the HMM, that is N, is increased. The best results are obtained when only one forward link is allowed in the HMM, that is when $\ell = 1$.

Dolfing's data set. The results for the HMM-based algorithm developed in this disser-

[1] In speech recognition the number of feature vectors is often 1.2 times the number of HMM states.

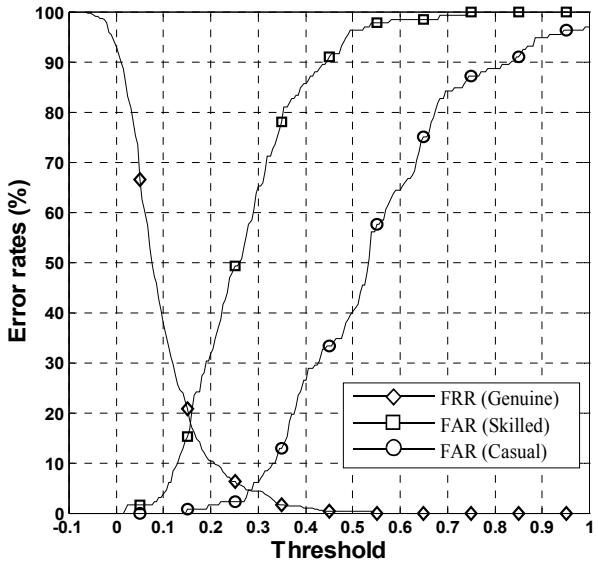


Figure 8.3: *HMM-based system implemented on the Stellenbosch data set. Graphs for the FRR and the FAR, when* $d = 512$, $N_\Theta = 128$, $N = 64$, *and* $\ell = 1$. *When only skilled forgeries and only casual forgeries were considered, EERs of approximately* 18% *and* 4.5%, *respectively were obtained.*

tation is compared to the results of an algorithm that was developed by Hans Dolfing for his Ph. D. thesis (Dolfing (1998b)).

Dolfing developed HMM-based algorithms for the verification and recognition of *dynamic* signatures. Since the HMM-based algorithm developed in this dissertation is suited for the verification of static signature images, we compare the results for this algorithm to those of Dolfing's algorithm which only considers the spatial coordinates of each dynamic signature. It is important to note that, in order to construct an HMM for each writer's signature, Dolfing's algorithm also uses the *sequence* in which these spacial coordinates were produced. The algorithm developed in this dissertation is therefore at a disadvantage, since it has no knowledge of the stroke sequence, and simulated time-evolution had to be created by taking the DRT of each signature image.

Figure 8.4 shows the FRR and FAR as functions of the threshold parameter $\tau \in [-0.1, 1]$, when $d = 512$, $N_\Theta = 128$, $N = 64$, and $\ell = 1$. The FRR, the FAR for a test set that contains only professional forgeries, and the FAR for a test set that contains only amateur forgeries are plotted on the same system of axes. The error rates for Dolfing's algorithm, when applied to only professional forgeries, were not published. For illustrational purposes however, we include the FAR for only professional forgeries in figure 8.4. When only amateur forgeries are considered, the HMM-based algorithm developed in this dissertation achieves an EER of 12.2%. This result compares well with the result for Dolfing's algorithm. Dolfing's algorithm (see Dolfing (1998b), p. 160) achieves an EER of 13.3%. These results are tabulated in table 8.4. It is interesting to note that, when *all* the available dynamic data is used, Dolfing's algorithm (Dolfing (1998b), p. 169) achieves an EER of only 2%, compared to an EER of 13.3% when only static data is used. This implies

d	N_Θ	N	ℓ	FRR (%)	FAR (%)	EER (%)
512	128	64	1			
Skilled forgeries				10.2	25.4	**17.9**
Casual forgeries				0.2	32.6	**4.5**
512	128	64	2			
Skilled forgeries				10.2	25.7	18.5
Casual forgeries				0.2	34.1	4.6
512	128	64	4			
Skilled forgeries				10.2	25.9	18.2
Casual forgeries				0.2	35.6	4.6
512	128	32	1			
Skilled forgeries				10.2	25.5	19.2
Casual forgeries				0.2	34.8	5.4
512	128	16	1			
Skilled forgeries				10.2	32.1	20.7
Casual forgeries				0.2	43.1	6.2

Table 8.2*: Summary of results for the HMM-based system developed in this dissertation when implemented on the Stellenbosch data set.*

that, when all the dynamic data is used, the verification accuracy of Dolfing's algorithm improves by an order of magnitude.

From the above discussion it is clear that, when the HMM-based system developed in this dissertation is implemented on both the data sets, that is the Stellenbosch data set and Dolfing's data set, the system performs better when implemented on Dolfing's data set. This is to a large extent due to the fact that the systems developed in this dissertation do not attempt to normalise the stroke-width within each signature. Dolfing's signatures have a uniform stroke-width, while the stroke-width of the signatures in the Stellenbosch data set varies. We elaborate on this in section 10.1.3.

Other density functions and dimension reduction. We mentioned in section 7.4 that each state in the HMM used in this dissertation is represented by a PDF for which only the mean vector is estimated. The corresponding covariance matrix is kept fixed. The dissimilarity between an observation sequence and the HMM is therefore based on an Euclidean distance measure. Due to the high dimension of the feature vectors used in this dissertation, a Mahalanobis distance measure cannot be used. We attempted to improve the performance of the HMM-based system developed in this dissertation by applying the KL-transform (see section 3.4.2) to each feature vector. We subsequently modelled each HMM state with a rotation invariant ellipsoidal PDF (see section 3.4.1). Neither this approach, nor a subsequent dimension reduction, improves the performance of the system.

d	N_Θ	N	ℓ	FRR (%)	FAR (%)	EER (%)
256	128	64	1			
Skilled forgeries				10.2	25.3	**17.7**
Casual forgeries				0.2	32.6	**4.5**
256	128	32	1			
Skilled forgeries				10.2	26.4	19.4
Casual forgeries				0.2	35.6	5.4
256	64	32	1			
Skilled forgeries				10.2	25.4	18.4
Casual forgeries				0.2	34.8	5.3
128	64	32	1			
Skilled forgeries				10.2	24.6	**18.3**
Casual forgeries				0.2	34.8	**5.4**
128	32	16	1			
Skilled forgeries				0.2	38.7	22.0
Casual forgeries				0.2	48.4	6.2

Table 8.3*: Summary of results for the HMM-based system developed in this dissertation when implemented on the Stellenbosch data set (continued).*

8.5 Comparison with prior work

Due to a lack of common signature databases, it is difficult to directly compare the systems (that we developed) to the systems discussed in sections 2.2 to 2.8. When comparing, we first consider whether an existing system is similar to a system that we developed. When an existing system is fundamentally different from a system that we developed, a combination of this system and the system that we developed should result in a superior fused (merged) system. The relationship between different merging techniques and different measures of diversity is discussed in a recent paper by Shipp and Kuncheva (2002).

The DTW-based system developed in this dissertation is not entirely novel. We elaborate

	Amateur forgeries from Dolfing's data set	
	Dolfing's HMM-based on-line algorithm	HMM-based off-line algorithm developed in this dissertation
EER (%)	13.3	12.2

Table 8.4*: Comparison of the results for the HMM-based off-line signature verification algorithm developed in this dissertation with the results for an HMM-based on-line signature verification algorithm developed by Hans Dolfing for his Ph. D. thesis. Only the amateur forgeries from Dolfing's data set were considered.*

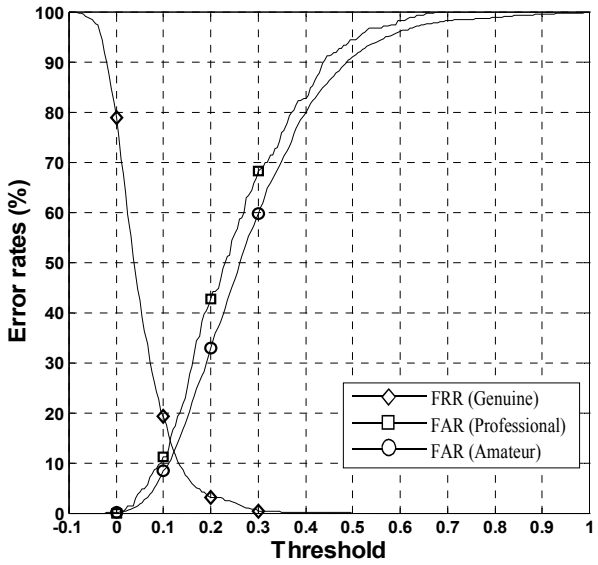


Figure 8.4*: HMM-based system implemented on Dolfing's data set. Graphs for the FRR and the FAR, when* $d = 512$, $N_\Theta = 128$, $N = 64$, *and* $\ell = 1$. *When only amateur (skilled) forgeries and only professional (skilled) forgeries were considered, EERs of approximately* 12% *and* 15%, *respectively were obtained.*

on this in section 8.5.1. The HMM-based system is novel in the sense that it utilises features, modelling techniques or verifiers that differs fundamentally from those used by existing systems. We elaborate on this in section 8.5.2.

8.5.1 DTW-based system

Unlike the HMM-based system developed in this dissertation (see section 8.5.2), the DTW-based system is not entirely novel. The DTW-based system is similar to two existing systems, which we discussed in chapter 2 (Shapiro et al. (1993); Fang et al. (2003)). We now elaborate on this.

In their survey, that covers the period from 1989 to 1993, Leclerc and Plamondon (1994) refer to the work of Shapiro and Bakalov (1993). The survey briefly mentions that Shapiro and Bakalov use projection profiles of signature images, in conjunction with a dynamic programming technique. This system is therefore similar to the DTW-based system developed in this dissertation. It is however not possible to compare their results to the results that we obtained, since neither their database(s), nor their results, are discussed in this survey.

Like the DTW-based method developed in this dissertation, the first method by Fang et al. (2003) also considers one dimensional projection profiles of each signature, but only in the horizontal and vertical directions. Like the DTW-based method developed in this dissertation, their modelling technique is also based on DTW, but differs slightly from the technique that we used, in the sense that the distance between the warped projection

profiles is not used in the decision. Instead, the positional distortion of each point of the sample profile, when warped onto a reference profile, is incorporated into a distance measure. Unlike the method we use, which is based on an Euclidean distance measure, they use a Mahalanobis distance measure. When only skilled forgeries are considered, their system achieves a best AER of 20.8% for binary signatures. The DTW-based system developed in this dissertation achieves an EER of approximately 18% when applied to the Stellenbosch data set.

8.5.2 HMM-based system

Of all of the systems discussed in sections 2.2 to 2.8, three systems are probably the closest to the HMM-based system developed in this dissertation.

As mentioned in the previous section, the first method by Fang et al. (2003) considers one dimensional projection profiles of each signature image. This is also the case for the HMM-based approach that we use. Their modelling *technique* however differs substantially from the technique that we use, and is based on DTW. When only skilled forgeries are considered, the HMM-based system developed in this dissertation achieves an EER of 17.7% when applied to the Stellenbosch data set and an EER of 12.2% when applied to Dolfing's data set. Their system achieves a best AER of 20.8%.

The method by Kaewkongka et al. (1999) utilises the Hough transform, which is similar to the Radon transform, but is able to detect not only straight lines, but other conical sections as well. Their modelling *technique* differs substantially from the technique that we use, and is based on a backpropagation NN. Their system is not a verification system though, and only aims to *recognise* signatures.

Like the HMM-based method developed in this dissertation, the method by Justino et al. (2001) also utilises an HMM to detect casual and skilled forgeries. However, they use *features* that are very different from the features that we use. A grid segmentation scheme is used to extract three features: a pixel density feature, a pixel distribution feature and an axial slant feature. Although their system achieves better error rates than the HMM-based system developed in this dissertation, one has to consider the fact that their system uses 40 training signatures per writer. The HMM-based system developed in this dissertation uses only 10 and 15 training signatures respectively, when applied to the Stellenbosch data set and Dolfing's data set.

The approaches described in Baltzakis et al. (2001), El-Yacoubi et al. (2000) and Sabourin et al. (1997c) use verifiers that are geared towards the detection of only *random* forgeries. The approaches described in Baltzakis et al. (2001), Deng et al. (1999), Fang et al. (2001), Fang et al. (2002), Guo et al. (2001), Mizukami et al. (2002), Quek et al. (2002) and Sabourin et al. (1997c) utilise *techniques* that are fundamentally very different from the techniques that we use, while the approaches described in Baltzakis et al. (2001), Deng et al. (1999), El-Yacoubi et al. (2000), Fang et al. (2001), Fang et al. (2002), Mizukami et al. (2002) and Sabourin et al. (1997c) utilise *features* that are fundamentally very different from the features that we use.

All of these systems are fundamentally different from the HMM-based system developed

in this dissertation. It is therefore very likely that a combination of any of these systems and the HMM-based system developed in this dissertation will result in a superior merged system. This will make their approach complementary to ours.

8.6 Manual verification

In order to compare the performance (the ability to classify a test signature as authentic or fraudulent) of the HMM-based system developed in this dissertation to that of a human being, we conducted *two* experiments on Dolfing's data set. To our amazement, we were unable to find any results for similar (manual verification) experiments in the literature.

8.6.1 Experimental setup

First manual verification experiment (with all the amateur forgeries used). For the first experiment we used all the genuine test signatures and *all* the amateur (over-the-shoulder and home-improved) forgeries in Dolfing's data set of 51 writers.

We supplied each student in a fourth year engineering class with a training set and a corresponding test set for *two* of the writers in Dolfing's data set. Each training set comprised of 15 genuine signatures. Each test set comprised of 15 genuine test signatures and 60 forgeries, with the exception of two writers (writer number 28 and writer number 38), for whom only 30 forgeries were available. Since there are an odd number of writers (51) in Dolfing's data set, one of the students received only one training set and one test set.

The training set for a specific writer was presented to a student on a single sheet of paper. The signatures (genuine test signatures and amateur forgeries) in a specific test set were randomly mixed, before the test set was also presented to the same student on a single sheet of paper. The students had no prior knowledge of how many forgeries each test set contained. Each student was given approximately 10 seconds per signature, to classify $2 \times 75 = 150$ test signatures.

This experiment has several shortcomings. For each writer, the number of forgeries (60) is very high compared to the number of genuine test signatures (15). This is an unrealistic scenario and one that the students (human verifiers) would not have expected. The fact that the *same* number of forgeries is present in *each* test set is also unrealistic.

Second manual verification experiment (randomly selected amateur forgeries). In order to measure the ability of *each* human verifier, we decided to rather present each individual with the signatures from *all* 51 writers in Dolfing's data set.

In this experiment we used different human verifiers than those used for the first experiment. These verifiers included faculty members, graduate students and departmental secretaries. We presented each individual with a training set (15 signatures) and a corresponding test set (*only* 15 signatures) for all 51 writers. Each test set contained a randomly selected number (any number between 0 and 15) of amateur (over-the-shoulder

and/or home-improved) forgeries. The remaining test signatures were randomly selected from the 15 genuine test signatures for the writer in question. It was therefore possible that a specific test set contained *only* genuine test signatures or *only* amateur forgeries. Each human verifier was presented with a total of $15 \times 51 = 765$ test signatures. The *total* number of genuine test signatures and forgeries turned out to be 432 and 333 respectively.

The training set (15 signatures) and the corresponding test set (15 signatures) for a specific writer were presented on two separate sheets of paper. An individual typically compared the test signatures, as a unit, with the corresponding training set and then decided which of the test signatures to reject.

Each individual verifier was instructed not to ponder over a decision, so as to simulate what a bank clerk is likely to do. As a result, most individuals took approximately 45 minutes to 1 hour to verify all the signatures, that is approximately 3.5 to 4.7 seconds per signature.

8.6.2 Results

First manual verification experiment (with all the amateur forgeries used). From table 8.5 it is clear that the manual verification results differ substantially from one writer to another. For writer number 7, for example, an FRR of 0% and an FAR of 0% is achieved. In contrast, for writer number 51, an FRR of 53.3% and an FAR of 58.3% is achieved.

When the signatures for all 51 writers are considered, an FRR of $174/765 = 22.7\%$ and a FAR of $652/3000 = 21.7\%$ is achieved. When the HMM-based system developed in this dissertation is implemented on the same signatures, the ROC graph (FAR against FRR) in figure 8.5 is obtained. The manual verification result is indicated by a circle. Since the circle is situated well above the ROC graph, it is clear that the HMM-based system outperforms the human verifiers.

Second manual verification experiment (randomly selected amateur forgeries). When the HMM-based system developed in this dissertation is implemented on the subset of signatures that was randomly selected for manual verification, that is 15 test signatures (0 to 15 genuine test signatures and 0 to 15 amateur forgeries) per writer, the ROC graph (FAR against FRR) in figure 8.6 is obtained. As expected, this graph closely resembles the graph in figure 8.5. The manual verification results (error rates) for twenty-two human verifiers are indicated by circles.

From figure 8.6 it is clear that only four human verifiers performed better than the HMM-based system. These results are indicated by circles below the ROC graph. It is important to note that circles (manual verification results) further away from the diagonal (that is where the FAR is equal to the FRR) are less significant than circles closer to the diagonal. After all, a human verifier can ensure that he/she performs as well as the HMM-based system by simply accepting or rejecting all the signatures. Also note that a human verifier has the advantage of being able to view *all* the test signatures for a specific writer *at once*, while the HMM-based system only considers one test signature at a time.

WR	FRs		FAs		WR	FRs		FAs		WR	FRs		FAs	
1	3	(15)	16	(60)	18	5	(15)	1	(60)	35	1	(15)	32	(60)
2	0	(15)	23	(60)	19	14	(15)	7	(60)	36	0	(15)	6	(60)
3	0	(15)	2	(60)	20	13	(15)	9	(60)	37	4	(15)	7	(60)
4	3	(15)	12	(60)	21	0	(15)	23	(60)	38	3	(15)	0	(30)
5	0	(15)	9	(60)	22	5	(15)	27	(60)	39	8	(15)	27	(60)
6	1	(15)	28	(60)	23	5	(15)	10	(60)	40	8	(15)	15	(60)
7	0	(15)	0	(60)	24	0	(15)	1	(60)	41	1	(15)	7	(60)
8	4	(15)	2	(60)	25	3	(15)	5	(60)	42	2	(15)	12	(60)
9	1	(15)	24	(60)	26	1	(15)	3	(60)	43	3	(15)	0	(60)
10	3	(15)	20	(60)	27	1	(15)	9	(60)	44	1	(15)	4	(60)
11	1	(15)	26	(60)	28	6	(15)	2	(30)	45	11	(15)	10	(60)
12	3	(15)	14	(60)	29	5	(15)	17	(60)	46	10	(15)	5	(60)
13	1	(15)	11	(60)	30	2	(15)	29	(60)	47	0	(15)	12	(60)
14	1	(15)	26	(60)	31	7	(15)	18	(60)	48	1	(15)	15	(60)
15	1	(15)	6	(60)	32	6	(15)	18	(60)	49	3	(15)	28	(60)
16	0	(15)	1	(60)	33	4	(15)	0	(60)	50	5	(15)	36	(60)
17	1	(15)	2	(60)	34	5	(15)	0	(60)	51	8	(15)	35	(60)

Table 8.5*: Results for the first manual verification experiment (with all the amateur forgeries used). For each writer (WR) the number of false rejections (FRs) and false acceptances (FAs) are tabulated with the number of genuine signatures and amateur forgeries in brackets. When the signatures for all* 51 *writers are considered, an FRR of* 22.7% *and a FAR of* 21.7% *is achieved. When the HMM-based system developed in this dissertation is implemented on the same set of signatures, an EER of* 12.2% *is achieved.*

Significance test. The above experiment consists of 22 identical independent trials, where a trial represents a specific human verifier's attempt to "outperform" the HMM-based system developed in this dissertation. Each trial has two possible outcomes, "success" or "failure". The experiment therefore follows a binomial distribution. The error rates for a human verifier are represented by a circle in figure 8.6. A trial is deemed successful when the circle is *below* the ROC graph obtained for the HMM-based system. The number of successes for the above experiment is therefore 4. Let us now assume that the probability of success is 0.5 and that this probability remains constant from one trial to another. Let the random variable X count the number of successes in all the trials. Given a 0.5 probability of success, the probability that a human verifier will outperform the HMM-based system 4 times or less, in 22 trials, is therefore given by

$$P(X \in [0,4]) = 0.5^{22} \sum_{k=0}^{4} \binom{22}{k} = 0.00217175483704 \approx 0.22\%. \tag{8.1}$$

The mean and variance of the binomial distribution $X \sim \text{Bin}(22, 0.5)$ is given by $\mu_X = 11$ and $\sigma_X^2 = 5.5$, respectively. Since a count of 4 is almost 3 standard deviations away from the mean, the probability of observing this count is extremely small. It is therefore safe to conclude that the probability of success is significantly less than 0.5, which implies that

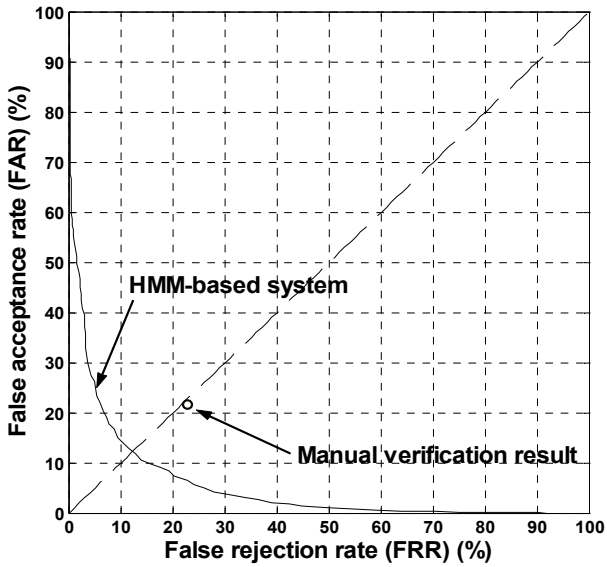


Figure 8.5*: ROC graph when the HMM-based system developed in this dissertation is implemented on Dolfing's data set (with* $d = 512$, $N_\Theta = 128$, $N = 64$, *and* $\ell = 1$*). All the genuine test signatures and all the amateur forgeries were considered. The result for the first manual verification experiment, where the same set of signatures are considered, is indicated by a circle, which represents an FRR of* 22.7% *and a FAR of* 21.7%*. Since the circle is situated well above the ROC graph, it is clear that the HMM-based system outperforms the human verifiers.*

the HMM-based system developed in this dissertation performs better than a typical human being. Therefore, as a substitute for human verification, the HMM-based system is a viable option.

8.7 Random forgeries

Even though the systems developed in this dissertation only focus on the detection of skilled and casual forgeries, and do not *specialise* in the detection of random forgeries − we gave the reasons for this in section 7.2 − we conclude this chapter by investigating these systems' ability to detect random forgeries.

Since random forgeries are easily obtainable − we simply use the signatures of other writers in the data set − it is worth while to implement the systems developed in this dissertation, without any impostor validation techniques (see section 7.2), on random forgeries and to analyse the results. The number of random forgeries that we used for this purpose is given in table 8.1.

When the HMM-based system (with $d = 512$, $N_\Theta = 128$, $N = 64$, and $\ell = 1$) is implemented on the Stellenbosch data set and Dolfing's data set respectively, and only random forgeries are considered, the error rates for different threshold values are given in figures 8.7 and 8.8. EER's of 4.5% and 4% are obtained for the Stellenbosch data set and

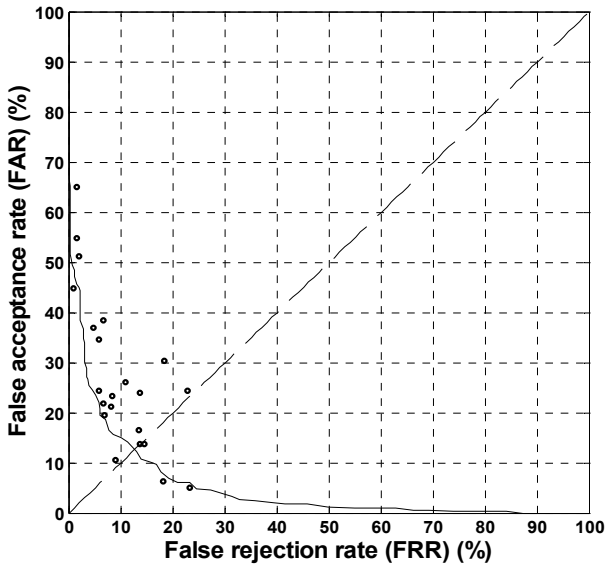


Figure 8.6: *ROC graph when the HMM-based system developed in this dissertation is implemented on a randomly selected subset of Dolfing's data set (with $d = 512$, $N_\Theta = 128$, $N = 64$, and $\ell = 1$). The results for the second manual verification experiment (when the same set of signatures are considered) are indicated by circles. Since four of these circles are below the ROC graph, it is clear that only four out of the twenty-two human verifiers outperformed the HMM-based system.*

Dolfing's data set, respectively. For the Stellenbosch data set, the EER of 4.5% when only random forgeries are considered is the same as the EER when only casual forgeries are considered. Since we did not use any impostor validation techniques, this is not a surprising result.

It should be possible to incorporate impostor validation techniques into the systems developed in this dissertation, so that much better results are obtained for random forgeries.

8.8 Concluding remarks

In this chapter we evaluated the performance of the DTW-based and HMM-based signature verification systems developed in this dissertation. We tested these systems on two independent data sets. When implemented on the Stellenbosch data set − these signatures were originally captured off-line − the DTW-based and HMM-based systems achieve similar results. However, we show in the next chapter that the HMM-based system is computationally more efficient than the DTW-based system. When tested on Dolfing's data set − these signatures were originally captured on-line and then converted into static signature images − the HMM-based system developed in this dissertation does a little better than one of Hans Dolfing's on-line algorithms (see table 8.4). This on-line algorithm also considers the *sequence* in which the spatial coordinates were produced. The systems developed in this dissertation do not outperform all existing systems, how-

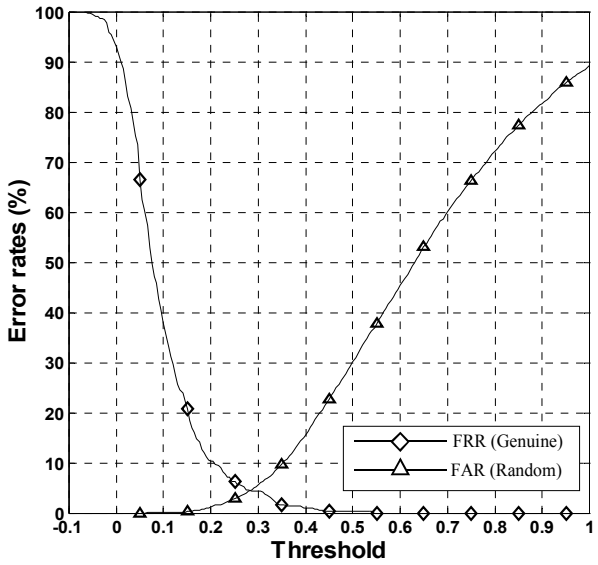


Figure 8.7: *HMM-based system implemented on the Stellenbosch data set. Graphs for the FRR and the FAR, when* $d = 512$, $N_\Theta = 128$, $N = 64$, *and* $\ell = 1$. *Only random forgeries were considered and no impostor validation techniques were used. An EER of* 4.5% *was obtained.*

ever, these systems are fundamentally different from the HMM-based system that we developed. It is therefore very likely that a combination of any of these systems and this HMM-based system will result in a superior merged system. This will make their approach complementary to ours. The HMM-based system developed in this dissertation outperforms most human verifiers, therefore, as a substitute for human verification this system is a viable option.

In the next chapter we discuss the computational requirements for the DTW-based and HMM-based systems developed in this dissertation.

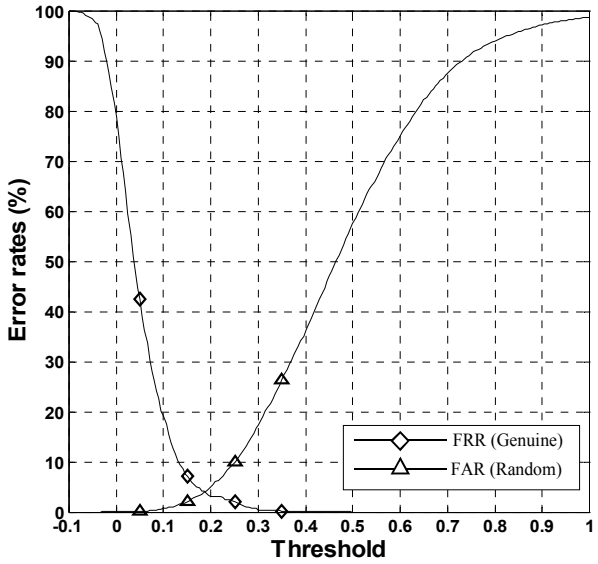


Figure 8.8: *HMM-based system implemented on Dolfing's data set. Graphs for the FRR and the FAR, when* $d = 512$, $N_\Theta = 128$, $N = 64$, *and* $\ell = 1$. *Only random forgeries were considered and no impostor validation techniques were used. An EER of* 4% *was obtained.*

Chapter 9

Computational requirements

9.1 Introduction

In this chapter we give a detailed analysis of the computational requirements for the systems developed in this dissertation. An automatic signature verification system can only be economically viable when the processing requirements are feasible. The practicality of the systems developed in this dissertation as real-time applications depends on the number of floating point operations (flops) required to verify a test signature.

The bulk of these flops are required to calculate the DRT of a test signature during the feature extraction phase (section 9.2). The other image processing algorithms, which are executed during the feature extraction phase (for example, median filtering and linear interpolation), require significantly fewer flops than the calculation of the DRT. We therefore do not discuss the computational requirements for these algorithms in more detail.

For the *DTW-based system* developed in this dissertation, the number of flops required to match a test sequence (without aligning it first) with a reference sequence, and the number of flops required to calculate a DRT, is of the same order of magnitude. For the *HMM-based system* developed in this dissertation, very few flops are required to match a test sequence with an HMM. We now give a more detailed analysis of the computational requirements.

9.2 Discrete Radon transform

Assume that an original signature image consists of Ψ pixels and that N_Θ angles (between 0^{O} and 180^{O}) are used to calculate the DRT. When N_φ (that is the number of beams per angle) is taken to be equal to the highest dimension of the original image, the number of flops required to calculate the DRT is on the order of $4\Psi N_\Theta$ (see Bracewell (1995)). This implies that for an average image of 500×300 pixels, with $N_\Theta = 128$ and $N_\varphi = 500$, the

number of flops required to calculate the DRT is on the order of 7.68×10^7.

For the *HMM-based system* developed in this dissertation we have the following. With the above parameters, an EER of 17.9% is achieved, when only skilled forgeries from the Stellenbosch data set are considered. However, the computational requirements can be significantly reduced, without compromising the performance of the system (see table 8.2). The number of flops required to calculate the DRT of an image of 256×128 pixels, with $N_\Theta = 64$ and $N_\varphi = 256$, is on the order of 8.4×10^6. With these parameters, an EER of 18.4% is achieved, when only skilled forgeries from the Stellenbosch data set are considered.

9.3 Matching

9.3.1 DTW-based algorithm

When we use the algorithm of section 5.3.2 to calculate the DTW-based distance between two *feature vectors*, we have to consider

$$d^2 - (d - H_{\text{Vec}})(d - H_{\text{Vec}} - 1) \tag{9.1}$$

nodes, where H_{Vec} denotes the bandwidth and d the feature vector dimension. For each of these nodes, the following operations are required. The calculation of the distance between the related test and reference features (components) requires one subtraction and one multiplication. The calculation of the distance associated with the partial optimal path requires one addition. This addition is not necessary for the first point. This implies that the number of flops required to match two feature vectors is given by

$$N_f = 3\left[d^2 - (d - H_{\text{Vec}})(d - H_{\text{Vec}} - 1)\right] - 1. \tag{9.2}$$

For the example in section 5.3.3, we have that $d = 8$ and $H_{\text{Vec}} = 4$. This implies that 52 nodes are considered and that 155 flops are required to match two feature vectors. For $d = 512$ and $H_{\text{Vec}} = 40$, $N_f = 119\,495$.

When N_Θ angles are used to calculate the DRT and the efficient *linear* approach of section 4.3.2 is used to align the observation sequences in such a way that provision is made for *any* possible rotation, the total number of flops necessary to match a test sequence with a reference sequence is on the order of

$$2N_\Theta^2\, N_f. \tag{9.3}$$

When this approach is used, with $d = 512$, $H_{\text{Vec}} = 40$, and $N_\Theta = 128$, on the order of 3.92×10^9 flops are required to match a test sequence with a reference sequence.

In general, when we deem it sufficient to only provide for rotations through an angle of up to $\theta_r \in (0°, 180°]$ degrees (in a CW or CCW direction) with respect to a reference signature, the number of flops necessary to match a test sequence with a reference sequence is on the order of

$$\frac{\theta_r {N_\Theta}^2\, N_f}{90}. \tag{9.4}$$

The above formula is not valid for $\theta_r = 0°$. When rotation invariance is assumed and no observation sequence alignment is performed, that is $\theta_r = 0°$, on the order of 1.53×10^7 flops are required to match a test sequence with a reference sequence. Note that a similar number of flops (on the order of 7.68×10^7) is required to calculate the DRT with these parameters (section 9.2).

When we use the *non-linear* (DTW-based) approach of section 5.4 to align the observation sequences, and choose $H_{\text{Seq}} = N_\Theta$, so that provision is made for any possible rotation, the number of flops necessary to match a test sequence with a reference sequence is on the order of

$$3N_\Theta^2 N_f. \tag{9.5}$$

When this approach is used, with $d = 512$, $H_{\text{Vec}} = 40$, $N_\Theta = 128$ and $H_{\text{Seq}}^{180°} = N_\Theta = 128$, on the order of 5.89×10^9 flops are required to match a test sequence with a reference sequence.

In general, when we use the *non-linear* (DTW-based) approach of section 5.4 to align the observation sequences, and we provide for rotations through an angle of up to $\theta_r \in (0°, 180°]$ degrees (in a CW or CCW direction) with respect to a reference signature, on the order of

$$4N_\Theta^2 N_f \left[1 - \left(\frac{360 - \theta_r}{360}\right)^2\right] \tag{9.6}$$

flops are required to match a test sequence with a reference sequence. This formula is also not valid for $\theta_r = 0°$. The required bandwidth is given by (5.33). With the above restriction, the non-linear (DTW-based) approach requires on the order of

$$\frac{720 - \theta_r}{360}$$

times more flops than the efficient linear method of section 4.3.2.

9.3.2 HMM-based algorithm

Since the states of the HMM used in this dissertation are organised in a ring and the dissimilarity value in (7.5) is based on an Euclidean distance measure, the number of flops required to match an observation sequence with an HMM is on the order of $T(N\ell + d)$. Therefore, despite the high dimensionality of the feature vectors, relatively few flops are required to match an observation sequence with an HMM. With $d = 512$, $T = 256$, $N = 64$ and $\ell = 1$, on the order of only 147 456 flops are required. With these parameters, an EER of 17.9% is achieved when only skilled forgeries from the Stellenbosch data set are considered. With $d = 256$, $T = 128$, $N = 32$ and $\ell = 1$, on the order of only 36 864 flops are required. With these parameters, an EER of 18.4% is achieved when only skilled forgeries from the Stellenbosch data set are considered (see table 8.2).

9.4 DTW-based system versus HMM-based system

Suppose that both of the systems developed in this dissertation calculate the DRT of a raw signature image using $d = 512$ (feature vector dimension) and $N_\Theta = 128$ (number of angles), and that the HMM-based system models each writer's signature with an HMM with 64 states ($N = 64$) and one state skip ($\ell = 1$). Also suppose that the DTW-based system uses a bandwidth of 40 for feature vector alignment ($H_{\text{Vec}} = 40$), that the "linear" approach is used for observation sequence alignment, and that the observation sequences are aligned is such a way that rotation invariance is ensured through *all* possible angles ($\theta_r = 180^\circ$). With these parameters the HMM-based system requires on the order of

$$\frac{7.68 \times 10^7 \text{ (DRT)} \ + 3.92 \times 10^9 \text{ (Matching)}}{7.68 \times 10^7 \text{ (DRT)} \ + 1.47 \times 10^5 \text{ (Matching)}} \approx 51.9$$

times fewer flops than the DTW-based system to verify a test signature *from scratch*.

When all the signatures are produced on a reference line (for example on a cheque), is reasonable to assume that they will not be rotated more than say 35° (in a CW or CCW direction) with respect to a reference signature. When we use the same parameters as for the previous case (with $\theta_r = 180^\circ$), but with

$$\theta_r = (25/128) \times 180^\circ = 35.15625^\circ,$$

the HMM-based system requires on the order of 10.9 times fewer flops than the DTW-based system to verify a test signature from scratch.

When we again use the above parameters, but this time with *no alignment* performed by the DTW-based system ($\theta_r = 0^\circ$), the HMM-based system requires only on the order of 1.2 times fewer flops than the DTW-based system to verify a test signature from scratch. However, for this to work, we have to be sure that all the original (raw) signatures are rotated in the same way.

It is therefore clear that the main advantage of the HMM-based system is the fact that it achieves rotation invariance much more efficiently than the DTW-based system. A detailed comparison of the computational requirements for the systems developed in this dissertation is given in table 9.1.

In the next chapter we discuss some outstanding issues before we conclude this dissertation.

DRT			**HMM-based system**				**DTW-based system**				**Ratio**
			Matching			DRT+ match	Matching *(linear alignment)*			DRT+ match	(DTW/ HMM)
d	N_Θ	flops	N	ℓ	flops	flops	H_{Vec}	θ_r	flops	flops	
512	128	7.68e7	64	1	1.47e5	7.69e7	40	180°	3.92e9	3.99e9	51.9
512	128	7.68e7	64	1	1.47e5	7.69e7	40	90°	1.96e9	2.03e9	26.4
512	128	7.68e7	64	1	1.47e5	7.69e7	40	35.2°	7.65e8	8.42e8	10.9
			Matching			DRT+ match	Matching *(non-linear alignment)*			DRT+ match	
d	N_Θ	flops	N	ℓ	flops	flops	H_{Vec}	θ_r	flops	flops	
512	128	7.68e7	64	1	1.47e5	7.69e7	40	180°	5.87e9	5.95e9	77.3
512	128	7.68e7	64	1	1.47e5	7.69e7	40	90°	3.43e9	3.50e9	45.5
512	128	7.68e7	64	1	1.47e5	7.69e7	40	35.2°	1.45e9	1.53e9	19.9
			Matching			DRT+ match	Matching *(no alignment)*			DRT+ match	
d	N_Θ	flops	N	ℓ	flops	flops	H_{Vec}	θ_r	flops	flops	
512	128	7.68e7	64	1	1.47e5	7.69e7	40	–	1.53e7	9.21e7	1.2

Table 9.1: *Computational requirements for the HMM-based and DTW-based off-line signature verification systems developed in this dissertation. The flops are given in order of magnitude. With the above parameters* $N_f = 119\,495$ *(see equation (9.2)) for each case.*

Chapter 10

Outstanding issues and conclusion

10.1 Outstanding issues and future work

The performance of the systems developed in this dissertation can be improved in a number of ways, including the use of imposter validation techniques to better detect random forgeries (section 10.1.1), the inclusion of local features (section 10.1.2), and stroke-width normalisation (section 10.1.3).

Some aspects of the systems developed in this dissertation need to be investigated further, for example the robustness of these systems with respect to variations in stroke-width and high noise densities (sections 10.1.3 and 10.1.4), the extent to which these systems are complementary to existing systems and to each other (section 10.1.5), the significance of the evolution of a writer's signature over a long period of time (section 10.1.6), and the prospect of also modelling the orientation of a signature (section 10.1.7). It should also be possible to apply the algorithms developed in this dissertation in other areas of pattern recognition (section 10.1.8). We now discuss these issues in more detail.

10.1.1 Random forgeries

The systems developed in this dissertation aim to detect only skilled and casual forgeries. However, many forged cheques contain random forgeries. As we mentioned in section 7.2, systems that aim to detect random forgeries use very specific verifiers. Subsets of other writers' training sets can be used to model "typical" forgeries. This is called "impostor validation" and can be achieved through strategies like test normalisation (Auckenthaler et al. (2000)). These techniques enable one to construct verifiers that detect random forgeries very accurately (El-Yacoubi et al. (2000); Sabourin et al. (1997c)).

It should be possible to incorporate these impostor validation techniques into the systems developed in this dissertation, so that much better results than those reported in section 8.7, are obtained for random forgeries.

10.1.2 Local features

The feature extraction method, for both of the systems developed in this dissertation, is based on the calculation of the DRT. This has several advantages (see section 4.4). However, this feature extraction technique also has certain restrictions. Each entry of a DRT represents the *cumulative* intensity of the pixels (of a raw signature image) that lie within a specific "beam" (see section 4.2). This implies that the systems developed in this dissertation use only *global* features. We expect a significant improvement in the results reported in this dissertation when local features, that is features at the stroke and sub-stroke level, are incorporated into these systems. Nel, Du Preez, and Herbst (2005), for example, suggest that this can be achieved by estimating the pen trajectory of a static test signature using the dynamic data of a genuine signature that was captured on-line. This is currently being investigated.

10.1.3 Stroke-width

The systems developed in this dissertation do not attempt to normalise the stroke-width within each signature. We reported in section 8.4.2 that, when the HMM-based system is implemented on both data sets, that is the Stellenbosch data set and Dolfing's data set, the system performs better when implemented on Dolfing's data set. When only amateur (skilled) forgeries are considered, EERs of approximately 18% and 12% are achieved for the Stellenbosch data set and Dolfing's data set, respectively. This is to a large extent due to the fact that Dolfing's signatures were originally captured on-line. Dolfing's signatures are therefore free of noise and have a *uniform* stroke-width of five pixels (see section 8.2.2). The signatures in the Stellenbosch data set, on the other hand, were captured off-line. As a result, the stroke-width *varies* from one signature to another, as well as within a specific signature image (see section 8.2.1).

It is reasonable to assume that, when applied to the Stellenbosch data set, the performance of the HMM-based system developed in this dissertation will improve significantly when the stroke-width of the signatures is normalised. Stroke-width normalisation is not a trivial task though and can be approached through morphological dilation and erosion (Gonzalez and Woods (2002), p. 523). Stroke-width normalisation is typically achieved through skeletonisation, followed by morphological dilation. Most standard skeletonisation techniques do not remove artifacts, which usually occur where strokes intersect. Zou and Yan (2001) developed a good skeletonisation algorithm that minimises these artifacts. Rocha (2003) improved this skeletonisation algorithm.

We did not conduct an experiment that estimates the effect of stroke-width variations on the overall performance of the systems developed in this dissertation. However, it should be easy to conduct such an experiment on Dolfing's data set. Since the signature images in Dolfing's data set were generated from the pen-tip coordinates obtained from dynamic signature data, they are effectively already skeletonised. One should therefore be able to generate new data sets with predetermined variations in stroke-width.

10.1.4 Noise

In this dissertation we assumed that a test signature has already been extracted from the background of a document, e.g. a cheque. We therefore assumed that a test signature is relatively free of smears and scratches. We explained in section 4.3.1 that a median filter is used to get rid of speckle noise in a raw signature image. Since it is unrealistic to assume that a test signature will be perfectly segmented from the background of the document − this is especially true for cheques − it is reasonable to expect that a significant amount of noise will be present in the extracted signature image. It is therefore sensible to conduct an experiment on Dolfing's data set which evaluates the robustness (with respect to noise) of the feature extraction technique developed in this dissertation. We did not conduct such an experiment. The signatures in Dolfing's data set have a uniform stroke-width. This will enable us to investigate the effect of different noise densities on the performance of the systems developed in this dissertation.

10.1.5 Merging of systems

It should be possible to combine the DTW-based and HMM-based systems developed in this dissertation using one of the merging techniques discussed in Shipp and Kuncheva (2002), and subsequently investigate the performance of the merged system. According to Shipp and Kuncheva (2002) one can only expect a merged system to outperform the original systems if the original systems are substantially different. The two systems developed in this dissertation use that same basic feature extraction technique, but use different models to represent a specific writer's signature. However, HMM-based and DTW-based models are related in the sense that an HMM-based representation generalises a DTW-based representation. We are therefore not very confident that a combined DTW/HMM-based system will outperform the individual systems discussed in this dissertation. It may however prove worth while to investigate this.

It will also be interesting to implement a few existing systems (see chapter 2) that either use features and/or pattern recognition techniques that are fundamentally different from those used by the HMM-based system developed in this dissertation. In this way one should be able to ascertain to what extent the HMM-based system is complementary to these existing systems.

10.1.6 Evolution of signatures

When the systems developed in this dissertation are used to authenticate signatures extracted from bank cheques, one should keep in mind that a client's signature tend to evolve over a long period of time. One way to deal with this problem is to collect training signatures at regular intervals. Another possible solution is to replace those training signatures that differs the most from the client's current signature model with one or more test signatures, and adapt the signature model accordingly. Test signatures that were accepted by the system, and differs the least from the client's current signature model, can be used for this purpose.

10.1.7 Signature orientation

The HMM used by the HMM-based system developed in this dissertation is constructed in such a way that it is equally likely to enter the HMM at any state (see section 6.4.1). It is also equally likely to exit the HMM from any state. This, in conjunction with the periodicity of the DRT, ensures that the HMM is a rotation invariant representation of the writer's signature. However, it is highly unlikely that somebody will write at an angle of 90° with respect to the horizontal. It is therefore possible that the HMM-based system will profit from a prior probability distribution favouring an angle of 0° with respect to the horizontal.

10.1.8 Other applications

It should be possible to also apply the feature extraction technique and signature models developed in this dissertation in other areas of pattern recognition, like off-line signature recognition, handwritten digit recognition, etc.

10.2 Conclusion

In this dissertation we developed two off-line signature verification systems: a DTW-based system and an HMM-based system. The feature extraction method, for both of these systems, is based on the calculation of the DRT.

The DRT is a stable and robust feature extraction method. At the same time, the DRT is especially useful, since it creates simulated time-evolution from one feature vector to the next. This enables the HMM-based system developed in this dissertation to model each writer's signature with an HMM. Feature vectors are extracted in such a way that rotation, shift and scale invariance are ensured.

The HMM-based system is efficient in the sense that relatively few flops are required to verify a test signature from scratch. By restricting the feature vector dimension and/or the length of an observation sequence, the computational requirements for the HMM-based system can be significantly reduced, without compromising the system's performance. This, together with the fact that the HMM-based system outperforms most human verifiers, makes the commercial implementation of the HMM-based system a viable option.

Given the robustness of the systems developed in this dissertation and the fact that only global features are considered, satisfactory results are obtained when they are applied to the Stellenbosch data set of 924 signatures from 22 writers. These results compare well with the results for existing systems.

When the HMM-based system is applied to Dolfing's data set of dynamic signatures, the results compare well with an algorithm by Dolfing which only considers the spatial coordinates of each dynamic signature. This is despite the fact that the HMM-based system developed in this dissertation only considers a static image of each signature,

while Dolfing's algorithm has knowledge of the sequence in which the spatial coordinates were produced. Dolfing's database consists of 4800 signatures from 51 writers.

The systems developed in this dissertation use only global features. However, most recent systems also utilise local features for verification purposes, that is features at stroke and sub-stroke level, and their feature extraction techniques require extensive preprocessing. The systems developed in this dissertation do not outperform all of these systems. These systems do however utilise either a technique or features that are fundamentally different from those used by the systems we developed. This is especially true for the HMM-based system, which implies that it is very likely that a combination of any of these systems and this HMM-based system, will result in a superior merged system.

We pointed out in this chapter that it is theoretically possible to improve the performance of the systems developed in this dissertation in a number of ways. The work conducted in this dissertation has therefore created many new opportunities for research in the field of off-line signature verification. We therefore intend to continue this research and hopefully further improve the performance of the systems developed in this dissertation. This is a very exciting prospect.

Bibliography

American Bankers Association (1998). *Check Fraud Survey Report.*

American Bankers Association (2000). Bad Checks Hit Small Banks. *American Banker.*

American Bankers Association (2000). Check Fraud Increases. *ABA Bankers News.*

American Bankers Association (2000). *Check Fraud Survey Report.*

Auckenthaler, R., Carey, M., and Lloyd-Thomas, H. (2000). Score Normalisation for Text-Independent Speaker Verification Systems, *Digital Signal Processing*, vol. 10, pp. 42-54.

Bajaj, R. and Chaudhury, S. (1997). Signature Verification Using Multiple Neural Classifiers. Pattern Recognition, vol. 30, no. 1, pp. 1-7.

Baltzakis, H. and Papamarkos, N. (2001). A New Signature Verification Technique based on a Two-Stage Neural Network Classifier, Engineering Applications of Artificial Intelligence, vol. 14, pp. 95-103.

Barua, S. (1992). Neural Networks and their Applications to Computer Security. *Proceedings of the SPIE - International Society for Optical Engineering*, pp. 735-742.

Bastos, L.C., Bortolozzi, F., Sabourin, R., and Kaestner, C. (1997). Mathematical Modelation of Handwritten Signatures by Conics. *Revista da Sociedade Paranaese de Matemática*, vol. 18, pp. 135-146.

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*, vol. 73, pp. 360-363.

Bracewell, R.N. (1995). *Two-Dimensional Imaging.* Prentice Hall.

Cardot, H., Revenu, M., Victorri, B., and Revillet, M.J. (1994). A Static Signature Verification System Based on a Cooperative Neural Network Architecture. *International Journal on Pattern Recognition and Artificial Intelligence*, vol. 8, no. 3, pp. 679-692.

Coetzer, J., Herbst, B.M., and Du Preez, J.A. (2004). Offline Signature Verification Using the Discrete Radon Transform and a Hidden Markov Model. *Eurasip Journal on Applied Signal Processing - Special Issue on Biometric Signal Processing, Bourland, H., Pitas, I., Lam, K. K., and Wang, Y., editors*, vol. 2004, no. 4, pp. 559-571.

Dehghan, M., Faez, K., and Fathi, M. (1997). Signature Verification Using Shape Descriptors and Multiple Neural Networks. *Proceeding of the IEEE TENCON - Speech and Image Technologies for Computing and Telecommunications Conference*, vol. 1, pp. 415-418.

Deller, J.R., Proakis, J.G. and Hansen, J.H. (1999). *Discrete-Time Processing of Speech Signals*. IEEE.

Deng, P.S., Liao, H.Y.M., Ho, C.W., and Tyan, H.R. (1999). Wavelet-Based Off-Line Handwritten Signature Verification. *Computer Vision and Image Understanding*, vol. 76, no. 3, pp. 173-190.

Djeziri, S., Nouboud, F., and Plamondon, R. (1998). Extraction of Signatures from cheque Background Based on a Filiformity Criterion. *IEEE Transactions on Image Processing*, vol. 7, no. 10, pp. 1425-1438.

Dolfing, J.G.A., Aarts, E.H.L., and Oosterhout, J.J.G.M. (1998). On-Line Signature Verification with Hidden Markov Models. *Proceedings of the Fourteenth Annual Conference on Pattern Recognition*, pp. 1309-1312.

Dolfing, J.G.A. (1998). *Handwriting Recognition and Verification - A Hidden Markov Approach*. Ph.D. Thesis.

El Yacoubi, A., Justino, E.J.R., Sabourin, R., and Bortolozzi, F. (2000). Off-Line Signature Verification Using HMMs and Cross-Validation. *IEEE International Workshop on Neural Networks for Signal Processing*, pp. 859-868.

Fang, B., Wang, Y.Y., Leung, C.H., and Tse, K.W. (2001). Off-Line Signature Verification by the Analysis of cursive strokes. *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 15, no. 4, pp. 659-673.

Fang, B., Leung, C.H., Tang, Y.Y., Kwok, P.C.K., Tse, K.W., and Wong, Y.K. (2002). Off-Line Signature Verification with Generated Training Samples. *IEE Proceedings - Vision, Image and Signal Processing*, vol. 149, no. 2, pp. 85 - 90.

Fang, B., Leung, C.H., Tang, Y.Y., Tse, K.W., Kwok, P.C.K., and Wong, Y.K. (2003). Off-line signature verification by tracking of feature and stroke positions. *Pattern Recognition*, vol. 36, pp. 91-101.

Foltyniewicz, R. and Sitnik, M. (1996). Verification of Persons via Face and Signature Analysis. *Proceedings of the International Conference on Image Processing (ICIP)*, vol. 3, pp. 495-498.

Forney, G.D. (1973). "The Viterbi algorithm". *Proceedings of the IEEE*, vol. 61, pp. 268-278.

Forte, A.M. and Impedovo, S. (1996). A New Adaptive Neural Network for an Off-Line Signature Verification System. Proceedings of the International Conference on Pattern Recognition, pp. 355-363.

Gonzalez, R.C. and Woods, R.E. (2002). *Digital Image Processing*, Prentice-Hall, New Jersey.

Guo, J.K., Doermann, D., and Rosenfeld, A. (1997). Local Correspondence for Detecting Random Forgeries. *Proceedings of the Fourth International Conference on Document Analysis and Recognition (ICDAR'97)*, pp. 319-323.

Guo, J.K., Doermann, D., and Rosenfeld, A. (2001). Forgery detection by local correspondence. *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 15, no. 4, pp. 579-641.

Gupta, J. and McCabe, A. (1998). A Review of Dynamic Handwritten Signature Verification. *Technical Report*, James Cook University, Australia.

Heckerman, D. (1999). A tutorial on learning with Bayesian networks. *Learning in graphical models*, pp. 9-17. MIT Press.

Huang, K. and Yan, H. (1997). Off-Line Signature Verification Based on Geometric Feature Extraction and Neural Network Classification. *Pattern Recognition*, vol. 30, no. 1, pp. 9-17.

Huang, K. and Yan, H. (2002). Off-Line signature verification using structural feature correspondence. *Pattern Recognition*, vol. 35, pp. 2467-2477.

Ismail, M.A. and Gad, S. (2000). Off-line Arabic signature recognition and verification. *Pattern Recognition*, vol. 33, no. 10, pp. 1727-1740.

Juang, B.H. (1985). Maximum likelihood estimation for multivariate stochastic observations of Markov chains. *AT&T Technical Journal*, vol. 64, no. 6, pp. 1235-1249.

Juang, B.H., Levinson, S.E., and Sondhi, M.M. (1986). Maximum likelihood estimation for multivariate mixture observations of Markov chains. *IEEE Transactions on Information Theory*, vol. 32, no. 2, pp. 307-309.

Justino, E.J.R., Bortolozzi, F., and Sabourin, R. (2001). Off-Line Signature Verification Using HMM for Random, Simple and Skilled Forgeries. *International Conference on Document Analysis and Recognition (ICDAR 2001)*, vol. 1, pp. 105-110, Seattle, USA.

Justino, E.J.R., Bortolozzi, F., and Sabourin, R. (2004). A Comparison of SVM and HMM Classifiers in Off-line Signature Verification. *To appear in Pattern Recognition Letters.*

Kaewkongka, T., Chamnongthai, K., and Thipakorn, B. (1999). Off-Line Signature Recognition Using Parameterised Hough Transform, Proceedings of the fifth International Symposium on Signal Processing and its Applications (ISSPA 1999), Brisbane, Australia.

Kak, A.C. and Slaney, M. (1988). *Principles of Computerized Tomographic Imaging.* IEEE, New York.

Koerich, A.L. and Lee, L.L. (1997). Automatic Extraction of Filled-in Information from Bank cheques Based on Prior Knowledge About Layout Structure. *Lecture Notes in Computer Science*, vol. 1339, pp. 322-333.

Leclerc, F. and Plamondon, R. (1994). Automatic Signature Verification: The State of the Art, 1989-1993. *International Journal on Pattern Recognition and Artificial Intelligence: Special Issue on Signature Verification*, vol. 8, no. 3, pp. 643-660.

Levinson, S.E., Rabiner, L.R., and Sondhi, M.M. (1983). An introduction to the application of the theory of probabilistic functions of a Markov process to automatic speech recognition. *Bell Syst. Tech. J.*, vol. 62, no. 4, pp. 1035-1074.

Liporace, L.A. (1982). Maximum likelihood estimation for multivariate observations of Markov sources. *IEEE Transactions on Information Theory*, vol. 28, no. 5, pp. 729-734.

Mighell, D.A., Wilkinson, T.S., and Goodman, J.W. (1989). Backpropagation and its application to Handwritten Signature Verification. *Advances in Neural Information Processing Systems 1, Touretzky, D.S., editor, Morgan Kaufman Publications*, pp. 340-347.

Mizukami, Y., Yoshimura, M., Miike, H., and Yoshimura, I. (2002). An off-line signature verification system using an extracted displacement function. *Pattern Recognition Letters*, vol. 23, pp. 1569-1577.

National Check Fraud Center (2000). *National Check Fraud Center Report.*

Nedbank (2000). *Nedbank ISS Crime Index*, vol. 4.

Nel, E-M., Du Preez, J.A., and Herbst, B.M. (2005). Estimating the Pen Trajectories of Static Signatures using Hidden Morkov Models. *To appear in IEEE Transactions on Pattern Analysis and Machine Intelligence.*

Nouboud, F. and Plamondon, R. (1994). Global parameters and curves for offline signature verification. *Proceedings of the International Workshop on Frontiers in Handwriting Recognition*, pp. 145-155.

Plamondon, R. and Lorette, G. (1989). Automatic Signature Verification and Writer Identification - The Sate of the Art. *Pattern Recognition*, vol. 22, no. 2, pp.107-131.

Plamondon, R. and Shihari, S.N. (2000). On-line and Off-line handwriting recognition: A comprehensive Survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 1, pp. 63-84.

Quek, C. and Zhou, R.W. (2002). Antiforgery: a novel pseudo product based fuzzy neural network driven signature verification system. *Pattern Recognition Letters*, vol. 23, pp. 1795-1816.

Rabiner, L.R. (1989). A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, vol. 77, pp. 257-286.

Rigoll, G. and Kosmala, A. (1998). A Systematic Comparison Between On-Line and Off-Line Methods for Signature Verification Using Hidden Markov Models. *Proceedings of the Fourteenth International Conference on Pattern Recognition (ICPR)*, vol. 2, pp. 1755-1757, Brisbane, Australia.

Rocha, J. (2003). Perceptually stable regions for arbitrary polygons. *IEEE Transactions on Systems, Man, and Cybernetics B*, vol. 33, no. 1, pp. 165-171.

Sabourin, R. and Drouhard, J.P. (1992). Off-Line Signature Verification Using Directional PDF and Neural Networks. *Proceedings of the Eleventh IAPR International Conference on Pattern Recognition*, pp. 321-325.

Sabourin, R., Plamondon, R., and Lorette, G. (1992). Off-Line Identification with Handwritten Signature Images: Survey and Perspectives. *Structured Document Image Analysis, Baird, H., Bunke, H., and Yamamoto, K., editors, Springer-Verlag*, pp. 219-234.

Sabourin, R., Cheriet, M., and Genest G. (1993). An Extended-Shadow-Code Based Approach for Off-Line Signature Verification. *Proceedings of the Second International Conference on Document Analysis and Recognition*, pp. 1-5.

Sabourin, R. and Genest, G. (1995). Définition et Évaluation d'une Famille de Représentations pour la Vérification hors Ligne des Signatures. *Traitement du Signal*, vol. 12, no. 6, pp. 587-596.

Sabourin, R. (1997). Off-Line Signature Verification: Recent Advances and Perspectives. *Proceedings of the First Brazilian Symposium on Document Image Analysis (BSDIA'97)*, pp. 84-98, Curitiba, Brazil.

Sabourin, R., Drouhard, J.P., and Wah, E.S. (1997). Shape Matrices as a Mixed Shape Factor for Off-Line Signature Verification. *Proceedings of the Fourth International Conference on Document Analysis and Recognition (ICDAR'97)*, pp. 661-665.

Sabourin, R., Genest, G., and Prêteux, F. (1997). Off-Line Signature Verification by Local Granulometric Size Distributions. *Transactions on Pattern Analysis and Machine Intelligence*, vol. 19, no. 9, pp. 976-988.

Santos, J.E.B., Bortolozzi, F., and Sabourin, R. (1997). A Simple Methodology to Bankcheck Segmentation. *Lecture Notes in Computer Science*, vol. 1339, pp.334-9999.

Shapiro, V.A. and Bakalov, I.S. (1993). Static Signature Verification as a Dynamic Programming Problem. *Proceedings of the Sixth International Conference on Handwriting and Drawing*, pp. 219-221, Paris.

Shipp, C.A. and Kuncheva, I.K. (2002). Relationships between combination methods and measures of diversity in combining classifiers. *Information Fusion*, vol. 3, pp. 135-148.

Toft, P. (1996). *The Radon Transform - Theory and Implementation.* Ph.D. Thesis.

Vapinik, V. (1998). Statistical Learning Theory. *Wiley*, New York.

Wilkinson, T.S. and Goodman, J.W. (1990). Slope histogram detection of forged handwritten signatures. *Proceedings SPIE - International Society for Optical Engineering*, pp. 293-304, Boston.

Xiao, X. and Leedham, G. (2002). Signature verification using a modified Bayesian network. *Pattern Recognition*, vol. 35, no. 5, pp. 983-995.

Yoshimura, M. and Yoshimura, I. (1997). An Application of the Sequential Dynamic Programming Matching Method to Off-Line Signature Verification. *Lecture Notes in Computer Science*, vol. 1339, pp. 299-310.

Zou, J.J. and Yan, H. (2001). Skeletonization of ribbon-like shapes based on regularity and singularity analysis. *IEEE Transactions on Systems, Man, and Cybernetics B*, vol. 31, no. 3, pp. 401-407.

Appendix A

Dynamic Programming: Key Concepts

Although there are a wide range of dynamic programming algorithms - the type of algorithm typically depends on the nature and constraints of the specific problem - they usually attempt to find a "shortest-distance" or "least-cost" path through a grid, like the one shown in figure A.1.

Each *node* in figure A.1 is indexed by an ordered pair of nonnegative integers. A *path* from node (s,t) to node (u,v) is an ordered set of nodes of the form

$$(i_0,j_0)(i_1,j_1)(i_2,j_2)(i_3,j_3),\ldots,(i_K,j_K). \tag{A.1}$$

We refer to a path as a *complete path* if $(i_0,j_0)=(0,0)$ (original node) and $(i_K,j_K)=(I,J)$ (terminal node). Depending on the problem, one or more restrictions can be placed on the intermediate nodes.

Three types of *costs* (or *distances*) can be assigned to a path, that is *transition costs*, *node-based costs*, and *combined costs*.

A transition cost is associated with a forward-going transition from one node (in a path) to another. We use the notation

$$D_{\text{Trans}}[(i_k,j_k)|(i_{k-1},j_{k-1})] \equiv \text{transition cost from node } (i_{k-1},j_{k-1}) \text{ to node } (i_k,j_k). \tag{A.2}$$

Although variations may occur, we usually assume that $D_{\text{Trans}}[\cdot]$ is a nonnegative quantity, and that any transition that originates at $(0,0)$ is costless. The latter assumption implies that

$$D_{\text{Trans}}[(i,j)|(0,0)]=0, \quad \text{for all } (i,j). \tag{A.3}$$

Node-based costs are associated with the nodes themselves, rather than with the transitions among them. We use the notation

$$D_{\text{Node}}(i,j) \equiv \text{cost associated with node } (i,j), \quad \text{for all } (i,j). \tag{A.4}$$

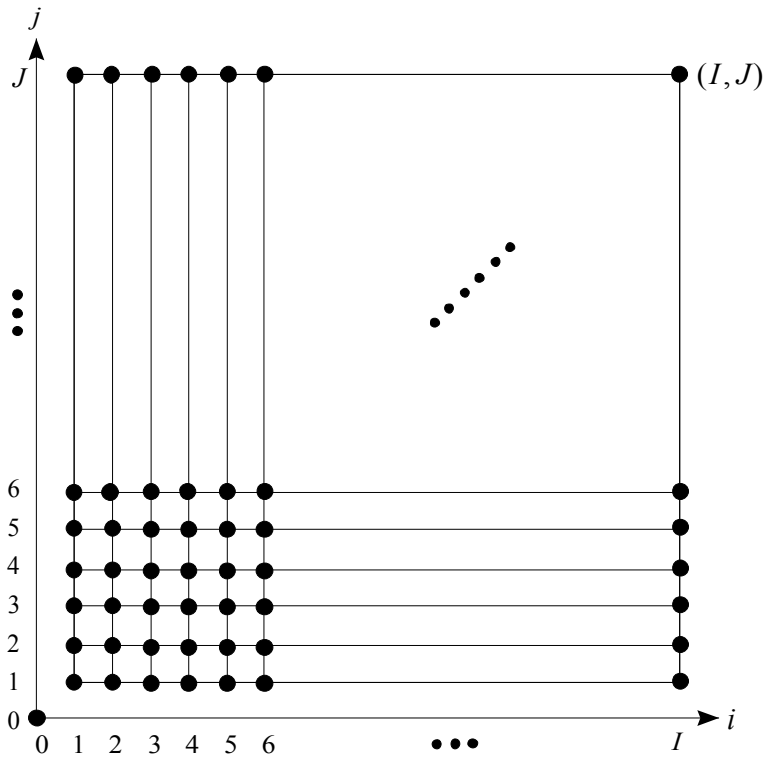


Figure A.1*: Grid used to illustrate dynamic programming.*

In general, we choose $D_{\text{Node}}(\cdot,\cdot)$ to be nonnegative. Problems are usually set up so that $(0,0)$ is costless. This implies that

$$D_{\text{Node}}(0,0) = 0. \tag{A.5}$$

However, variations may occur. For example, $D_{\text{Node}}(0,0)$ is taken to be unity if the node costs are combined by multiplication ((A.7) and (A.9)).

Combined costs are associated with both transitions and nodes. The transition and node-based costs are usually combined by addition at the given node,

$$D_{\text{Comb}}[(i_k, j_k)|(i_{k-1}, j_{k-1})] \equiv D_{\text{Trans}}[(i_k, j_k)|(i_{k-1}, j_{k-1})] + D_{\text{Node}}(i_k, j_k). \tag{A.6}$$

The most frequent exception to this case is when the costs are combined by multiplication,

$$D_{\text{Comb}}[(i_k, j_k)|(i_{k-1}, j_{k-1})] \equiv D_{\text{Trans}}[(i_k, j_k)|(i_{k-1}, j_{k-1})] \times D_{\text{Node}}(i_k, j_k). \tag{A.7}$$

Note that for the multiplication case it is required that $D_{\text{Trans}}[(i,j)|(0,0)] = 1$ and $D_{\text{Node}}(0,0) = 1$ for a "costless" initiation.

When combined costs are used, the distance associated with the complete path $D_{\text{Comb}}^{(\text{Compl})}$ is usually taken as the sum of the combined costs along the path, and is expressed as

$$D_{\text{Comb}}^{(\text{Compl})} = \sum_{k=1}^{K} D_{\text{Comb}}[(i_k, j_k)|(i_{k-1}, j_{k-1})], \tag{A.8}$$

where K is the number of transitions in the path, $(i_0, j_0) = (0, 0)$, and $(i_K, j_K) = (I, J)$. Expressions similar to (A.8) exist for the transition and node-based cases.

The objective of a dynamic programming algorithm is therefore to find the specific path that minimises $D_{\text{Comb}}^{(\text{Compl})}$. The most common variations on this problem include cases in which $D_{\text{Comb}}^{(\text{Compl})}$ is found by multiplication of the individual costs,

$$D_{\text{Comb}}^{(\text{Compl})} = \prod_{k=1}^{K} D_{\text{Comb}}[(i_k, j_k)|(i_{k-1}, j_{k-1})], \tag{A.9}$$

and cases in which $D_{\text{Comb}}^{(\text{Compl})}$ is to be maximised, rather than minimised.

Appendix B

Hidden Markov Models: Key Concepts

B.1 Types of HMMs

We distinguish between two major types of HMMs, namely continuous and discrete observation HMMs. For a *continuous observation HMM*, each observation is represented by a multidimensional *feature vector*. A sequence of T continuous observations is denoted by (6.1).

A *discrete observation HMM*, on the other hand, is used to model an observation sequence, where each observation is restricted to an alphabet of M symbols. We use the following notation for a sequence of T discrete observations,

$$\mathbf{O}_1^T = o_1, o_2, \ldots, o_T, \tag{B.1}$$

where $o_i,\ i = 1, 2, \ldots T$ denotes the ith symbol in the sequence. We refer to (6.1) and (B.1) as a continuous and discrete observation sequence, respectively.

Note that every continuous observation can be mapped onto a discrete observation, using an appropriate vector quantisation technique. Such a technique typically maps every region in the feature space onto a specific symbol (see figure B.1).

Although this thesis focuses on off-line signature verification, it is possible to extract a continuous observation sequence from an on-line signature in a much more intuitive way. For this reason we briefly introduce a simple feature extraction technique for on-line signatures in the next section, that is section B.2. We do this to provide the reader with a concrete example of what a continuous observation sequence may represent, before we formally introduce the theory of HMMs.

Although continuous observation HMMs are more popular than discrete observation HMMs the theory of HMMs can be explained in a more intuitive way through a simulation that illustrates the discrete case. Such a simulation is presented in section B.3.

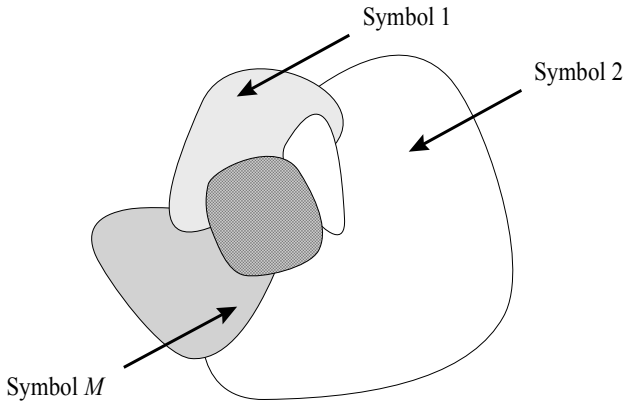


Figure B.1: *Conceptualisation of vector quantisation.*

Sections B.4 and B.5 discuss the theory of discrete HMMs. This theory is extended to the continuous case in section B.6. Some implementation issues are discussed in section B.7.

B.2 Example of a continuous observation sequence

In order to better explain the concept of a continuous observation sequence, we use this section to briefly discuss a simple feature extraction technique for on-line signatures.

On-line signatures are captured with a digitising tablet and a pressure sensitive pen. The digitising tablet records the two-dimensional coordinates of the successive points of the writing, pen pressure, as well as the angle and direction of the pen, at a device-specific sampling rate. The pen usually has a touch sensitive switch in its tip, so that only the "pen-down" samples (that is when the pen touches the paper) are recorded. The data is then stored as a function of time.

A *time frame* is a sequence of successive points (samples) of the writing. We assume that the feature extraction technique uses the same number of time frames, say T, for each signature. For each time frame, a feature vector is constructed, where each component represents a feature.

For the first time frame, for example, a feature vector may be constructed in such a way that the first four components represent the minimum and maximum displacement of the writing in the horizontal and vertical directions, that is $\Theta x_{\min}$, $\Theta x_{\max}$, $\Theta y_{\min}$, and $\Theta y_{\max}$. The next two components may represent the minimum and maximum pen pressure, that is $p_{\min}$ and $p_{\max}$. The next four components may represent the minimum and maximum velocity ($v_{\min}$ and $v_{\max}$) and the minimum and maximum acceleration ($a_{\min}$ and $a_{\max}$), etc. This vector represents the first observation in an observation sequence, that is $\mathbf{x}_1$. This process is then repeated for the other time frames, so that $\mathbf{x}_2$ to $\mathbf{x}_T$ are obtained. This concept is illustrated in figure B.2.

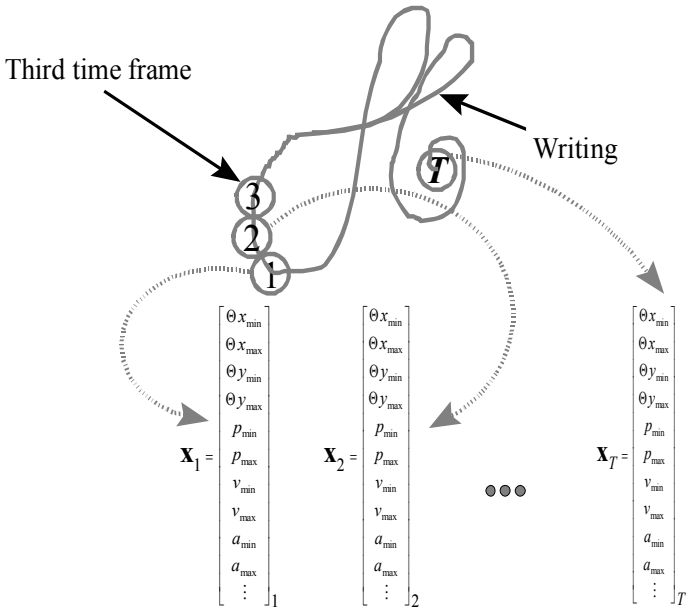


Figure B.2*: Conceptualisation of feature extraction for an on-line signature.*

From this discussion it is clear that a model is required that does not only describe the individual observations, but also describes the *relationship* between these observations, and even the relationship between *groups* of these observations. The remainder of this discussion on key HMM concepts is based on a paper by Rabiner (1989).

B.3 Simulation of a discrete observation HMM

In this section we simulate the process by which a discrete observation HMM generates a discrete observation sequence. Note that an HMM is not only a generator of observations. The value of an HMM rather lies in the fact that it can the trained by a set of patterns (observation sequences) that belongs to the same pattern class. The HMM then represents this class. A test pattern is matched with the trained HMM and the distance between the test pattern and the model is used for classification purposes. These concepts are discussed in more detail in section B.5. The following game simulates the process by which a discrete observation HMM generates a sequence of symbols:

Two players, player 1 and player 2, reside in separate rooms (see figure B.3). Player 1 is in a room with several urns, that is glass bowls like those used in lotteries. There is one small, detached urn, as well as N pairs of urns, where each pair consists of one large and one small urn. Each of the small urns contains a certain number of ping-pong balls, where each ball is numbered. This can be any number between 1 and N. Each of the large urns also contains a certain number of balls, where each ball has one of M possible colours. Player 2 is familiar with the content of each urn and with the values of M and

N, but (s)he does not know which pair of urns player 1 is visiting at a given point in time.

The HMM is represented by the urns and their contents. The actions of player 1 represent the process by which the HMM generates an observation sequence.

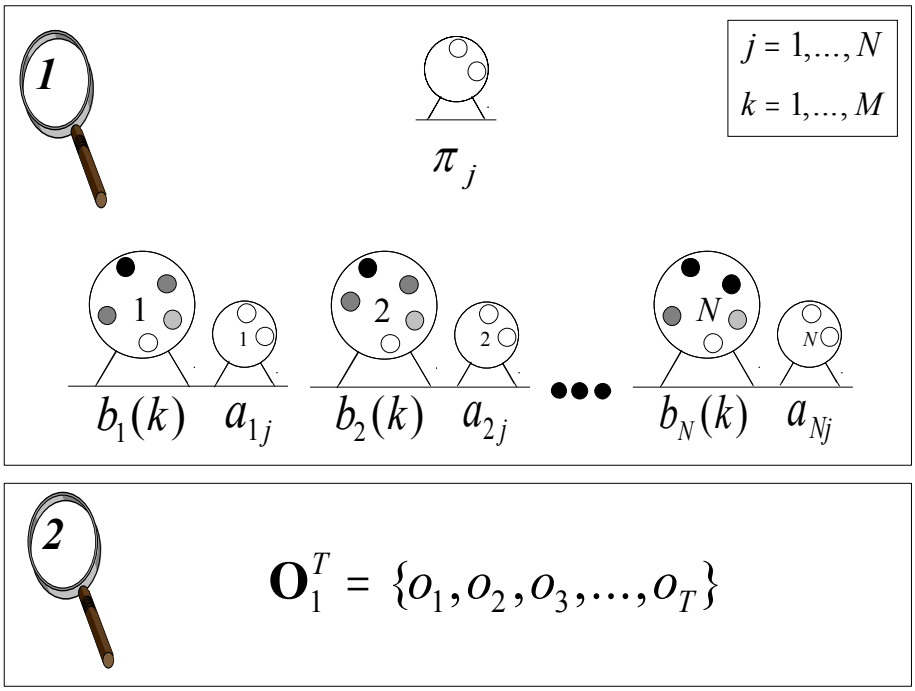


Figure B.3*: A game that simulates the process by which a discrete observation HMM generates an observation sequence,* $\mathbf{O}_1^T = \{o_1, o_2, \ldots, o_T\}$*. The "a", "b" and "π" parameters are explained in section B.4.*

The game proceeds as follows. Player 1 must select a ball from the small detached urn at random. Suppose that the selected ball has the number "2". The probability of that happening is the ratio of the number of balls in the detached urn with the number "2", and the total number of balls in the urn. This information is not conveyed to player 2 and the ball is placed back into the urn.

Since player 1 selected a ball with the number "2", (s)he must now go to the *second* pair of urns, select a ball from the large urn at random and tell player 2 what the colour of the ball is. Let's assume that this ball is red. Player 2 now records this information so that "`red`" occupies the first position in the observation sequence, that is the observation at the first clock time. The ball is placed back into the urn. Player 1 must now proceed to the corresponding small urn and select a ball at random. The number of this ball automatically determines the next pair of urns to be visited. The same pair of urns can be visited again. Player 1 now proceeds to the selected pair of urns, select a ball at random from the large urn and player 2 records its colour at the second clock time. Let's assume that this ball is green. This process is repeated for T clock times so that player 2

is presented with the entire observation sequence (string of symbols), for example

$$\mathbf{O}_1^T = \{\texttt{red, green, red, red, blue, ...,blue}\}. \tag{B.2}$$

Player 2 must then determine the most probable sequence in which the urns were visited by player 1, in order to explain the observation sequence. This amounts to uncovering the "hidden" part of the model and represents one of the three basic problems that have to be solved, for an HMM to be useful. These problems (and their solutions) are discussed in section B.5.

B.4 Elements of an HMM

The example in the previous section illustrates the process by which a discrete observation HMM with N *states* (each pair of urns constitutes a state) generates a discrete observation sequence. An *initial state distribution* (small detached urn) determines the first state to be visited. At each clock time a new state is entered based upon a *transition probability distribution* (small urn) which depends on the current state. The transition may be such that the process remains in the current state. After each transition is made, an observation symbol is "emitted" according to an *observation probability distribution* (large urn) which depends on the current state. The transition and observation probability distributions are held fixed for the state regardless of when and how the state is entered.

A discrete observation HMM is characterised by the following elements:

- The number of states in the model, N. We denote the individual states as

$$\mathbf{S} = \{s_1, s_2, \ldots, s_N\}, \tag{B.3}$$

  and the state at time t as q_t.

- The alphabet size, M, that is the number of distinct observation symbols per state. We denote the individual symbols as

$$\mathbf{V} = \{v_1, v_2, \ldots, v_M\}. \tag{B.4}$$

- The state transition probability distribution $\mathbf{A} = \{a_{ij}\}$, where

$$a_{ij} = P(q_{t+1} = s_j | q_t = s_i),\ i = 1, \ldots N,\ j = 1, \ldots N. \tag{B.5}$$

- The observation symbol probability distribution $\mathbf{B} = \{b_j(k)\}$, where

$$b_j(k) = P(x_t = v_k | q_t = s_j),\ j = 1, \ldots N,\ k = 1, \ldots M. \tag{B.6}$$

- The initial state distribution $\boldsymbol{\pi} = \{\pi_i\}$, where

$$\pi_i = P(q_1 = s_i),\ i = 1, \ldots N. \tag{B.7}$$

Given appropriate values for N, M, $\mathbf{A}$, $\mathbf{B}$ and $\boldsymbol{\pi}$, the HMM can be used as a generator to give an observation sequence $\mathbf{O}_1^T = \{o_1, o_2, \ldots, o_T\}$ as follows:

1. Choose an initial state $q_1 = s_i$ according to the initial state distribution $\boldsymbol{\pi}$.
2. `Set` $t = 1$.
3. Choose $x_t = v_k$ according to the symbol probability distribution in state s_i, that is $b_i(k)$.
4. Make the transition to a new state $q_{t+1} = s_j$ according to the state transition probability distribution for state s_i, that is a_{ij}.
5. `Set` $t = t + 1$.
6. `If` $t \leq T$
   return to step 3,
   `else`
   terminate procedure.

The above procedure can be used as both a generator of observations, and as a model for how a given observation sequence was generated by an appropriate HMM. For convenience, we use the compact notation,

$$\lambda = (\mathbf{A}, \mathbf{B}, \boldsymbol{\pi}), \tag{B.8}$$

to indicate the complete parameter set for a discrete observation HMM.

The two most popular graphical representations for HMMs is the Moore representation and the time-state representation. In the *Moore representation* the states are indicated by circles and the state transition probabilities by arrows. It is assumed that each state emits a symbol *after* the transition is made to that state (Deller et al. (1999)). Figure B.4 shows a Moore representation of an HMM with five states. This is an ergodic model, since transitions between all the states are allowed. Other topologies are discussed in section B.7.1.

In the *time-state representation* an HMM is represented by a lattice, where each node represents a state at a certain clock time. Two non-emitting states, an initial and terminal state, are often included in this representation. This better illustrates the initial state probabilities and also allows for termination probabilities. For the purposes of this discussion we assume that the termination probabilities are all set to one. Each path through the lattice therefore represents a possible state sequence. Figure B.5 shows a lattice representation of an ergodic HMM with N states.

HMMs can only be used to recognise patterns if three basic problems are solved. In the next section we discuss these problems (and their solutions) for the discrete case. We address the continuous case in section B.6.

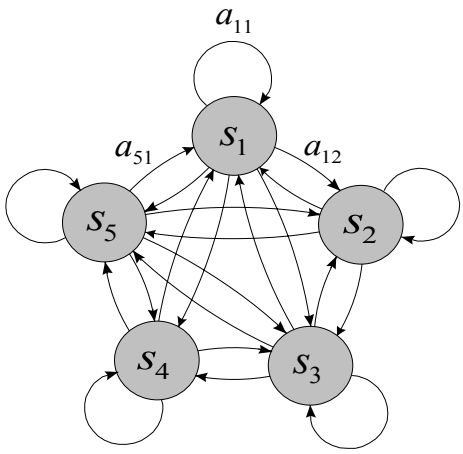


Figure B.4: *A Moore representation of an ergodic HMM with five states.*

B.5 The three basic problems of HMMs

In pattern recognition applications an HMM represents a pattern class, where each pattern is represented by an observation sequence. For an HMM to be useful, one should be able to *train* it with a set of sample (training) patterns, and then *apply* it by matching a test pattern with the trained model. The training and recognition problems are as follows:

Training. Given an initialised HMM and a set of training patterns, one should be able to optimally adjust the HMM parameters so that the *new* HMM better represents the training data.

Recognition. Given a set of trained HMMs, that is HMMs that represents the respective pattern classes, and an observation sequence that represents a test pattern, one should be able to match this observation sequence with every HMM in the set, and then assign the pattern to the model with the highest score (likelihood).

In order to train and apply an HMM, three basic problems have to be solved. We now discuss these problems for the discrete case.

Problem 1: The matching problem. Given a model $\lambda = (\mathbf{A}, \mathbf{B}, \boldsymbol{\pi})$ and an observation sequence $\mathbf{O}_1^T = \{o_1, o_2, \ldots, o_T\}$, calculate the probability that the observation sequence was generated by the model, that is calculate $P(\mathbf{O}_1^T|\lambda)$.

Problem 2: Uncovering the hidden part of the model. Given a model $\lambda = (\mathbf{A}, \mathbf{B}, \boldsymbol{\pi})$ and an observation sequence $\mathbf{O}_1^T = \{o_1, o_2, \ldots, o_T\}$, find the state sequence,

$$\mathbf{Q} = \{q_1, q_2, \ldots, q_T\}, \tag{B.9}$$

that most probably generated these observations.

Problem 3: Reestimating the model parameters. Given a model $\lambda = (\mathbf{A}, \mathbf{B}, \boldsymbol{\pi})$ and an observation sequence $\mathbf{O}_1^T = \{o_1, o_2, \ldots, o_T\}$, find an optimal way to adjust the

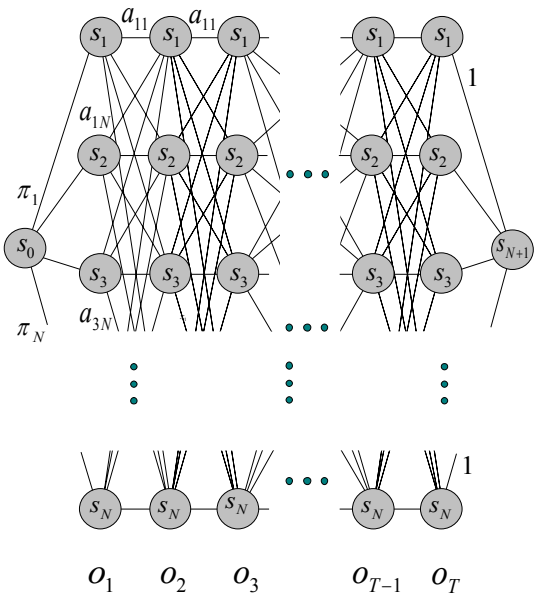


Figure B.5*: A lattice representation of an ergodic HMM with N states.*

model parameters, so as to ensure that the new model, $\lambda^{\triangle} = (\mathbf{A}^{\triangle}, \mathbf{B}^{\triangle}, \boldsymbol{\pi}^{\triangle})$, better matches the observation sequence, that is $P(\mathbf{O}_1^T|\lambda^{\triangle}) \geq P(\mathbf{O}_1^T|\lambda)$.

The solutions to these three basic problems are discussed in the following three subsections.

B.5.1 Problem 1: The matching problem

The straightforward approach. We first consider a naive approach that is computationally very exhaustive. Consider a fixed state sequence $\mathbf{Q}_1^T = \{q_1, q_2, \ldots, q_T\}$, where q_1 is the initial state. When we assume that the observations are statistically independent, the probability for the observation sequence $\mathbf{O}_1^T$, given the state sequence and the model is

$$P(\mathbf{O}_1^T|\mathbf{Q}_1^T, \lambda) = \prod_{t=1}^{T} P(o_t|q_t, \lambda) \tag{B.10}$$

$$= b_{q_1}(o_1)b_{q_2}(o_2)b_{q_3}(o_3)\ldots b_{q_T}(o_T). \tag{B.11}$$

The probability of such a state sequence $\mathbf{Q}_1^T$ is

$$P(\mathbf{Q}_1^T|\lambda) = \pi_{q_1} a_{q_1 q_2} a_{q_2 q_3} \ldots a_{q_{T-1} q_T}. \tag{B.12}$$

The joint probability of $\mathbf{O}_1^T$ and $\mathbf{Q}_1^T$, that is the probability that $\mathbf{O}_1^T$ and $\mathbf{Q}_1^T$ occur simultaneously, is simply the product of the above two terms, that is

$$P(\mathbf{O}_1^T, \mathbf{Q}_1^T|\lambda) = P(\mathbf{O}_1^T|\mathbf{Q}_1^T, \lambda)P(\mathbf{Q}_1^T|\lambda). \tag{B.13}$$

The probability of $\mathbf{O}_1^T$, given the model, is obtained by summing this joint probability over all possible state sequences giving

$$P(\mathbf{O}_1^T|\lambda) = \sum_{\text{All } \mathbf{Q}_1^T} P(\mathbf{O}_1^T|\mathbf{Q}_1^T,\lambda)P(\mathbf{Q}_1^T|\lambda) \tag{B.14}$$

$$= \sum_{\text{All } \mathbf{Q}_1^T} \pi_{q_1} b_{q_1}(o_1) a_{q_1 q_2} b_{q_2}(o_2) a_{q_2 q_3} \dots a_{q_{T-1} q_T} b_{q_T}(o_T). \tag{B.15}$$

The calculation of $P(\mathbf{O}_1^T|\lambda)$, according to this direct definition, involves on the order of $2T \cdot N^T$ calculations. This calculation is not computationally feasible, even for small values of N and T, for example, for $N = 5$ (states) and $T = 100$ (observations), there are on the order of 10^{72} computations. Fortunately a more efficient procedure exists, namely the forward-backward procedure (Baum et al. (1967)).

The forward-backward procedure. We define the forward variable $\alpha_t(i)$ as follows,

$$\alpha_t(i) = P(\mathbf{O}_1^t, q_t = s_i|\lambda), \tag{B.16}$$

that is the probability of the partial observation sequence $\mathbf{O}_1^t$ and state s_i at time t, given the model λ.

We can solve $\alpha_t(i)$ inductively (forward method), as follows:

Initialisation

$$\alpha_1(i) = \pi_i b_i(o_1),\ i = 1, 2, \dots, N \tag{B.17}$$

Induction

$$\alpha_{t+1}(j) = \left[\sum_{i=1}^{N} \alpha_t(i) a_{ij}\right] b_j(o_{t+1}),\ t = 1, 2, \dots, T-1,\ j = 1, 2, \dots, N \tag{B.18}$$

Termination

$$P(\mathbf{O}_1^t|\lambda) = \sum_{i=1}^{N} \alpha_T(i) \tag{B.19}$$

The induction is based on the fact that state s_j can be reached at time $t+1$ from N possible states, $s_i,\ i = 1, 2, \dots, N$, at time t (see figure B.6). The calculation of $P(\mathbf{O}_1^T|\lambda)$, according to this forward method, involves on the order of N^2T calculations. For $N = 5$ (states) and $T = 100$ (observations), the forward method requires about 3000 computations, compared to 10^{72} computations for the straight-forward approach.

Although the forward method efficiently solves the matching problem, the backward method can also be used for this purpose. We introduce the backward method, since both the forward method and the backward method are used to solve the last two problems.

The backward variable $\beta_t(i)$ is defined as

$$\beta_t(i) = P(\mathbf{O}_{t+1}^T, q_t = s_i|\lambda), \tag{B.20}$$

that is the probability of the partial observation sequence from $t+1$ to the end, given state s_i at time t and the model λ.

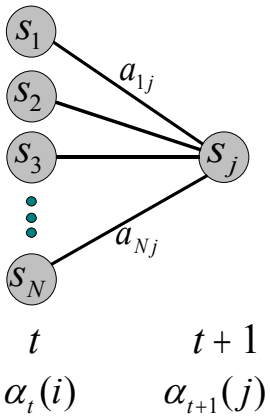


Figure B.6: *Illustration of the sequence of operations required for the computation of the forward variable* $\alpha_{t+1}(j)$.

Again we can solve $\beta_t(i)$ inductively (backward method), as follows:

Initialisation

$$\beta_T(i) = 1,\ i = 1, 2, \ldots, N \tag{B.21}$$

Induction

$$\beta_t(i) = \sum_{i=1}^{N} a_{ij} b_j(o_{t+1}) \beta_{t+1}(j),\ t = T-1, T-2, \ldots, 1,\ i = 1, 2, \ldots, N \tag{B.22}$$

Termination

$$P(\mathbf{O}_1^t|\lambda) = \sum_{i=1}^{N} \beta_1(i) \tag{B.23}$$

Here the induction is based on the fact that transitions to N states, $s_j,\ j = 1, 2, \ldots, N$, at time $t+1$ are possible from state s_i at time t (see figure B.7). The calculation of $P(\mathbf{O}_1^T|\lambda)$, according to the backward method, also involves on the order of N^2T calculations.

Viterbi alignment. The first problem can also be solved by calculating the state sequence $\mathbf{Q}_1^{T*}$ that most probably generated the observation sequence, using the Viterbi algorithm (Forney (1973)). The Viterbi algorithm is discussed in the next section and is based on a dynamic programming technique, similar to the one discussed in section 5.3.2. The probability that the state sequence $\mathbf{Q}_1^{T*}$ and the observation sequence occur simultaneously, given the model, is then calculated as follows

$$P^* = P(\mathbf{O}_1^T, \mathbf{Q}_1^{T*}|\lambda). \tag{B.24}$$

This method gives a good *approximation* to the forward or the backward method.

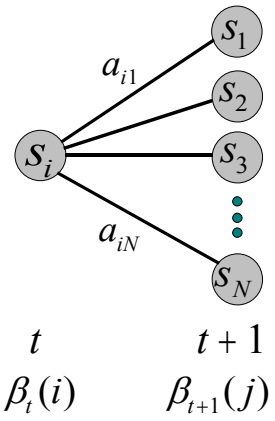


Figure B.7: *Illustration of the sequence of operations required for the computation of the backward variable* $\beta_t(i)$.

B.5.2 Problem 2: Uncovering the hidden part of the model

There is no exact solution to this problem, but two approaches can be taken to find the state sequence that *most probably* generated the observation sequence. One can attempt to find every individual state that most probably generated the observed symbol at a given time and therefore attempt to *maximise the expected number of correct individual states*. This approach is problematic however, since the resulting state sequence may include transitions that are not allowed or are very unlikely. The more popular approach is to calculate the *single best state sequence*, using an algorithm that is based on dynamic programming techniques, namely the Viterbi algorithm.

Maximising the expected number of correct individual states. Although this approach is rarely used to solve the second problem, it does contribute to the solution of the third problem and is therefore briefly discussed here. We define the variable

$$\gamma_t(i) = P(q_t = s_i|\mathbf{O}_1^T, \lambda), \tag{B.25}$$

that is the probability of being in state s_i at time t, given the observation sequence $\mathbf{O}_1^T$ and the model λ. Equation (B.25) can be expressed as follows in terms of the forward-backward variables,

$$\gamma_t(i) = \frac{\alpha_t(i)\beta_t(i)}{P(\mathbf{O}_1^T|\lambda)} = \frac{\alpha_t(i)\beta_t(i)}{\sum_{i=1}^{N}\alpha_t(i)\beta_t(i)}, \tag{B.26}$$

since $\alpha_t(i)$ accounts for the partial observation sequence $\mathbf{O}_1^t$ and the state s_i at t, while $\beta_t(i)$ accounts for the remainder of the observation sequence $\mathbf{O}_{t+1}^T$, given state s_i at t. The normalisation factor $P(\mathbf{O}_1^T|\lambda) = \sum_{i=1}^{N}\alpha_t(i)\beta_t(i)$ makes $\gamma_t(i)$ a probability measure so that

$$\sum_{i=1}^{N}\gamma_t(i) = 1. \tag{B.27}$$

Using $\gamma_t(i)$, we can find the individually most likely state q_t at time t,

$$q_t = \underset{i=1,\ldots,N}{\operatorname{argmax}}[\gamma_t(i)]. \tag{B.28}$$

Finding the best state sequence using the Viterbi algorithm. To find the single best state sequence $\mathbf{Q}_1^{T*} = \{q_1^*, q_2^*, \ldots, q_T^*\}$ for the given observation sequence $\mathbf{O}_1^T = \{o_1, o_2, \ldots, o_T\}$, we need to define the quantity

$$\delta_t(i) = \max_{\text{All } \mathbf{Q}_1^{t-1}} P(q_t = s_i, \mathbf{O}_1^t|\lambda), \tag{B.29}$$

that is the best score along a single path, which accounts for the first t observations, and ends in state s_i. By induction we have that

$$\delta_{t+1}(j) = \max_{i=1,\ldots,N}[\delta_t(i)a_{ij}]b_j(o_{t+1}). \tag{B.30}$$

To retrieve the state sequence, we need to keep track of the argument which maximised (B.30) for each t and j. We do this via the array $\psi_t(j)$.

The complete procedure for finding the best state sequence (Viterbi algorithm) can now be stated as follows:

Initialisation

$$\delta_1(i) = \pi_i b_i(o_1),\ i = 1, 2, \ldots, N \tag{B.31}$$

$$\psi_1(i) = 0 \tag{B.32}$$

Recursion

$$\delta_t(j) = \max_{i=1,\ldots,N}[\delta_{t-1}(i)a_{ij}]b_j(o_t),\ t = 2, \ldots, T,\ j = 1, \ldots N \tag{B.33}$$

$$\psi_t(j) = \underset{i=1,\ldots,N}{\operatorname{argmax}}[\delta_{t-1}(i)a_{ij}],\ t = 2, \ldots, T,\ j = 1, \ldots N \tag{B.34}$$

Termination

$$P^* = \max_{i=1,\ldots,N}[\delta_T(i)] \tag{B.35}$$

$$q_T^* = \underset{i=1,\ldots,N}{\operatorname{argmax}}[\delta_T(i)] \tag{B.36}$$

Backtracking

$$q_t^* = \psi_{t+1}(q_{t+1}^*),\ t = T-1, T-2, \ldots, 1 \tag{B.37}$$

Here $P^* = P(\mathbf{O}_1^T, \mathbf{Q}_1^{T*}|\lambda)$ is an *approximation* to the forward or the backward method, as discussed in the previous section.

B.5.3 Problem 3: Reestimating the model parameters

It is not possible to analytically reestimate the model parameters so as to maximise the probability of the observation sequence given the new model $\lambda^{\triangle}$, that is to maximise $P(X_1^T|\lambda^{\triangle})$. We can however reestimate the model parameters in such a way that the probability of the observation sequence given the new model $P(X_1^T|\lambda^{\triangle})$ is greater than or equal to the probability of the observation sequence given the current model $P(X_1^T|\lambda)$. This implies that an iterative algorithm, that converges to a *local maximum*, can be used. Such an algorithm is the Baum-Welch method.

The Baum-Welch method. In order to describe the Baum-Welch method, we first define $\xi_t(i,j)$ as the joint probability of being in state s_i at time t and in state s_j at time $t+1$, given the model and the observation sequence, that is

$$\xi_t(i,j) = P(q_t = s_i, q_{t+1} = s_j|\mathbf{O}_1^T, \lambda). \tag{B.38}$$

The sequence of events leading to the conditions required by (B.38) is illustrated in figure B.8.

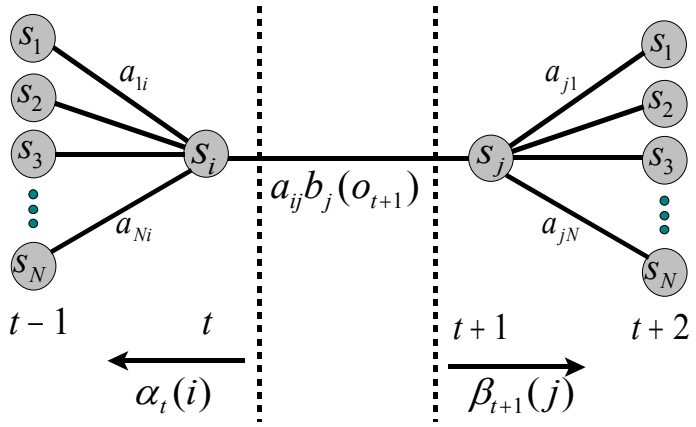


Figure B.8*: Illustration of the sequence of operations required for the computation of the joint event that the system is in state s_i at time t and in state s_j at time $t+1$.*

It should by clear from the definitions of the forward and backward variables, that we can write $\xi_t(i,j)$ in the form,

$$\xi_t(i,j) = \frac{\alpha_t(i)a_{ij}b_j(o_{t+1})\beta_{t+1}(j)}{P(\mathbf{O}_1^T|\lambda)} \tag{B.39}$$

$$= \frac{\alpha_t(i)a_{ij}b_j(o_{t+1})\beta_{t+1}(j)}{\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_t(i)a_{ij}b_j(o_{t+1})\beta_{t+1}(j)}, \tag{B.40}$$

where the numerator term is just $P(q_t = s_i, q_{t+1} = s_j, \mathbf{O}_1^T|\lambda)$ and the division by $P(\mathbf{O}_1^T|\lambda)$ gives the desired probability measure.

We can relate $\gamma_t(i)$ to $\xi_t(i,j)$ by summing over j, giving

$$\gamma_t(i) = \sum_{j=1}^{N} \xi_t(i,j). \tag{B.41}$$

If we sum $\gamma_t(i)$ from $t=1$ to $t=T-1$, we obtain a quantity which can be interpreted as the expected number of times that state s_i is visited, or equivalently, the expected number of transitions made from state s_i. Similarly, summation of $\xi_t(i,j)$ from $t=1$ to $t=T-1$ can be interpreted as the expected number of transitions from state s_i to state s_j, that is

$$\sum_{t=1}^{T-1} \gamma_t(i) = \text{ expected number of transitions from state } s_i, \tag{B.42}$$

$$\sum_{t=1}^{T-1} \xi_t(i,j) = \text{ expected number of transitions from state } s_i \text{ to state } s_j. \tag{B.43}$$

Using the above formulas, the HMM parameters can be reestimated as follows,

$$\pi_i^{\triangle} = \text{ expected number of times in state } s_i \text{ at time } t=1 \tag{B.44}$$

$$= \gamma_1(i), \tag{B.45}$$

$$a_{ij}^{\triangle} = \frac{\text{expected number of transitions from state } s_i \text{ to state } s_j}{\text{expected number of transitions from state } s_i} \tag{B.46}$$

$$= \frac{\sum_{t=1}^{T-1} \xi_t(i,j)}{\sum_{t=1}^{T-1} \gamma_t(i)}, \tag{B.47}$$

$$b_j^{\triangle}(k) = \frac{\text{expected number of times in state } s_j \text{ and observing symbol } v_k}{\text{expected number of times in state } s_j} \tag{B.48}$$

$$= \frac{\sum_{\substack{t=1 \\ x_t = v_k}}^{T-1} \gamma_t(i)}{\sum_{t=1}^{T-1} \gamma_t(i)}. \tag{B.49}$$

If we use the current model $\lambda = (\mathbf{A}, \mathbf{B}, \boldsymbol{\pi})$ to compute the right-hand-sides of equations (B.45) to (B.49) and define the new model as $\lambda^{\triangle} = (\mathbf{A}^{\triangle}, \mathbf{B}^{\triangle}, \boldsymbol{\pi}^{\triangle})$, Baum and Egon (1967) proved that either

- the current model λ defines a critical point of the likelihood function, in which case $\lambda^{\triangle} = \lambda$, or
- model $\lambda^{\triangle}$ is more likely than model λ in the sense that $P(\mathbf{O}_1^T|\lambda^{\triangle}) > P(\mathbf{O}_1^T|\lambda)$.

The reestimation formulas of (B.45) to (B.49) can be derived directly by maximising Baum's auxiliary function,

$$Q(\lambda, \lambda^{\triangle}) = \sum_{\text{All } \mathbf{Q}_1^T} P(\mathbf{Q}_1^T|\mathbf{O}_1^T, \lambda) \log(P(\mathbf{O}_1^T, \mathbf{Q}_1^T|\lambda^{\triangle})), \tag{B.50}$$

over $\lambda^{\triangle}$. Baum and Egon (1967) proved that the maximisation of $Q(\lambda, \lambda^{\triangle})$ leads to an increased likelihood, that is

$$\max_{\lambda^{\triangle}}(Q(\lambda, \lambda^{\triangle})) \;\Rightarrow\; P(\mathbf{O}_1^T|\lambda^{\triangle}) \geq P(\mathbf{O}_1^T|\lambda). \tag{B.51}$$

If we iteratively use $\lambda^{\triangle}$ in place of λ and repeat the reestimation calculation, the likelihood function will therefore converge to a *local* maximum.

Viterbi reestimation. We showed in the previous section how the Viterbi algorithm can be used to find the single best state sequence $\mathbf{Q}_1^{T*}$, that is the state sequence that most probably produced the observation sequence $\mathbf{O}_1^T$, given the model λ. When the states and transitions along $\mathbf{Q}_1^{T*}$ are tallied, it is possible to reestimate the model parameters more efficiently than the Baum-Welch method, without significantly compromising the performance of the model. This is accomplished by reestimating the HMM parameters as follows,

$$a_{ij}^{\triangle} = \frac{\text{number of transitions from state } s_i \text{ to state } s_j \text{ in } \mathbf{Q}_1^{T*}}{\text{number of times state } s_i \text{ occurs in } \mathbf{Q}_1^{T*}}, \tag{B.52}$$

$$b_j^{\triangle}(k) = \frac{\text{number of times state } s_j \text{ and symbol } v_k \text{ occurs simultaneously in } \mathbf{Q}_1^{T*}}{\text{number of times state } s_j \text{ occurs in } \mathbf{Q}_1^{T*}}. \tag{B.53}$$

As is the case with the Baum-Welch method, the iterative application of this method yields a sequence of models for which the likelihood function converges to a *local* maximum.

It should be pointed out that in most problems of interest, the optimisation surface is very complex and has many local maxima. Finding good initial estimates of the HMM parameters is therefore essential (see figure B.9). These and other implementation issues are discussed in section B.7.

So far we only discussed discrete observation HMMs. The three basic problems (and their solutions) for continuous observation HMMs are discussed in the next section.

B.6 Continuous observation HMMs

When the observations are continuous, that is unquantised feature vectors, a multivariate PDF, $f(\mathbf{x}|s_j, \lambda)$, is associated with each state s_j, $j = 1, \ldots N$. The PDF models the distribution of observations within the state. This is in contrast with a discrete observation

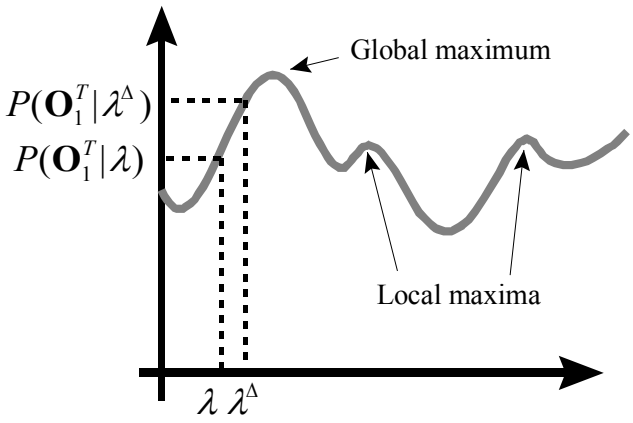


Figure B.9*: Conceptualisation of the HMM likelihood as a function of model parameters.*

HMM where the probability distribution of the discrete observation symbols is modelled by a histogram, $b_j(k),\ k = 1, \ldots M$. For convenience we use the compact notation

$$\lambda = (\mathbf{A}, \{f(\mathbf{x}|s_j), j = 1, \ldots, N\}, \boldsymbol{\pi}), \tag{B.54}$$

to indicate the complete parameter set for a continuous observation HMM.

B.6.1 Recognition

For the first two HMM problems we can simply resort to the use of any of the methods described for the discrete case. The resulting measure computed for a model λ and an observation sequence $\mathbf{X}_1^T = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_T\}$ is $f(\mathbf{X}_1^T|\lambda)$. Note that for the continuous case this measure is a *likelihood* and not a proper probability.

B.6.2 Training

In order to use a continuous observation HMM, some restrictions have to be placed on the form of the PDFs to ensure that the parameters of each PDF can be reestimated in a consistent way. Reestimation formulas have been worked out for a broad class of PDFs. The most widely used member of this class is the Gaussian mixture density, which is of the form,

$$f(\mathbf{x}|s_j) = \sum_{m=1}^{N_m} c_{jm}\eta(\mathbf{x}, \boldsymbol{\mu}_{jm}, \boldsymbol{\Sigma}_{jm}),\ j = 1, \ldots, N, \tag{B.55}$$

where $\mathbf{x}$ is a feature vector, c_{jm} is the mixture coefficient for the mth mixture in state s_j and η is a Gaussian density function, with mean vector $\boldsymbol{\mu}_{jm}$ and covariance matrix

$\mathbf{\Sigma}_{jm}$, for the mth mixture component in state s_j. The mixture gains satisfy the stochastic constraint

$$\sum_{m=1}^{N_m} c_{jm} = 1,\ j = 1, \ldots, N, \tag{B.56}$$

$$c_{jm} \geq 1,\ j = 1, \ldots, N,\ m = 1, \ldots, N_m, \tag{B.57}$$

so that the PDFs are properly normalised, that is

$$\int_{-\infty}^{\infty} f(\mathbf{x}|s_j)\, dx = 1,\ j = 1, \ldots, N. \tag{B.58}$$

Baum-Welch reestimation. It can be shown (Liporace (1982); Juang (1985); Juang et al. (1986)) that the reestimation formulas for the parameters of the mixture density, (B.55), are of the form,

$$c_{jk}^{\triangle} = \frac{\sum_{t=1}^{T} \gamma_t(j,k)}{\sum_{t=1}^{T} \sum_{k=1}^{N_m} \gamma_t(j,k)}, \tag{B.59}$$

$$\boldsymbol{\mu}_{jk}^{\triangle} = \frac{\sum_{t=1}^{T} (\gamma_t(j,k)\mathbf{x}_t)}{\sum_{t=1}^{T} \gamma_t(j,k)}, \tag{B.60}$$

$$\mathbf{\Sigma}_{jk}^{\triangle} = \frac{\sum_{t=1}^{T} \left(\gamma_t(j,k)(\mathbf{x}_t - \boldsymbol{\mu}_{jk})(\mathbf{x}_t - \boldsymbol{\mu}_{jk})'\right)}{\sum_{t=1}^{T} \gamma_t(j,k)}, \tag{B.61}$$

where $'$ denotes the transpose and $\gamma_t(j,k)$ is the probability of being in state s_j at time t with the kth mixture component accounting for x_t, that is

$$\gamma_t(j,k) = \frac{\alpha_t(j)\beta_t(j)}{\sum_{j=1}^{N} \alpha_t(j)\beta_t(j)} \times \frac{c_{jk}\eta(\mathbf{x}_t, \boldsymbol{\mu}_{jk}, \mathbf{\Sigma}_{jk})}{\sum_{m=1}^{N_m} c_{jk}\eta(\mathbf{x}_t, \boldsymbol{\mu}_{jk}, \mathbf{\Sigma}_{jk})}. \tag{B.62}$$

The reestimation formula for a_{ij} is identical to the one used for discrete observation densities, that is (B.47).

The formulas, (B.59) to (B.61), can be interpreted as follows. The reestimation formula for c_{jk} is the ratio between the expected number of times the system is in state s_j using the kth mixture component, and the expected number of times the system is in state s_j. Similarly the reestimation formula for the mean vector $\boldsymbol{\mu}_{jk}$ weighs each numerator term of (B.59) by the observation, thereby giving the expected value of the portion of the observation vector accounted for by the kth mixture component. A similar interpretation can be given for the reestimation term for the covariance matrix $\mathbf{\Sigma}_{jk}$.

Viterbi reestimation. If the Viterbi approach is used, the mean vectors and covariance matrices for the observation densities are reestimated by simple averaging. This is easily described when there is only one mixture component per state. In this case, for a given λ, the single best state sequence, that is the state sequence that most probably produced the observations, is calculated, using the Viterbi algorithm. Each observation vector is then assigned to the state that produced it on the optimal path by examining the backtracking information. This amounts to the segmentation of the feature space. When we use "$\mathbf{x}_t :\Rightarrow s_i$" to denote the fact that observation $\mathbf{x}_t$ is assigned to state s_i, and assume that N_i of the T observation vectors are assigned to this state, each mean vector and covariance matrix is reestimated as follows,

$$\boldsymbol{\mu}_i^{\triangle} = \frac{1}{N_i} \sum_{\substack{t=1 \\ \mathbf{x}_t :\Rightarrow s_i}}^{T} \mathbf{x}_t, \tag{B.63}$$

$$\boldsymbol{\Sigma}_i^{\triangle} = \frac{1}{N_i} \sum_{\substack{t=1 \\ \mathbf{x}_t :\Rightarrow s_i}}^{T} (\mathbf{x}_t - \boldsymbol{\mu}_i)(\mathbf{x}_t - \boldsymbol{\mu}_i)'. \tag{B.64}$$

When $N_m > 1$ mixture components appear in a state, the observation vectors assigned to that state must be subdivided into N_m subsets, prior to averaging. This can be done by clustering, using the K-means algorithm, for example. If N_{il} vectors are assigned to the lth mixture in state s_i, the reestimation formula for the mixture coefficient $c_{il}^{\triangle}$ is as follows,

$$c_{il}^{\triangle} = \frac{N_{il}}{N_i}. \tag{B.65}$$

B.7 Implementation issues

We conclude this chapter with a brief discussion of a few remaining practical implementation issues.

B.7.1 HMM topology

In the previous sections, we only discussed fully connected HMMs in which every state in the model can be reached in a single step from every other state. These models are called ergodic HMMs (see figures B.4 and B.5).

For certain applications, other types of HMMs have been found to account for the observed properties of the signal (that is being modelled) better than the standard ergodic model. HMMs that are ideal for modelling signals of which the properties change over time, are the so-called left-to-right models. These models are popular for modelling handwriting and speech signals (see figure B.10).

Although the two basic topologies for HMMs are ergodic and left-to-right, many possible variations and combinations are possible.

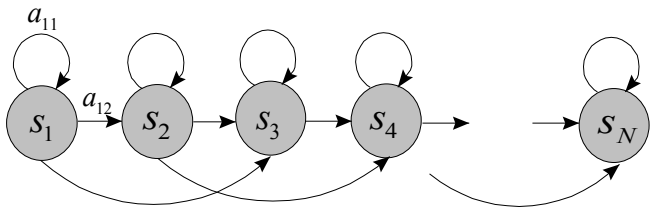


Figure B.10*: A left-to-right HMM with N states and one state skip.*

B.7.2 Scaling

Baum-Welch reestimation and the forward-backward procedure. The Baum-Welch reestimation technique and the forward-backward procedure require that the forward and backward variables, that is $\alpha_t(i)$ (B.16) and $\beta_t(i)$ (B.20), are calculated. When these variables are calculated by recursion, products of probabilities are accumulated at each time step. This results in a sequence that goes exponentially to zero and numerical underflow problems result. This problem can be solved through the scaling of the parameters. We do not discuss the scaling procedure here and the reader is referred to Levinson et al. (1983) and Rabiner (1989).

Viterbi reestimation and matching. No scaling is required when logarithms are used in the following way. For a discrete observation HMM, we first define

$$\phi_t(i) = \max_{\text{All } \mathbf{Q}_1^t} \ln\{P(\mathbf{Q}_1^t, \mathbf{O}_1^t|\lambda)\}, \tag{B.66}$$

and then calculate $\ln P^*$ as follows:

Initialisation

$$\phi_1(i) = \ln\{\pi_i\} + \ln\{b_i(o_1)\},\ i = 1, 2, \ldots, N \tag{B.67}$$

Recursion

$$\phi_t(j) = \max_{i=1,\ldots,N}[\phi_{t-1} + \ln a_{ij}] + \ln\{b_j(o_t)\},\ t = 2, \ldots, T,\ j = 1, \ldots N \tag{B.68}$$

Termination

$$\ln P^* = \max_{i=1,\ldots,N}[\phi_T(i)] \tag{B.69}$$

B.7.3 Training with multiple observation sequences

In order to provide a more complete representation of the statistical variations likely to be present across more than one observation sequence, it is essential to train a given HMM with multiple training sequences. The modification of the reestimation procedure

is straightforward and is a follows. For a discrete observation HMM, we denote a set of Υ observation sequences as

$$\mathbf{O} = \{\mathbf{O}^{(1)}, \mathbf{O}^{(2)}, \mathbf{O}^{(3)}, \ldots, \mathbf{O}^{(\Upsilon)}\}, \tag{B.70}$$

where $\mathbf{O}^{(k)} = \mathbf{O}_1^{T(k)}$ is the kth observation sequence. We assume that each observation sequence is independent of every other observation sequence, and the goal is to adjust the parameters of the model λ, in order to maximise

$$P(\mathbf{O}|\lambda) = \prod_{k=1}^{\Upsilon} P(\mathbf{O}^{(k)}|\lambda) \tag{B.71}$$

$$= \prod_{k=1}^{\Upsilon} P_k. \tag{B.72}$$

Baum-Welch reestimation. Here we assume that we are working with a left-to-right model so that there is no need for reestimating the initial state probabilities. Since the reestimation formulas are based on frequencies of occurrence of various events, the reestimation formulas for multiple observation sequences are modified by adding the individual frequencies of occurrence for each sequence. Thus the modified reestimation formulas for $a_{ij}^{\triangle}$ and $b_j^{\triangle}(l)$ are

$$a_{ij}^{\triangle} = \frac{\sum_{k=1}^{\Upsilon} \frac{1}{P_k} \sum_{t=1}^{T_k-1} \alpha_t^k(i) a_{ij} b_j(o_{t+1}^{(k)}) \beta_{t+1}^k(j)}{\sum_{k=1}^{\Upsilon} \frac{1}{P_k} \sum_{t=1}^{T_k-1} \alpha_t^k(i) \beta_t^k(i)} \tag{B.73}$$

and

$$b_j^{\triangle}(l) = \frac{\sum_{k=1}^{\Upsilon} \frac{1}{P_k} \sum_{\substack{t=1 \\ x_t = v_l}}^{T_k-1} \alpha_t^k(i) \beta_t^k(i)}{\sum_{k=1}^{\Upsilon} \frac{1}{P_k} \sum_{t=1}^{T_k-1} \alpha_t^k(i) \beta_t^k(i)}. \tag{B.74}$$

Viterbi reestimation. If the Viterbi approach is used, optimal paths are obtained for each of the observation sequences. These paths assign a state to every observation and the model parameters are adjusted using the formulas (B.52), (B.53), (B.63), (B.64) and (B.65).

B.7.4 Initial estimates of HMM parameters

In theory, the reestimation equations should give values of the HMM parameters that correspond to a *local* maximum of the likelihood function (see figure B.9). A key question is therefore: how do we choose initial estimates of the HMM parameters so that the local maximum is also the *global* maximum of the likelihood function?

Basically there is no simple answer to the above question. Instead, experience has shown that either random or uniform estimates of the $\boldsymbol{\pi}$ and $\mathbf{A}$ parameters are adequate for giving useful reestimates of these parameters in almost all cases. However, for the $\mathbf{B}$ parameters, experience has shown that good initial estimates are helpful in the discrete case, and are *essential* in the continuous case. Such initial estimates can be obtained in a number of ways, including manual segmentation of the observation sequences into states with averaging of observations within states, maximum likelihood segmentation of observations with averaging, segmental K-means segmentation with clustering, etc.