THE UNIVERSITY OF
WAIKATO
Te Whare Wānanga o Waikato

Off-line Signature Verification

Robert L. Larkins

This thesis is submitted in partial fulfillment of the requirements for the Degree of Master of Science at the University of Waikato.

March 2009

Abstract

In today's society signatures are the most accepted form of identity verification. However, they have the unfortunate side-effect of being easily abused by those who would feign the identification or intent of an individual. This thesis implements and tests current approaches to off-line signature verification with the goal of determining the most beneficial techniques that are available. This investigation will also introduce novel techniques that are shown to significantly boost the achieved classification accuracy for both person-dependent (one-class training) and person-independent (two-class training) signature verification learning strategies.

The findings presented in this thesis show that many common techniques do not always give any significant advantage and in some cases they actually detract from the classification accuracy. Using the techniques that are proven to be most beneficial, an effective approach to signature verification is constructed, which achieves approximately 90% and 91% on the standard CEDAR and GPDS signature datasets respectively. These results are significantly better than the majority of results that have been previously published. Additionally, this approach is shown to remain relatively stable when a minimal number of training signatures are used, representing feasibility for real-world situations.

Acknowledgements

Throughout this thesis I had the good fortune of working with like minded people who shared an interest in both computer science and specifically computer vision. Foremost I would like to thank my supervisor Mike Mayo who gave me ideas and inspiration when I was unsure how to proceed. His patience in going over both my conference paper and thesis numerous times was much appreciated.

To James Foulds, Edmond Zhang and Jesse Read who I worked along side in the Machine Learning Lab, I thank you. You guys made my voyage through Masters so much more enjoyable, for being on hand to answer my multitude of questions and especially for our many adventures both in rain or shine to the bakery.

I would also like to thank my family and friends, who while not always understanding what I tried to explain always took an interest and encouraged me. A special "thank you" goes to my partner Michelle McGonigal, who put up with me disappearing in the evenings and weekends to complete this thesis, and was tolerant and supportive at the most stressful of moments.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

This thesis puts to the test automatic methods for hand-written signature verification. Signature verification is a necessary task in society, as signatures have a well established and accepted place as a formal means of personal verification. Due to this, they are used in government, legal and commercial transactions, and they are the most accepted method of identity verification (Jain et al., 2004).

An inevitable side-effect of signatures is that they can be exploited for the purpose of feigning a document's authenticity. Because of this, the identification of forgeries is one of the oldest forms of forensic science, with references being made to it in Roman Law as far back as the 3rd Century AD, and forgery being a statutory offence in Britain in the 13th Century (Lutterbeck et al., 2000; White, 2004).

Even today, signature forgeries are still a common problem, with the total global cost to businesses, financial institutions and individuals being untold. The Federal Trade Commission estimated that in 2002, identity theft (in which signature forgeries are an attributing factor) caused $53 billion in total losses to the United States alone (Zimmerman et al., 2004) – and is only the tip of the iceberg.

Signature verification reduces the risk of a forged signature being accepted as a genuine. The problem that arises though, is that validation of the signature often needs to be carried out immediately, so that completion of the intended process can be permitted. This commonly occurs when dealing with monetary transactions, especially via credit cards and bank cheques (Ma et al., 2007; Jain et al., 2002). The types of automatic signature verification considered in this thesis are suitable

for alleviating this problem, as a human examiner need only be consulted when the authenticity of a signature is disputed (White, 2004).

1.1 Signatures as a Biometric

Signatures are a behavioural biometric, which like writing, walking and talking, is a trait that is learnt over time. Behavioural biometrics differ from physiological biometrics as they measure a person's physical features, such as the face, fingerprint or retina (Al-Shoshan, 2006), and tend to remain relatively stable over time. The behavioural characteristics of a person tend to change over both the short and long terms due to health, physical state and ageing, which makes them more difficult to distinguish than the physical characteristics. This is especially true for signatures, which can also vary depending on the signatory's mental state, fatigue, and their writing position (Zimmerman et al., 2004). As a result of this high variability in the signing process, the verification of a signature is not a trivial task, for both human experts, and especially computers, which is the focus of this thesis (Impedovo and Pirlo, 2007).

1.2 Verification Methods

Currently, institutes that rely on signatures do not actively check the authenticity of each signature. Instead, in place measures are relied upon that attempt to restrict the occurrence of forgeries, with any suspect signatures requiring visual confirmation in all instances (Judd, 2008). The introduction of automatic procedures have the potential of further reducing the possibility of forgeries slipping through these measures.

In signature verification, there are two overall methods that are employed for determining whether a signature is a genuine or a forgery. The first and most common method is manual verification, in which a human determines the validity of a signature. The second method involves the use of a computer for automatic

verification, and is becoming more practicable as research advances.

1.2.1 Human Forensic Document Examination

Document examination is a forensic science that has a number of areas of expertise, which include the identification of handwriting and signatures, and the composition of inks, papers and materials from which documents are produced. A specialist in this field is known as a forensic document examiner, an occupation that requires an extensive amount of training in order to become proficient. In the United Kingdom, the majority of forensic document examiners are employed by government laboratories. When graduate scientists join these laboratories, they will first work for up to two years alongside highly experienced document examiners before handling their own case work (White, 2004). For a properly trained examiner, the examination of a document is not carried out by merely looking at the handwriting, but also requires an array of equipment and techniques that are developed to extract as much information as possible from the document without damaging or altering it. The examiner, when studying a signature, will also use a chart of elemental characteristics such as ticks, smoothness of curves, pressure changes, spacing and slant to help identify the signature (Srihari et al., 2008).

1.2.2 Automatic Verification

Automatic signature verification is a method that as of yet has not gained wide acceptance, but could be used as a substitute for or a tool used by a forensic document examiner. This is because the feasibility of using automatic methods is more suited for activities such as boarding an aircraft, entering a secure physical location and performing financial transactions (Zimmerman et al., 2004). For these activities, verification of the signature class is determined algorithmically by going through a sequence of steps. These steps transform the signature from a piece of writing on a document into a set of features that can be used to classify the signature. These features differ significantly from the ones used in the manual

method. This is because a computer can not visually analyse or comprehend the entirety of a signature the same way that a human can. Instead, a computer deals with a signature at the pixel level, with features being produced from the pixels and how these pixels relate to each other. The methods used in automatic verification produce a range of results that can vary based on a number of factors related to the choice of algorithms. Currently, the best classification accuracies being achieved are approximately 90% – 92% on a database of 2,640 signatures for 55 subjects (Section 15.3).

1.3 Learning Strategies

In signature verification, there are often two learning strategies that are used to determine a signature's class (Srihari et al., 2004, 2008). These learning strategies are "person-dependent" and "person-independent".

The person-dependent approach involves building a classifier using genuine signatures from a single person. Essentially it measures the distance in feature space between all genuine signatures taken from this person. These distances then constitute the distribution of the genuine signatures in feature space. This distribution can then be used to calculate a probability or similarity by finding the distance between an unknown signature and each genuine. If this similarity is within the bounds that is set by the known distribution, then the unknown signature is considered to be a genuine, otherwise it is classified as a forgery.

Person-independent is a more general approach to signature verification, and accounts for all varieties of verification that involves multiple classes. It differs from person-dependent in the fact that the classifier is built from two or more classes, where these classes can be either genuine or forgeries from as many people as desired. This approach at the basic level is a nearest neighbour approach that calculates a likelihood ratio of an unknown signature belonging to a particular class. If the likelihood is above a certain threshold, then the signature is deemed to be a genuine, otherwise it is considered to be a forgery.

The research that was carried out in this thesis primarily focused on the person-dependent approach, though the effect that machine learning has on the person-independent approach was investigated.

1.4 Signature Capture

Signature capture is the first step required in being able to determine a signature's authenticity, that is, whether it is genuine or a forgery. Capturing a signature is the process of obtaining information about the signature, which can include both spatial and dynamic information. This information is turned into a form that can be interpreted and processed by a computer. The spatial information is the visual aspects of a signature, such as its shape and size, while the dynamic information relates to the writing process of the signature and defines aspects such as the speed and pressure used. Once this has been accomplished, the defining features can then be extracted from the signature. This section looks at the two data acquisition methods used for capturing written signatures and the data that is extracted from them; these are the dynamic (on-line) and static (off-line) methods. The remaining chapters will be focusing on static signatures and the techniques that are related to off-line signature verification.

1.4.1 Dynamic Signatures

Dynamic signatures capture both the spacial and the dynamic information of a signature. The dynamic features that are captured include one or more of the following attributes as the signature is written: acceleration and velocity; the position of the pen; the pressure that is applied; and the pen inclination (Al-Shoshan, 2006; Impedovo and Pirlo, 2007). Dynamic information is not limited to these factors as it can include any information that is relevant to the writing process. The capture of a signature's dynamic information requires the use of specialised hardware such as a digitising tablet (Jain et al., 2002), or an electronic pen, that capture the written attributes and converts them into processable data.

The advantage of using dynamic signatures for verification is that they produce a greater number of defining features than their static counterparts, making forged signatures easier to detect. The added advantage of digitally capturing this data is that there is typically less pre-processing required than static signatures, making feature extraction easier.

Using this method to capture the writing process also has its associated disadvantages. The main disadvantage is that these tools are not very common and as such, cannot be utilised for every situation where signature verification is required. As well as this, their use is unnatural for the user, which could negatively affect the writing of the signature. More recent approaches exploit the use of a video camera which is focused on the writing of the signature and can be carried out using ordinary pen and paper. This allows the handwriting to be recovered from its spatio-temporal representation, which is given by the sequence of images that are produced (Impedovo and Pirlo, 2007).

1.4.2 Static Signatures

A static signature is the visual representation of a signature that has been written out in its entirety. Because of this, a static signature is only captured once the writing process has been completed, allowing for visual aspects such as the size, slope and curvature to be extracted. It is these visual aspects and their derivatives that allow similar signatures to be grouped together and differentiated from others that do not belong in the same group. A signature is generally written on paper using a standard pen, meaning that the signature is not initially in a digital format. Because of this, the signature has to be captured and transformed into a format that can be processed by a computer. The conversion of a signature to a digital format is carried out either by taking a photograph or scanning the signature (Kalera et al., 2004). Both of these methods produce a digital image from which defining features of the signature can then be extracted.

In society, a signed hardcopy of a document is the general requirement for binding that document to the signatory. For this purpose, a digitally captured signature

is not always sufficient. As well as this, static signatures are much more prevalent than digitising tablets and electronic pens, as they only require a standard pen and paper. The use of static signatures does pose some disadvantages for automatic verification though. One disadvantage is the fact that once a signature has been captured, it will usually require pre-processing in order to remove unwanted artefacts, and normalise the size and binarisation of the signature image. Another disadvantage is that static signatures do not produce as much data as dynamic signatures, causing the verification of the signature's validity to be based solely on the extracted features.

1.5 Forgery Types

In signature verification, forged signatures can be broken up into three different categories. These categories are based on how similar a forgery is in relation to the genuine signature and are known as random, simple and skilled (Justino et al., 2001; Zhang, 2006). A random forgery is one in which the forger does not know the signer's name or signature shape. A simple forgery is produced knowing the name of the original signer but not what their signature looks like. A skilled forgery is a close imitation of the genuine signature and is produced by a forger who has seen and practised writing the genuine signature. It is these skilled forgeries that this thesis will focus on for signature verification.

1.6 Thesis Summary

Automatic verification of off-line signatures is not a new task, with many approaches having been produced. Unfortunately these approaches are under-utilised as they do not achieve a classification accuracy that is acceptable in practise. In this thesis each step of the verification process will be explored to determine if the current state-of-the-art techniques can be improved upon or simplified. The goal of this is to determine what combination of techniques has the greatest effect on

classification accuracy, as well as attempting to automate each step as effectively as possible. This thesis has organised each step of the classification process into individual chapters, starting with the background of the current state-of-the-art approaches in Chapter 2. The process that is followed through for classifying an unknown signature has three major steps, preprocessing, feature extraction and classification.

Chapter 3 covers the different preprocessing techniques, while Chapter 4 describes the feature extraction. Chapters 5 and 6 investigate the addition of spatial information via the use of region sampling and spatial pyramids respectively. Having extracted the features, the manner in which they are represented is described in Chapter 7. Chapter 8 explores different techniques for converting between these representations, and also introduces a new and novel approach termed adaptive feature thresholding (Larkins and Mayo, 2008) which improves upon all previous methods.

For the classification of signatures there were two approaches tested, one and two class classification. One class classification is described in Chapter 9 and is based on the person-dependent learning strategy, while Chapter 10 investigates the use of machine learning techniques for the person-independent approach.

Each of these steps are then evaluated using two datasets which are outlined in Chapter 11. The evaluation is divided across Chapters 12, 13, 14 and 15, where the first three chapters evaluate the effectiveness of the preprocessing, feature extraction and classification steps. The fourth classification chapter summarises this evaluation and determines the ability of the tried methods in relation to the current state-of-the-art approaches. The thesis is then concluded in Chapter 16.

Chapter 2

Background

Off-line signature verification is a well established field of research, with a large variety of literature having been written; in regards to this, both Sabourin (1997) and Impedovo and Pirlo (2007) provide literature reviews on the topic. This chapter will briefly outline some of the more recent work that has been carried out, as well as describing in more detail the methods that can be compared against. The methods that can be compared against use one of two datasets (CEDAR and GPDS), which have started to see wider acceptance for gauging classification ability. Section 15.3 compares the approaches that use these two datasets with those implemented in this thesis.

2.1 One Class Classification

The one class approach that was implemented in this thesis is based largely on methodologies described in previous literature, with its overall structure being based on the approach that Kalera et al. (2004) introduced. This section will describe in detail the approaches that can be compared against and will then briefly outline other published approaches.

The approach by Kalera et al. (2004) begins by preprocessing each signature. This converts them from grey-scale to binary as well as performing rotation normalisation. The signatures are then uniquely defined by performing feature extraction using the GSC ensemble (Favata and Srikantan, 1996), which is a collection of five

features designed for hand-written text, and described in Section 4.8. The use of a 4×4 equimass grid (Section 5.2.1) allows spatial information to be added to the features. The frequency vector that each feature creates is then converted into a binary vector using a threshold. Most literature that uses the GSC ensemble does not discuss how these thresholds are chosen, but are generally found through manual trial and error testing (Favata, 2008).

Having constructed a binary feature vector for each signature, classification was tested using the CEDAR dataset (Section 11.1), which took 16 of the 24 genuine signatures per subject for training, while the remaining 8 genuine and 24 forgery signatures were used for testing. The verification of each test signature was carried out by finding the mean similarity between it and every training signature. The threshold that is used to determine whether it is genuine or forgery is found by first calculating the mean similarity between each training signature, the threshold is then found by subtracting a manually chosen offset from the mean similarity value. The final classification accuracy of 78.1% was found as the equal error rate for the entire dataset.

Subsequent literature increased the classification accuracy by making modifications to this approach. Chen and Srihari (2005) changed the method of preprocessing by keeping the original orientation of the image, but carried out broken stroke connection instead. Broken stroke connection attempts to rebuild parts of the signature that have been lost in either the writing or capturing stages. Feature extraction did not use the GSC ensemble, but instead extracted the contour of the signature. The contour was then used in conjunction with Zernike moments to produce values that represent the image. The classification of each signature was then carried out in the same manner as the previously described approach, and achieved an accuracy of 83.6%.

Chen and Srihari (2006) build upon these two previous approaches by introducing a novel method that uses graph matching. This method begins by finding and extracting the extremas from each signature's contour. These extremas are then combined with thin-plate spline mapping to deform the region sampling grid,

allowing each region to more accurately capture the same structural components between signatures. Having produced an alternative region sampling grid, the GSC ensemble is used to extract the features and the classification is carried out using the same approach as Kalera et al. (2004), and attained an accuracy of 92.2%. This method was not tested in this thesis as not enough detail was provided to replicate the algorithm. Additionally, any attempts to contact the authors in regards to this work was in vain due to their lack of response.

The GPDS dataset (Section 11.2) was tested using an alternative approach, in which Tian et al. (2007) introduces a scheme that performs verification using Discrete Wavelet Transform (DWT) and fuzzy nets. To begin, each signature was preprocessed to a dimension of 60 by 120, and then thinned using the SPTA thinning algorithm. Following this each signature had five distinct features extracted. Moment features were found as global characteristics, while the four remaining features used a 4×4 regioning grid. These four features were pixel density (mass), angle features, gravity centre distance and predominant slant, each of which are used to generate a feature vector as input for the DWT.

The DWT decomposes these features into lowpass and highpass information, where only the highpass is kept as it represents the features that contain sharper variations in the time domain. The classification is carried out by building a fuzzy net from the training signatures, where if the input signal of an unknown signature is genuine, the error between it and the fuzzy net is small, otherwise if the error is too large, the signature is deemed to be a forgery. The experiments were carried out by randomly selecting 12 genuine signatures to train with, using the remaining genuine and forgeries for testing. An additional system of selecting preferred training signatures was tested, with classification also being tried with hidden Markov models, support vector machines and Euclidean distance. The optimal method that was presented achieved 87.4%.

A variety of additional approaches have been tested for signature verification in the person-dependent or one class domain that do not utilise the CEDAR or GPDS datasets. Deng et al. (1999) proposed a wavelet system based on the contours of

the signatures that accomplished 92.6%, while Fang et al. (2003) investigated and attempted to track variations between genuine signatures, and to utilise these variations for determining whether an unknown signature was genuine; this approach achieved 81.9%. The kernel principal component self-regression model proposed by Zhang (2006) attained 96% by utilising principal components from kernel space to characterise the signatures from a selected individual. Mizukami et al. (2002) reached 75.1% using the displacement extraction method, where the sum of the squared Euclidean distance between signatures was used for the classification. Another approach was tested by Majhi et al. (2006) who achieved 84.5% when using the geometric centre of a signature to find feature points to distinguish between signatures.

2.2 Two Class Classification

The use of two class classification methods for signature verification has increased as machine learning techniques have become more established. This section will describe in detail the approaches that can be compared with, followed by a brief outline of techniques that can not be contrasted against.

The person-independent or two class methodology has been applied to the CEDAR dataset using a number of approaches. Typically the dataset is preprocessed in the same manner as the person-dependent approach, with the same or similar features being extracted. The difference lies in the fact that an additional class is used and different classification methods can therefore be utilised. Srihari et al. (2004) begins by preprocessing each signature by performing salt-and-pepper noise removal, followed by slant normalisation. The GSC ensemble of features is then extracted. The training of both a naïve Bayes classifier and a distance statistics method used 16 genuine and 16 forgery signatures, where the genuine class was trained using genuine-genuine pairs and the forgery class used genuine-forgery pairs. The similarity between each pair of signatures was found using the correlation distance. Additionally to this, the distance statistics method, naïve Bayes

and support vector machines were used to test the classification accuracy when 16 genuine signatures and 5 forgeries were used for training. The support vector machine achieved 90.7%, the highest accuracy of the three.

Building on this previous work, Srihari et al. (2008) tests a range of machine learning algorithms in combination with both person-dependent and person-independent classification. The preprocessing and feature extraction was carried out using the same procedure as described previously. The classification of unknown signatures was tested using one of six classifiers, where the number of genuine training signatures was varied between 5 and 20. These classifiers are Kolnogorov-Smirnov (KS), Kullback-Leibler (KL), reverse KL, symmetrised KL, Jensen-Shannon and the combination of KS and KL. The KS and KL combination attained the highest accuracy at 82.44%. Though the use of a person-independent approach is mentioned throughout this paper, there is no description to the number of forgery signatures that are used for training.

Armand et al. (2006) explores the effect of using a person-independent learning strategy with the GPDS dataset, where a combination of features are used to differentiate between signatures. The modified direction feature is the primary feature used, and is used either independently, or in combination with one or more of the following features: centroid, triSurface and length feature. Once the features are extracted, two neural networks, resilient backpropagation and radial basis function, are used to classify the signatures. These two classifiers were then trained using 18 genuine and 22 forgery signatures from each of the 39 signature sets, resulting in 1560 signatures being used to train with and 546 signatures being used for testing. To ensure that the final result was resilient, four fold cross validation was employed. The combination of all of these features with the radial basis function achieved the highest accuracy at 91.1%.

Justino et al. (2001) achieved 87.3% when hidden Markov models in conjunction with extended-shadow-codes were tested together, while Coetzer et al. (2004) gained 82% when hidden Markov models in combination with the discrete radon transform were tested. Justino et al. (2005) compares the effectiveness of support

vector machines and hidden Markov models in relation to off-line signature verification; the comparison concluded that support vector machines showed better ability with a result of 91.0%. Support vector machines were also used by Özgündüz et al. (2005) with a variety of extracted features, where it is shown that they outperform artificial neural networks by 12.5%, attaining 93.5%. Zhang et al. (2007) accomplished 98.2% with a novel method of verification using one-class-one-network classification, which is a fixed-size neural-network-based classifier. Ma et al. (2007) achieved 96.8% by expanding on the one-class-one-network approach by incorporating it with adaptive multi-resolution wavelets from which zero-crossing features were extracted. The combination of different support vector machines in the receiver operating characteristic space is presented by Oliveira et al. (2008) and is shown to reduce the false rejection rate while keeping the false acceptance rate at acceptable levels, from which a 91.8% accuracy is gained.

Chapter 3

Signature Preprocessing

Once a signature has been captured from paper and turned into a digital image, defining features about the signature can then be extracted. The disadvantage of extracting features from a signature straight after it has been captured is that the capturing process may have distorted the signature. The result of this is that the extracted image features could potentially be misconstrued, which could adversely affect the classification capability. This is where preprocessing comes in, as it modifies the signature to help improve the representation of extracted image features by cleaning and repairing the signature's structure. By doing this, the signature image should more accurately represent what the signatory intended it to look like. This chapter will explain the different methods of preprocessing that were explored.

3.1 Binarisation

When a signature is captured from paper, the device that was used and how it was set up determines what the colour space of the signature image will be. This colour space is usually grey-scale, Figure 3.1 shows an example of this. Binarisation simplifies the image by converting it to be pure black and white pixels, with the black pixels making up the signature and the background being filled by the white pixels. By using a binarised image, the complexity of both the computational performance and the extraction of image features is decreased as the variability of each pixel value is also decreased. This is because each pixel in a binary image has

$2^1 = 2$ possible values, while grey-scale has $2^8 = 256$. This section will describe different binarisation algorithms, which are of the global or the locally adaptive variety. Global calculates a single threshold that is then applied to every pixel, while locally adaptive generates a threshold for each pixel based on its neighbours.

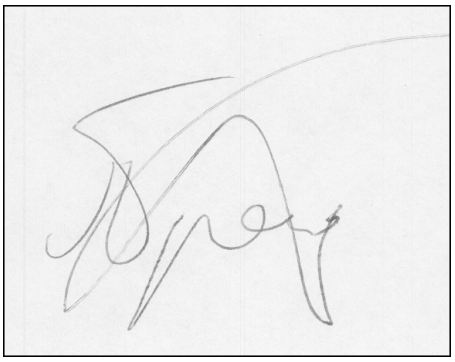
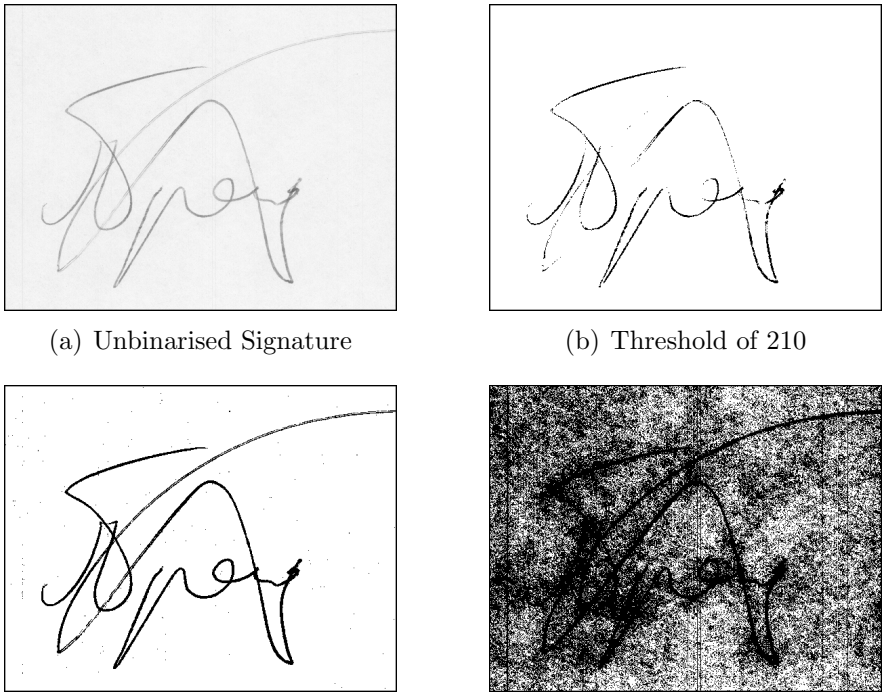


Figure 3.1: An unbinarised signature

3.1.1 Global Threshold

The simplest method of global binarisation is to use a fixed threshold that is manually chosen. Each pixel in the image is then independently converted to black or white depending on its value in relation to the threshold, where if the value is below the threshold, the pixel becomes black and is part of the signature, otherwise the pixel is set to white and is part of the background. The effectiveness of using this method in accurately thresholding an image is dependent upon the chosen value for the threshold. Figure 3.2 shows how three different thresholds affect image binarisation in relation to the unbinarised signature.

The primary advantage of using a fixed threshold for binarisation is its simplicity, as it only requires one pass over the signature. Outside of this, using a fixed threshold does have the disadvantage of not being able to adapt to the image. Therefore, what may work well for one image could negatively impact the binarisation of another. This is because a threshold that is at a too high a level allows background clutter to be included, while a threshold at too low a level results in a loss of information. Another disadvantage is the fact that the threshold is chosen



(a) Unbinarised Signature

(b) Threshold of 210

(c) Threshold of 240

(d) Threshold of 248

Figure 3.2: Signature binarisation with different levels of thresholding

manually, meaning that the algorithm needs human intervention for calibration, removing the fully automatic aspect which is often required.

3.1.2 Iterative Threshold

Iterative thresholding (Ridler and Calvard, 1978) is an automatic method for finding a global threshold through the use of the grey level histogram of the image. If two distinct peaks exist in the histogram, then one peak generally corresponds to the signature and the other to the background. Iterative thresholding attempts to find the two mean points of these peaks and from here, the optimum divergence between them. In signature binarisation, where a large portion of the image is background, only one distinct peak may exist. In this case, a divergence can still be found, as a minor peak should still exist. Figure 3.3 shows a generated histogram of pixel frequencies for the unbinarised signature in Figure 3.1. Because the minor peak is substantially smaller than the major peak, it has been emphasised by using the log of the frequencies. The dashed line shows where the final threshold value

will lie.

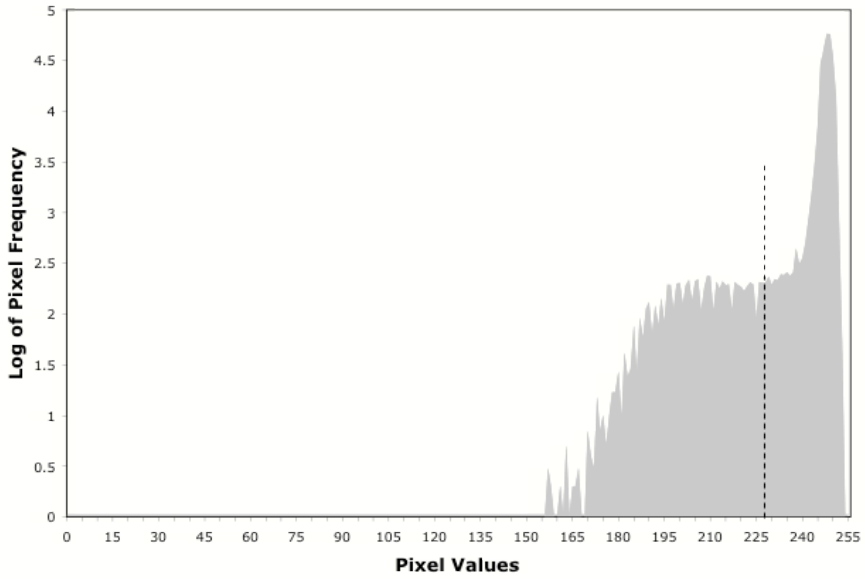


Figure 3.3: Histogram of pixel frequencies for Figure 3.1, where the dashed line is the final threshold

Signature binarisation using this method is carried out by first providing a threshold, such as the average grey level of all pixels, that will split the histogram. The mean of the pixel values of each half of the histogram is then calculated and a new threshold is found as the midpoint between these means. This process is then repeated with the new threshold and is continued until the threshold stabilises. Figure 3.4 shows how a signature will be binarised using this particular method.

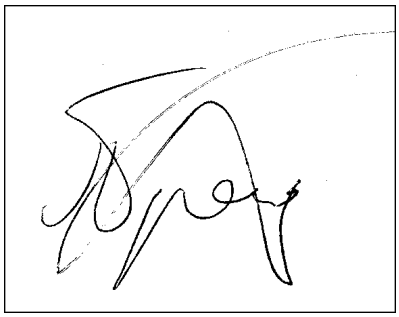


Figure 3.4: Signature that has been binarised using iterative thresholding

The iterative thresholding method is an effective method for binarising a sig-

nature if there is a distinct contrast between the signature and its background. It is also a computationally fast algorithm which after constructing the histogram, only requires approximately four passes over the grey level histogram to determine a threshold. The disadvantage of iterative thresholding is that if the pixel values that make up the signature vary substantially the calculated threshold maybe too low, resulting in sections of the signature being thresholded out.

3.1.3 Otsu

Otsu's method (Otsu et al., 1979) is one of the most common binarisation techniques for thresholding grey-scale images. It is an unsupervised clustering method for automatically choosing a global threshold that will split the signature into the foreground and background classes. The manner in which the optimal threshold is chosen is by exhaustively searching through the normalised frequency histogram to find the point which minimises the intra-class variance, or in other words, minimising the spread of the grey levels in each class. A problem arises though in the fact that computing the intra-class variance for each threshold is computationally expensive. Another way of achieving this is to find the between-class variance instead, as when this is maximised, the intra-class variance is minimised.

Finding the optimal threshold begins with constructing a normalised frequency histogram. This is carried out by taking the number of pixels n at each grey level i and normalising this number by dividing it by the total number of pixels N in the image. The resulting value p_i is the probability that this level occurs. This calculation is shown in Equation 3.1.

$$p_i = \frac{n_i}{N} \tag{3.1}$$

Using this normalised histogram, each level is then tested to determine how effective it is at being a threshold, where the current threshold is designated k. The first step is to calculate the probability that each of the two classes occur, with the probability of the class below k being found via Equation 3.2, and represented by

$\omega(k)$. The class above k can be found by $1 - \omega(k)$.

$$\omega(k) = \sum_{i=0}^{k} p_i \tag{3.2}$$

The next step is to find the mean level of the class below the kth level. This mean level is also the first-order cumulative moment of the histogram up to the kth level. The way in which this is found is shown in Equation 3.3 and is represented by $\mu(k)$.

$$\mu(k) = \sum_{i=0}^{k} i \times p_i \tag{3.3}$$

This process is then repeated to produce the total mean level of the signature image, where Equation 3.4 shows how this is calculated. The variable L is the total number of levels in the histogram.

$$\mu(L) = \sum_{i=0}^{L} i \times p_i \tag{3.4}$$

Using the previous calculated values, the between-class variance for the kth level can then be found by Equation 3.5 as described by Otsu et al. (1979).

$$\sigma_B^2(k) = \frac{[\mu(L) \times \omega(k) - \mu(k)]^2}{\omega(k) \times [1 - \omega(k)]} \tag{3.5}$$

The optimal threshold is then found at the k value which maximises σ_B^2, as this value will also minimise the inter-class variance. Figure 3.5 shows the effect that Otsu binarisation has on a signature.

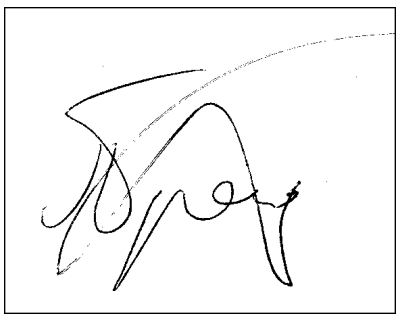


Figure 3.5: Signature that has been binarised using Otsu's method

Otsu's method is often chosen for image binarisation as it has a simple algorithm which only utilises the zeroth and the first-order cumulative moments of the grey-scale histogram. As well as its simplicity, Otsu's method requires no human involvement, allowing for an optimal threshold to be selected automatically and stably. This selection is not based on the differentiation of local properties in the histogram, such as valleys, but from the integration of the entire histogram. The result of this is that the Otsu method can deal with images that do not have a defined distinction between the foreground and background.

3.1.4 Niblack

The Niblack method (Trier and Jain, 1995; Blayvas et al., 2006) is a locally adaptive binarisation technique that calculates an independent threshold for each pixel, based on a sample area around the pixel. This threshold T at pixel (x, y) is calculated by Equation 3.6, where $m(x, y)$ is the mean of pixel values in the sample area and $s(x, y)$ is the respective standard deviation. The variable k is then used to adjust how much of the boundary around the signature is taken as part of the signature; it has been found that $k = -0.2$ gives well-defined results (Trier and Jain, 1995).

$$T(x, y) = m(x, y) + k \times s(x, y) \tag{3.6}$$

The pixels that make up the sample area are found in a square of $r \times r$ in

size, and is centred at (x, y). This sample area for binarisation has been set to 9×9 instead of the recommended 15×15 as it better captures an equal amount of signature and background when the sample area is centred on a signature pixel. Figure 3.6 shows the binarising ability of Niblack.

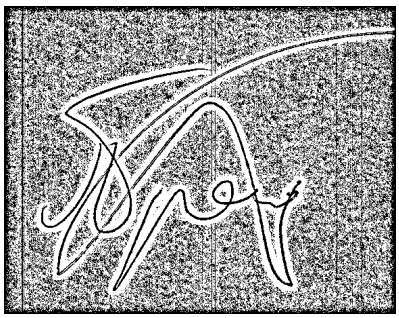


Figure 3.6: Signature that has been binarised using Niblack's method

It can be seen in this binarisation example that the capability of Niblack is effective in identifying the wanted signature, but outside of the signature area a large portion of the background pixels are incorrectly thresholded. The result of this is that the capability of Niblack in correctly thresholding a signature image is diminished, and as such, severely effects the extraction of image features that define the signature. Due to this, the Niblack method will not be employed for binarisation.

3.1.5 Bernsen

Bernsen's method (Bernsen, 1986) is another locally adaptive binarisation technique that thresholds each pixel independently. The first step for a pixel found at (x, y) is to determine the contrast value C of the sample area around the pixel, with this contrast being calculated by Equation 3.7. The variables Z_{high} and Z_{low} are the highest and lowest pixel values of the sample area respectively. If the contrast value is less than a predetermined level ℓ, then the pixel is set as background. (Trier and Jain, 1995) state that $\ell = 15$ has been found to be a good contrast level.

$$C(x, y) = Z_{high}(x, y) - Z_{low}(x, y) \tag{3.7}$$

If C is greater than ℓ, then a threshold T will need to be applied to the pixel, with T being found via Equation 3.8.

$$T(x, y) = \frac{Z_{low} + Z_{high}}{2} \tag{3.8}$$

Following Niblack, the sample area centred around a pixel was chosen as 9×9. The ability of Bernsen's method is shown in Figure 3.7.

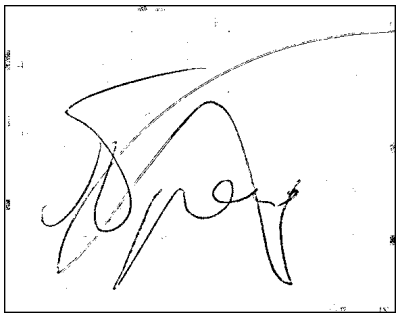


Figure 3.7: Signature that has been binarised using Bernsen's method

Bernsen's method often results in patches of incorrectly thresholded pixels. This is due to the contrast value being greater than 15, causing a threshold to be calculated in response. Because of these unwanted patches, Bernsen's method will adversely affect the extraction of image features, and due to this, will not be used.

3.1.6 Local Iterative Method

Locally adaptive binarisation methods have the advantage of correctly thresholding parts of the signature that would be lost with global binarisation. Their downfall is that they do not deal well with the background, as they tend to incorrectly threshold these pixels where the sample area does not contain part of the signature. From testing these methods, it was noticed that an alternative method of binarisation was needed, and in response, the local iterative method was devised and built. It takes the advantages of previous methods and adapts them to produce an algorithm that thresholds signatures in a more controlled manner. By doing this, the local iterative method limits the amount of noise generated, as well as attempting to reconstruct sections of the signature that are disjointed.

The local iterative method is carried out by following a process of steps for each sample area found in a 9×9 square centred on the pixel at (x, y). This process starts by first finding the mean $m(x, y)$ and standard deviation $s(x, y)$ of the pixel values in the sample area. Using these, any pixel in the sample area that has a value greater than two standard deviations from the mean, is temporarily set as the mean for that sample area. By doing this, any pixels that could potentially be noise are adjusted. The next step is to test if the sample area is solely background through the use of the contrast method described in Section 3.1.5, using the recommended ℓ value of 15. If the contrast level of the sample area is greater than ℓ, then a threshold is used to determine if the pixel at (x, y) is part of the signature or the background. This threshold is calculated using the iterative method (Section 3.1.2), and is only applied to the pixel located at (x, y). The ability of the local iterative method is shown in Figure 3.8.

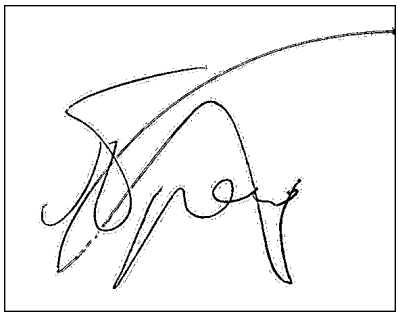


Figure 3.8: Signature that has been binarised using the local iterative method

The local iterative method can in the majority of instances binarise a signature while reconstructing sections that would normally be lost. But due to this procedure, it does have the side-effect of producing a slight border around the signature. This is a result of the sample area capturing a portion of the signature, adversely affecting how the pixel is thresholded. This noise can easily be removed with a median filter (Section 3.2), which also helps to further reconstruct areas of the signature, the achieved effect can be seen in Figure 3.9.

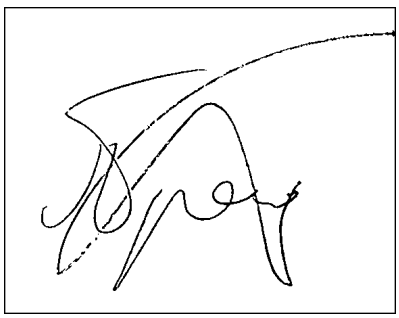


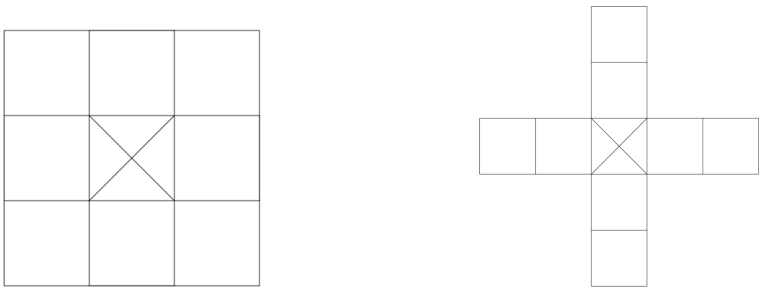
Figure 3.9: Signature that has been improved upon via a median filter

3.2 Median Noise Removal

Often when a signature is binarised, noise from the capturing process will remain. The removal of this noise helps in extracting image features that more accurately represent the signature. Noise removal can be accomplished using a variety of methods, but for this research, only the median approach was tested.

The median approach is a simple technique that is carried out by taking the eight neighbours of a pixel, as well as itself and sorting them by their values. From these nine sorted values, the middle or median value is chosen to be the pixel's new value.

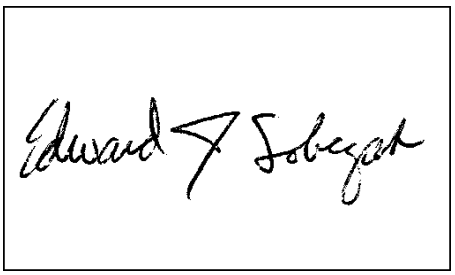
When applying median noise removal, different shaped filters can be used. The 3×3 rectangular filter, shown in Figure 3.10(a), is the most common approach, but has the disadvantage of damaging sharp corners in the signature. Another filter shape that can be used is the + shape, shown in Figure 3.10(b), and is designed to maintain these sharp corners.



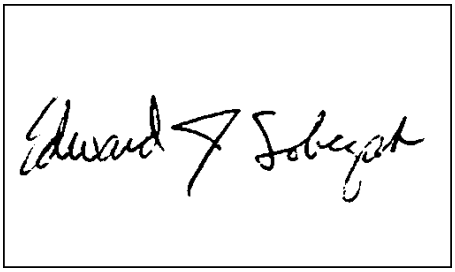
(a) Rectangular filter (b) Plus shaped filter

Figure 3.10: Two filter shapes for median noise removal

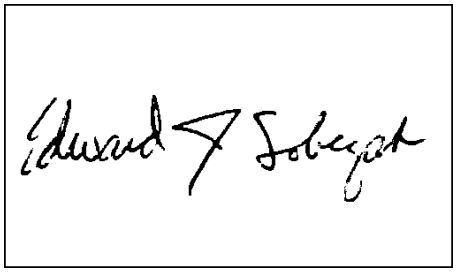
The effect of each filter is shown in Figure 3.11. From these examples, it can be seen that both filtering methods help reduce the amount of noise that occurs, with both approaches being very similar. Because of this similarity, the ability of each filter is unclear, and as such will be determined via experimentation in Section 12.3.1.



(a) Original Image



(b) Rectangular Filtering



(c) Plus Filter

Figure 3.11: Comparison of the two median filters with an original signature.

3.3 Rotation Normalisation

When a signature is captured, its rotation maybe out of sync in relation to similar signatures. This incorrect rotation has the potential of distorting any features that are extracted, adversely affecting the classification. Rotation normalisation was tested using two methods, axis of least inertia and region rotation.

3.3.1 Axis of Least Inertia

A generalised method of rotation normalisation is possible with the axis of least inertia (Kalera et al., 2004). This axis is designed to minimise the spread of black

pixels on either side of itself, and will generally pass through the longest length of the signature. From here, the signature can then be rotated so that its longest length is horizontal.

The axis of least inertia is found by first locating the centre of mass, where $\bar{u}$ and $\bar{v}$ correspond to the x and y coordinates of the centre of mass respectively. The second order moments of the signature are $\overline{u^2}$, $\overline{v^2}$ and $\overline{uv}$ and are calculated by Equations 3.9, 3.10 and 3.11 respectively, where $u(i)$ is the x-coordinate of the ith black pixel in the signature image, while $v(i)$ is the y-coordinate of the same pixel and n is the total number of black pixels in the signature.

$$\overline{uv} = \frac{1}{n}\sum_{i=1}^{n}(u(i) - \bar{u})(v(i) - \bar{v}) \tag{3.9}$$

$$\overline{u^2} = \frac{1}{n}\sum_{i=1}^{n}(u(i) - \bar{u})^2 \tag{3.10}$$

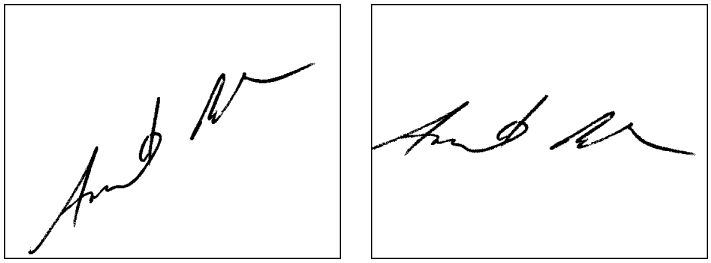
$$\overline{v^2} = \frac{1}{n}\sum_{i=1}^{n}(v(i) - \bar{v})^2 \tag{3.11}$$

The orientation of the axis of least inertia is then found by the orientation of the least eigen vector of the matrix in Equation 3.12, as shown by Kalera et al. (2004).

$$I = \begin{pmatrix} \overline{u^2} & \overline{uv} \\ \overline{uv} & \overline{v^2} \end{pmatrix} \tag{3.12}$$

Having calculated the angle of the axis, the signature can then rotated accordingly, so that the axis of least inertia will correspond with the horizontal axis. The effect that this rotation has on a signature is shown in Figure 3.12.

The disadvantage of using the axis of least inertia is that not all signatures will be rotated correctly, as some signatures will end up being out of alignment. This is due to the signature shape, where if it taller than it is wide, the signature will be rotated on to its side. As a result, manual checking will be required to ensure



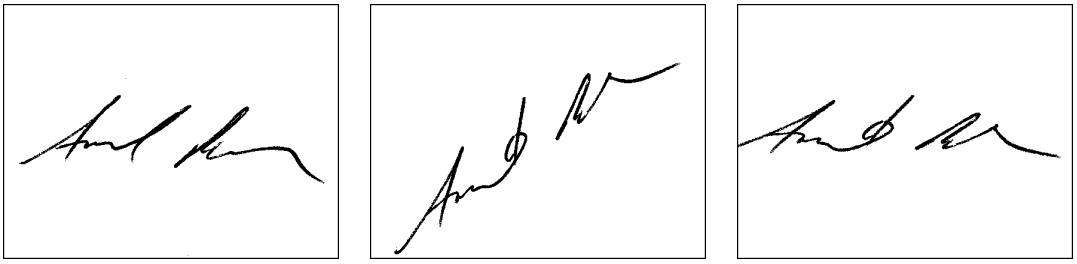
(a) Signature needing rotation (b) A rotated signature

Figure 3.12: Effect of rotating by the axis of least inertia

that the signature is in fact correctly rotated. The use of manual checking defeats the purpose of having an automatic system, and as such, using this method may decrease the classification accuracy.

3.3.2 Region Rotation

An alternative method to the axis of least inertia is to use a base signature from a set to determine how each remaining signature should be rotated. This is carried out by first counting the number of black pixels that occur in each region when the signature is divided by equimass regioning (Section 5.2.1). The similarity between the base signature and the signature that is being rotated can be calculated using the black pixel frequency vector (Section 7.1) with the Euclidean distance. This similarity and the rotation angle are then stored and the process is repeated with a different rotation. The rotation angle that has the greatest similarity with the base signature is then chosen as the final rotation angle. The effect that this approach has can be seen in Figure 3.13.



(a) The base signature (b) Signature needing rotation (c) Rotated signature

Figure 3.13: Effect that region rotation has in relation to the base signature

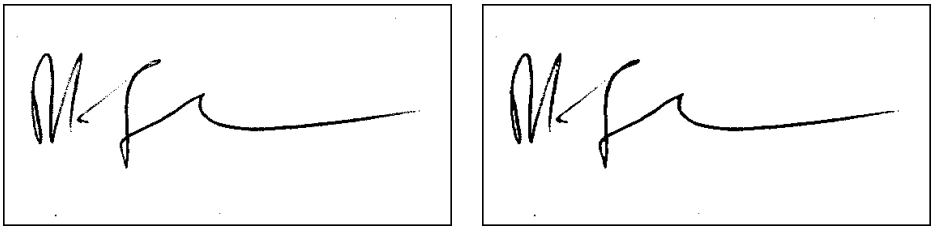
This method is a much slower technique for rotation normalisation as it has a much greater level of complexity than the axis of least inertia. As well as this, it is dependent on a base signature for determining the correct rotation. Region rotation does have the advantage of being less likely to produce an incorrectly rotated signature, but like the axis of least inertia, it is not guaranteed to correctly rotate every signature.

3.4 Region Growing

When a signature is written, the strokes that make up the signature may become damaged due to a variety of reasons, such as a lack of writing pressure or a defective pen. Region growing attempts to repair these broken strokes by filling in the gaps between disjointed pixels, with the aim being to repair each stroke to the form that it was originally intended. This section will cover two methods that can be employed for repairing these broken strokes.

3.4.1 Broken Stroke Connection

Broken stroke connection (Shi and Govindaraju, 1996) is one method of repairing strokes in a signature that have become disjointed. The algorithm behind this method is described with detail in (Shi and Govindaraju, 1996), but basically, it starts at the left most pixel of the signature and searches the area around it, locating short runs of pixels on different rows. If these runs are determined to be broken, they are then completed. The vertical area between these runs are then filled so that any new horizontal lines are the average length of the rows above and below it. Using the top and bottom left most points of this reconstructed area, the process is repeated, until all the pixels have been tested and disjointed areas recreated. Figure 3.14 shows the effect that this method has in repairing a signature.

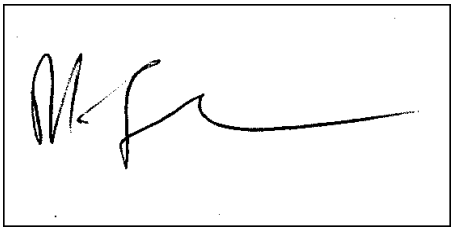


(a) Original signature (b) Signature with region growing

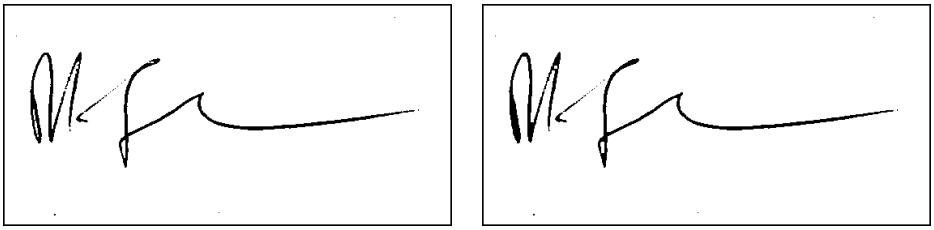
Figure 3.14: Effect that broken stroke connection has on a signature

3.4.2 Dilate and Erode

Dilate and erode reconstructs broken strokes by simply expanding each black pixel, so that the disjointed pixels in the broken strokes reconnect. Eroding then shrinks the black pixel mass back down to its original size. This method is implemented by expanding each black pixel in all directions by a set size and then eroding them by repeating this same process with each of the white pixels. The result of dilating and eroding each black pixel in a signature can be seen in Figure 3.15.



(a) Original signature



(b) Dilate and erode with size 1 (c) Dilate and erode with size 2

Figure 3.15: Effect that dilate and erode has on a signature

This method works when the pixels are dilated by a small amount, as if the size is too large, internal areas of the signature are at risk of being completely filled in, and as a result there is a permanent lose of information about the signature. Figure 3.15(c) shows the result of having a size that is to large, with an internal

area of the M being completely filled in, as well as this, internal areas that have been eroded end up being square patches instead if their original shape.

3.5 Normalisation

After all desired preprocessing has been carried out, the next step is to normalise the signatures, so that no bias is introduced that could skew the achieved results. Signature normalisation is carried out by cropping the signature so that there is no unnecessary background padding on any of the four sides of the signature image. Having cropped the signatures, each image feature is then normalised by dividing its frequency by the number of pixels in the image. This results in the frequency being changed into a percentage, making it scale invariant, allowing the signatures to remain at their initial size without adversely effecting the classification accuracy. Having normalised the signatures, preprocessing is complete and classification can be undertaken.

Chapter 4

Image Features

When classifying a signature, one of the key requirements is the extraction of features that uniquely define it. This information allows the signature image to be broken down into components that are more useful for describing its structure, differentiating it from other signatures when compared. Unlike digital images, these features are not designed to be understood by humans, but rather they allow information about the image to be captured in a statistical manner. This statistical information provides a new representation (Section 7), that allows two signature images to be compared with greater ease. This chapter will describe a variety of different features that can be extracted, each of which defines a different structural property of a signature.

4.1 Mass

One of the easiest features to extract from a binary image is its mass, where mass is simply the frequency of primary pixels within the image. In the case of signatures, the primary pixels are black, and constitute the signature. In general, this feature does not provide enough comprehensive structural information that will allow two signatures to be differentiated from one another. Therefore, the use of region sampling (Chapter 5) is commonly employed to define the image with more accuracy, as it provides spatial information.

4.2 Centre of Mass

The centre of mass of a binary image is located at the symmetrical centre of the signature mass, where this is essentially the average (x, y) position of the black pixels. The centre of mass is denoted by $(\bar{x}, \bar{y})$ and is used in a number of image applications, such as rotation normalisation (Section 3.3) and ring region sampling (Section 5.2.2). The mean and median approaches are two methods for calculating the centre of mass; both of which are described below.

4.2.1 Mean

The prevalent method for calculating the coordinates of the centre of mass is to use the mean coordinates of the pixels that make up the signature mass (Kalera et al., 2004). Using this method, $(\bar{x}, \bar{y})$ is found using equations 4.1 and 4.2, where the variable n is the total number of pixels that comprise of the signature mass. The variables u and v are the vectors of the x and y coordinates respectively of each pixel in the mass.

$$\bar{x} = \frac{1}{n} \sum_{i=1}^{n} u_i \tag{4.1}$$

$$\bar{y} = \frac{1}{n} \sum_{i=1}^{n} v_i \tag{4.2}$$

Figure 4.1 shows an example of where the mean centre of mass will be located for a particular signature image.

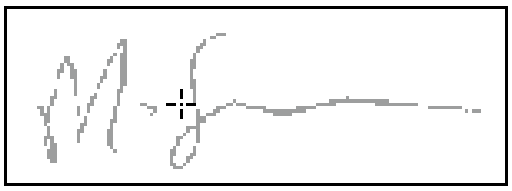


Figure 4.1: Example location of the mean centre of mass in a signature

4.2.2 Median

The median method is an alternative approach for calculating the centre of mass. The coordinate $(\bar{x}, \bar{y})$ is found at the median points of u and v, which are the vectors of the x and y coordinates respectively of each pixel in the mass. Figure 4.2 shows an example of where the median centre of mass is located within a signature image.

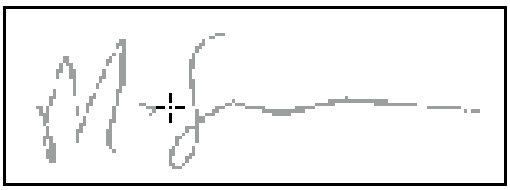


Figure 4.2: Example location of the median centre of mass in a signature

4.3 Gradient

The gradient feature measures the local characteristics of a pixel based on its nearest eight neighbours, where these characteristics describe the likelihood that the pixel is part of an edge as well as the direction that this edge is orientated. The manner in which both of these characteristics are found begins by convolving the pixel P and its neighbours with the two Sobel kernels (Srikantan et al., 1996), which are shown in Equation 4.3, where G_x is the measure of horizontal change and G_y is the measure for vertical change.

$$G_x = \begin{vmatrix} +1 & 0 & -1 \\ +2 & 0 & -2 \\ +1 & 0 & -1 \end{vmatrix} * P \qquad G_y = \begin{vmatrix} +1 & +2 & +1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{vmatrix} * P \tag{4.3}$$

Having found the two gradients, they can then be used to produce the magnitude of the pixel, where the larger the magnitude, the greater the likelihood that the pixel is part of an edge. This calculation is shown in Equation 4.4.

$$G = \sqrt{G_x^2 + G_y^2} \tag{4.4}$$

The direction that the edge is orientated for the pixel at (x, y) can also be derived from G_x and G_y. Equation 4.5 shows how this direction θ is found.

$$\theta(x, y) = \tan^{-1}\left(\frac{G_y}{G_x}\right) \tag{4.5}$$

The direction of each gradient can range from 0.0 to 2π radians, with these directions commonly being split into a number of non-overlapping segments, as shown in Figure 4.3, where s represents the number of segments that the directions are discretized into.

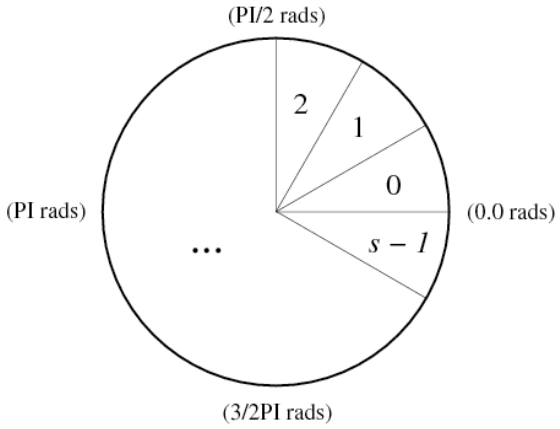


Figure 4.3: The s possible segments that a direction can be divided

The method of determining which segment a direction should be in is found by Equation 4.6, where d is the direction of the captured gradient.

$$segment = round\left(\frac{d \times s}{2\pi}\right) \qquad d \in [0, 2\pi] \tag{4.6}$$

Once every pixel has been assigned to a particular segment, a frequency histogram can be produced that describes either the entire image, or a particular region in terms of its overall gradient.

In image classification, the gradient direction of a pixel is more commonly used than its magnitude, as the direction can be used to produce a gradient map. The magnitude of each pixel can instead be used to produce an edge detected representation of the image. The process for creating this edge detected image is to test

whether the magnitude of a particular pixel is above a certain threshold, where if it is, then the pixel is deemed to be part of an edge, otherwise it is set to background.

4.4 Structural

Using the gradient map (Section 4.3), the embedded structural features of the image can be extracted (Favata and Srikantan, 1996). These structural features are short strokes that occur across several adjacent pixels and describe the intermediate characteristics of the image. Applying a set of 12 rules to each pixel and its neighbours, allows these structural features to be derived from the gradient map, where Figure 4.4 shows the association between a pixel and its eight neighbours.

N3	*N2*	*N1*
N4	Pixel	*N0*
N5	*N6*	*N7*

Figure 4.4: Pixel Neighbours

To achieve the extraction of these structural features, each pixel is first labelled with a number based on the pixel's gradient direction, where Figure 4.5 shows how the value for each label is determined.

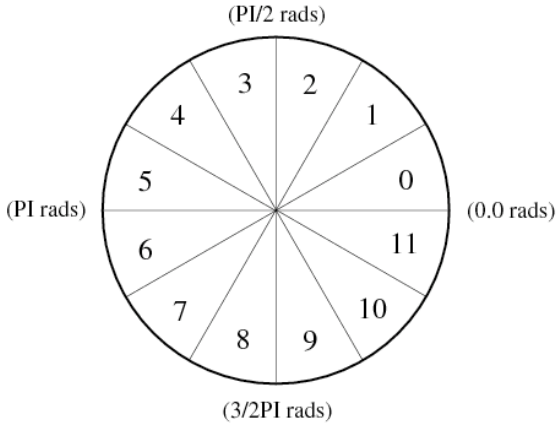


Figure 4.5: Equispaced segments for gradient direction

Using the gradient direction as well as the association between a pixel and its neighbours, the structural features that occur at each pixel can be determined by applying the 12 rules listed in Table 4.1.

Rule	Description	Neighbour 1	Neighbour 2
1	Horizontal line, type 1	$N0$ (2,3,4)	$N4$ (2,3,4)
2	Horizontal line, type 2	$N0$ (8,9,10)	$N4$ (8,9,10)
3	Vertical line, type 1	$N2$ (5,6,7)	$N6$ (5,6,7)
4	Vertical line, type 2	$N2$ (1,0,11)	$N6$ (1,0,11)
5	Diagonal Rising, type 1	$N5$ (4,5,6)	$N1$ (4,5,6)
6	Diagonal Rising, type 2	$N5$ (10,11,0)	$N1$ (10,11,0)
7	Diagonal Falling, type 1	$N3$ (1,2,3)	$N7$ (1,2,3)
8	Diagonal Falling, type 2	$N3$ (7,8,9)	$N7$ (7,8,9)
9	Corner 1	$N2$ (5,6,7)	$N0$ (8,9,10)
10	Corner 2	$N6$ (5,6,7)	$N0$ (2,3,4)
11	Corner 3	$N4$ (8,9,10)	$N2$ (1,0,11)
12	Corner 4	$N6$ (1,0,11)	$N4$ (2,3,4)

Table 4.1: Structural Feature Rules

Each rule examines a particular pattern of neighbouring pixels for allowed gradient ranges. For example, rule one states that if the neighbouring pixel at $N0$ has a

gradient range of 2, 3 or 4 and the pixel at $N4$ also has a gradient range of 2, 3 or 4 then a horizontal line is identified and the rule is satisfied. Figure 4.6 shows an example of this rule, where each value represents the calculated gradient direction for that pixel. For each rule that the pixel conforms to, that rule is marked as true. This allows the frequency of each structural feature to be measured and constructed into a histogram, thus two signature can then be compared via their structural features and their similarity determined.

6	6	6
3	3	3
3	3	3

Figure 4.6: Example of rule one, where the values represent the gradient direction segment as computed using Equation 4.6 for each of the nine pixels

4.5 Large Strokes

The features that are extracted from an image are often low dimensional, that is, they are derived from a single pixel, or from a pixel's immediate neighbourhood. Higher dimensional image features are found based on the layout of the entire image. One of these image features is the large-strokes feature, which attempts to capture large horizontal and vertical strokes from within the signature mass. If the run lengths of these strokes are above a predefined threshold, then the stroke is counted, otherwise it is disregarded. Figure 4.7 shows an example of a signature, in which the strokes of length 12 or more pixels are identified and highlighted; there are three horizontal and four vertical strokes. Generally, the frequency of these strokes will be found on a region by region basis, allowing for a histogram to be produced for each region.

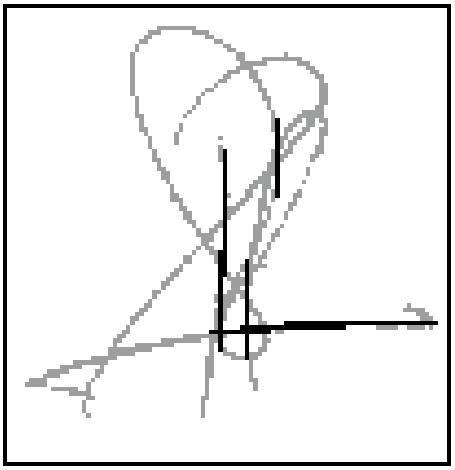


Figure 4.7: A signature containing both horizontal and vertical strokes

4.6 Concavity

The concavity features (Favata and Srikantan, 1996) of an image differ from other features in that each pixel is labelled based on a star-like operator instead of the pixels nearest neighbours. The star-like operator shown in Figure 4.8 is an eight pointed star that extends out from a pixel, allowing it to determine its surroundings; be it the signature or the edge of the image.

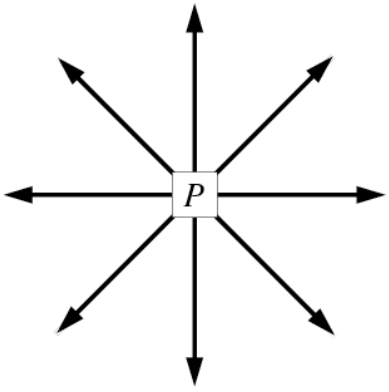


Figure 4.8: Star shaped operator with eight arms

The result of using this operator is that it can determined if a pixel is in a concavity or not based on where each arm ends. Each pixel is then labelled by one of the eleven possible options. The first eight options are the concavities that open

in each of the eight direction, with Figure 4.9 showing a right opening concavity. For a concavity to open in one of these particular directions, the arm of the operator that goes in that direction must hit the image border, while the other seven arms must hit the inside edges of the signature (Favata, 2008).

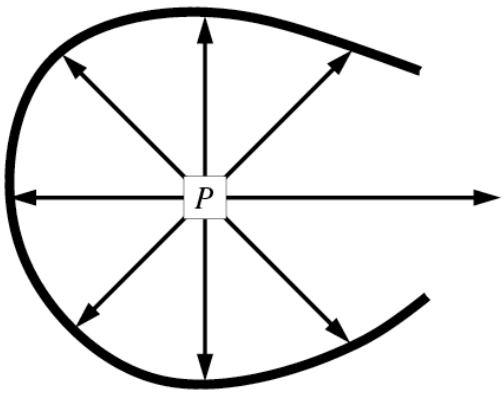


Figure 4.9: Right opening concavity

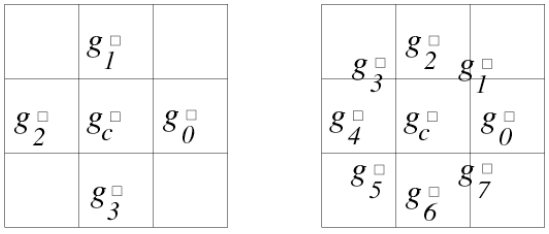
The ninth option is if all of the arms hit the signature, then the pixel is deemed to be in a hole, which is an area entirely enclosed by the signature. A pixel is labelled as being open if two or more arms of the operator hit the border of the image; this is the tenth option, while the eleventh is if the pixel is part of the signature mass (a black pixel), in which case it is labelled as such. The result is that the concavity features also incorporates the mass feature described in section 4.1.

The concavity feature has a variety of possibilities for which it could be expanded upon, based on the number of possible combinations of the operator's arms hitting either the edge of the image or the signature. Previously the concavity feature only measured five aspects, these were the concavities that open in the horizontal and vertical directions as well as any holes. Section 13.2 will determine whether using diagonal opening concavities as well will have an improvement on the classification accuracy or whether they detract from its ability to distinguish one signature from another.

4.7 Local Binary Patterns

Local binary patterns (LBP) are a multiresolution approach to grey scale and rotation invariant texture classification (Mäenpää et al., 2000; Ojala et al., 2002). They essentially allow the texture that surround each pixel to be captured and represented as a ring of binary values. Having constructed the binary pattern that surrounds a pixel, it is then possible to apply different techniques to make these patterns invariant to rotation. This allows for a single texture that has been rotated differently to be determined as still being the same.

The approach that is used to extract the LBP from around a pixel begins by first choosing the P and R values, where P is the number of points around the pixel that will make up the LBP and R is the radius at which these points are found. The manner in which these points are laid out can be seen in Figure 4.10, where g_c is the grey value of the centre pixel, while g_p ($p \in 0, 1, \cdots, P-1$) is the grey value at each of the points. If the coordinates of g_c are at (x, y), then the coordinates of g_p are found via $(x + R \times \cos(\frac{2\pi p}{P}), y - R \times \sin(\frac{2\pi p}{P}))$. The grey values which do not fall exactly in the centre of a pixel are determined by interpolation from the surrounding pixels.



(a) $P = 4, R = 1.0$ (b) $P = 8, R = 1.0$

Figure 4.10: Circularly symmetric LBP neighbourhoods

Having found the values at each point, they then need to be converted in to binary values. This is carried out by subtracting the g_c value from each g_p value, where if the result is greater than or equal to 0, then binary value is 1, otherwise it is 0; this process can be seen in Equation 4.7. The texture that surrounds the central pixel can then be represented as $T = (f(g_0), f(g_1), \cdots, f(g_{P-1}))$, which is

essentially a ring of binary values which has a length of $P-1$.

$$f(g_p) = \begin{cases} 1 & \text{if } g_p - g_c \geq 0 \\ 0 & \text{if } g_p - g_c < 0 \end{cases} \qquad p = 0, 1, \cdots, P-1 \tag{4.7}$$

Producing a single value from the LBP to represent the local texture of a pixel can be achieved using one of several methods, each of which allows the texture to be described in a different manner. These methods are standard, rotation invariant and uniform, each of which are described in Sections 4.7.1, 4.7.2 and 4.7.3, respectively. The final step is to take the produced LBP values and convert them into a frequency histogram. This allows the frequency histograms of two signatures to be compared and from here, their similarity to be determined.

4.7.1 Standard

When producing a single value from a LBP, the simplest method is to take each point and assign a binomial factor 2^p to each produced binary value. This allows each combination of points to be transformed into a unique $LBP_{P,R}$ number that defines the local texture around the central pixel. The manner in which the $LBP_{P,R}$ number is calculated is shown in Equation 4.8.

$$LBP_{P,R} = \sum_{p=0}^{P-1} f(g_p) \times 2^p \tag{4.8}$$

For this approach, if there are eight points around the central pixel, then there will be 256 possible values for describing the texture. Figure 4.11 shows an example of a pixel P that is surrounded by a ring of binary values that have been found using the method described previously, except that the pixel's value is used instead of interpolation. This string starting at middle right and going anti-clockwise is 11001010, which results in this texture being labelled with the value 83.

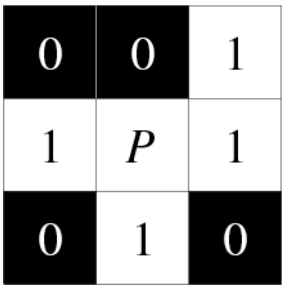


Figure 4.11: Ring of Binary values surrounding a pixel

The disadvantage of using this method is that it is not rotation invariant, meaning that if the texture is rotated at all, then the LBP value will change accordingly. There is a solution to this problem, which can be approached using two different methods, both of which are described in the following sections.

4.7.2 Rotation Invariant

The LBP value that is produced from the set of grey value points always begins with the first point being to the right of g_c. The result of this is that the standard method is not rotation invariant, meaning that if the texture is rotated at all, then the grey values will correspondingly move around the perimeter surrounding g_c, causing a different value to be produced. The effect of rotation can be removed by assigning the same unique identifier to each rotated variation of a LBP. This is carried out by generating the LBP value by starting at each possible point and only keeping the minimum value as the unique identifier. Equation 4.9 shows how this will be carried out, where $ROR(x, i)$ performs a circular bit-wise right shift i times on the P-bit number x. The superscript ^{ri} identifies the formulas as being $r$otation $i$nvariant.

$$LBP^{ri}_{P,R} = min\{ROR(LBP_{P,R}, i) \quad | \quad i = \{0, 1, \cdots, P-1\} \tag{4.9}$$

Using this method to produce rotation invariant LBPs will result in 36 unique values when $P = 8$. The majority of the original LBP values will not naturally be of the desired value and as such will require that the LBP be rotated and recalculated.

Because of this, invariant LBP values are computationally more expensive, but as a result are more robust.

4.7.3 Uniform

It has been observed that the invariant approach does not provide very good discrimination between LBP values (Pietikäinen et al., 2000) as there are certain local binary patterns that are fundamental properties of texture, and as such, provide the majority of the possible LBPs that occur. Because of this, an alternative method was devised by Ojala et al. (2002) that produces rotation invariant values with which to identify the patterns. This alternative method introduces the uniformity measure $U(LBP^{ri}_{P,R})$ as shown in Equation 4.10, which measures the number of spatial transitions (bitwise changes from 0 to 1) that occur in each rotation invariant LBP.

$$U(LBP^{ri}_{P,R}) = |f(g_{P-1}) - f(g_0)| + \sum_{p=1}^{P-1} |f(g_p) - f(g_{p-1})| \tag{4.10}$$

Once the number of bitwise changes have been counted, then a unique value is assigned to each LBP based on how many times that 1 occurs when $U \leq 2$; otherwise if $U > 2$, then the value is set to $P + 1$. This procedure is shown in Equation 4.11, where the superscript ^{riu2} refers to the use of *r*otation *i*nvariant '*u*niform' patterns that have a U value of at most 2.

$$LBP^{riu2}_{P,R} = \begin{cases} \sum_{p=0}^{P-1} f(g_p) & \text{if } U(LBP^{ri}_{P,R}) \leq 2 \\ P + 1 & \text{if } U(LBP^{ri}_{P,R}) > 2 \end{cases} \tag{4.11}$$

For example, if $P = 8$, then there will be a total of ten possible values that can be assigned to an LBP, where the first nine are the rotation invariant LBPs that have $U \leq 2$, with these LBPs being shown in Table 4.2. The tenth value is for all other possible patterns, which in this case is 27, as $(P + 1) + 27 = 36$.

1	00000000
2	00000001
3	00000011
4	00000111
5	00001111
6	00011111
7	00111111
8	01111111
9	11111111

Table 4.2: The 9 circular rotational patterns that can occur when $U = 2$

4.8 GSC Ensemble

The Gradient, Structural and Concavity (GSC) ensemble is a combination of five image features that are commonly used for a range of hand-written text classification problems. This ensemble allows for a quasi-multiresolution approach for measuring image characteristics at difference scales (Chen and Srihari, 2006; Favata and Srikantan, 1996; Srikantan et al., 1996). The image features that GSC is made up of are described previously in this chapter, with gradient features being covered in Section 4.3 and structural features in Section 4.4. The concavity component of GSC is the combination of three separate image features, with these being mass, large strokes and concavity. Each of these were described in Sections 4.1, 4.5 and 4.6, respectively.

Chapter 5

Region Sampling

When features are extracted from an image, their associated spatial information is often lost, as only the frequency of the feature is kept, while its corresponding location within the image is disregarded. It is this frequency information that is used to construct a histogram that represents the signature. Because no spatial information is captured, a potential problem that occurs is that two signatures which have very different appearances may produce similar histograms. The result of this is that these two signatures will produce a similarity score (Section 9.3) that is not representative of their true similarity.

A method of solving this problem is to use region sampling, which is a technique that provides spatial information for the extracted features by breaking an image up into regions. The concept behind region sampling is that the same region across multiple signatures will capture the same sections of signature. The goal of this is that when corresponding regions are compared, the same sections will also be compared, producing a more accurate similarity.

The sections in this chapter will cover the types of region sampling methods and how they are implemented. These methods are uniform, adaptive and irregular adaptive.

5.1 Uniform

Uniform region sampling is the simplest method for splitting a signature up into regions and is carried out by ensuring that each region has the same number of pixels between each of its edges in relation to all of the other regions in the image.

5.1.1 Grid

A uniform grid (Favata and Srikantan, 1996) creates rectangular regions for sampling, where each region is of the same size and shape. This is carried out by placing the grid lines at equally spaced positions along the x-axis of the image; creating the vertical regions. Similarly, the horizontal regions are produced by placing grid lines at equally spaced positions down the y-axis. The positions of both the horizontal and vertical grid lines are found by Equation 5.1, where the value p is the vector of line positions, n is the number of horizontal or vertical regions, and l is the width or the height of the image. The purpose of using the ceiling in Equation 5.1 is to correct the location of the grid lines so that they are more evenly distributed across the signature.

$$p_i = \left\lceil i \times \frac{l}{n} \right\rceil \qquad i = 1, 2, \ldots, n-1 \tag{5.1}$$

An example of how a uniform grid will be placed on a signature is shown in Figure 5.1. This signature has a resolution of 476×147 and has been overlaid with an 8×4 grid. The first vertical grid line is found at $\lceil i \times \frac{476}{8} \rceil = 60$, when $i = 1$. The rest of the vertical lines are found by varying i, producing lines at 119, 179, ... , 417. The horizontal grid lines are found in the same manner, with the first being located at $\lceil i \times \frac{147}{4} \rceil = 37$, when $i = 1$. The rest of the horizontal lines are once again found by varying i, producing lines at 74 and 111.

The uniform grid requires each signature to be the same size, shape and rotation for each region to capture the same section across corresponding signatures. The problem that occurs when proportions such as the size and rotation differ between signatures is that each region will capture sections that do not correlate with

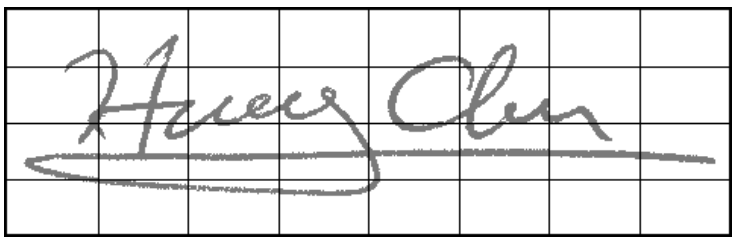


Figure 5.1: Example of regioning using a uniform grid

corresponding regions in other signatures. This is especially true with the uniform grid, as it does not adapt to where the signature is located.

5.1.2 Ring

The uniform ring regioning method is introduced in this thesis, and uses concentric circles centred around the signatures centre of mass (Section 4.2) to create each region. These regions are defined as being the area between subsequent rings, where each ring is found at an equal distance from the previous. The radius of each ring is found by Equation 5.2, where p is the vector of radius lengths, d is the maximum distance that any one pixel lies from the centre of mass and n is the total number of regions.

$$p_i = \left\lfloor \frac{d}{n} \right\rfloor \times i \qquad i = 1, 2, \ldots, n-1 \tag{5.2}$$

The floor in this equation ensures that no regions are lost due to any previous regions being larger than they should. The carry on effect is that the last region usually has a larger distance between d and the last ring than the distance between the previous rings.

Figure 5.2 shows an example of how a signature is divided up into regions based on the uniform ring regioning method. This signature has a resolution of 476×147 and has been divided up into six regions. The mean centre of mass is found at the coordinates 210×77, with the furthest pixel being located in the top right corner of the signature, resulting in d equalling 276. From this, the first ring is found at a distance of $\lfloor \frac{276}{6} \rfloor \times i = 46$ from the centre of mass when $i = 1$. Each of the

subsequent rings are then found by varying i, producing rings with radiuses of 92, 138, 184 and 230 from the centre of mass.

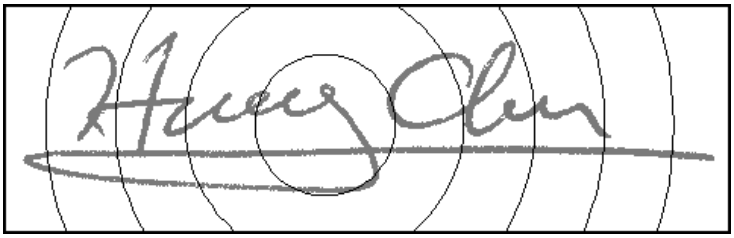


Figure 5.2: Example of regioning using a uniform ring

The purpose of using circular regions instead of a rectangular grid is to make the region sampling rotation invariant. This allows the same structural properties be captured by corresponding regions when each signature has a different rotation.

5.2 Adaptive

Adaptive regioning is the next progression from uniform regioning in which the region boundaries are adjusted to better express the signature sections. By doing this, the similarity between corresponding regions from multiple signatures can be calculated with more accuracy, as they are more likely to contain the same sections of signature.

5.2.1 Equimass Grid

An equimass grid is similar to the uniform grid described in Section 5.1.1, except the horizontal and vertical grid lines are adapted to the number of black pixels, also known as the mass, of the signature (Favata and Srikantan, 1996). Based on this, the grid lines that define each region are found at the equimass divisions of the horizontal and vertical mass histogram of the image. That is, where the total mass between all adjacent points on either the x-axis or the y-axis are as close to equal as possible. Equation 5.3 finds this average mass M_A between two adjacent points, where M is the total mass of the signature and n is the number of horizontal or vertical regions.

$$M_A = \frac{M}{n} \tag{5.3}$$

The horizontal lines are found by counting the number of black pixels horizontally until the count equals M_A, at which point a grid line is placed, and is repeated until the bottom of the signature is reached. This same process is used for placing the vertical lines, except the black pixels are counted vertically until the opposite side of the signature is reached.

An example of an equimass grid is shown in Figure 5.3, in which the signature has a resolution of 476×147 and has been broken up with an 8×4 grid. The total mass of this signature is 6420, making the average mass of the vertical regions $\frac{6420}{8} \approx 803$, the horizontal regions are then $\frac{6420}{4} \approx 1605$. From this, the vertical grid lines are found at 69, 105, ... , 356, while the horizontal grid lines are found at 62, 78 and 95.

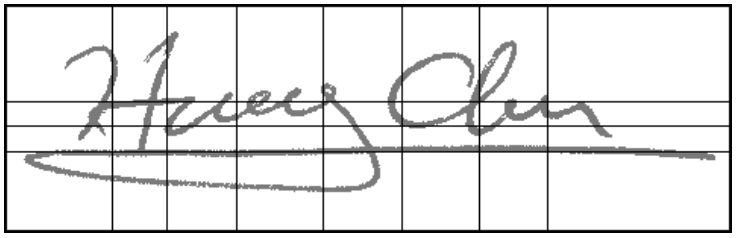


Figure 5.3: Example of regioning using an equimass grid

5.2.2 Equimass Ring

The equimass ring is an adaptive version of the uniform ring described in Section 5.1.2. The size of each region is dependent upon the total number of black pixels in the signature, with this total being split as evenly as possible between each region. The goal of splitting the signature in this manner is to better capture the same sections between similar signatures. Figure 5.4 shows the layout of the equimass ring when applied to a signature.

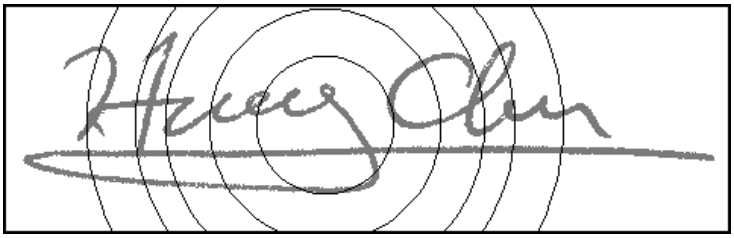


Figure 5.4: Example of regioning using an equimass ring

5.2.3 Centre of Mass Grid

Having an equal amount of signature mass in each region is an effective method of capturing signature sections which correspond between similar signatures. The equimass method described in Section 5.2.1 attempts to achieve this, and for the most part succeeds, but has the disadvantage of some regions differing in the amount of mass. To overcome this, the centre of mass grid was introduced, building the regioning grid in a new fashion. This grid is built by first finding the centre of mass (Section 4.2) of the signature image. Then using this point, a horizontal and vertical line is placed dividing the signature into quadrants. Due to the nature of the centre of mass, each of these quadrants will contain an equal share of the total signature mass. A new centre of mass is then found for each of these quadrants, with the process beginning again, until the wanted number of horizontal or vertical regions is reached. If the number of horizontal and vertical regions differ, then the grid is built in such a fashion that for certain quadrants, only the horizontal or vertical lines are added.

Figure 5.5 shows an example of how a signature will be regioned using the centre of mass grid, given that the wanted grid is 8×4 in size. To further illustrate the construction of Figure 5.5, the first centre of mass is indicated with $\bigotimes$, while the secondary centre of masses are identified with $\bigcirc$.

5.3 Irregular Adaptive

The next step up from adaptive grids is to distort the grid lines so that they have a shape that more accurately captures the same sections across multiple signatures.

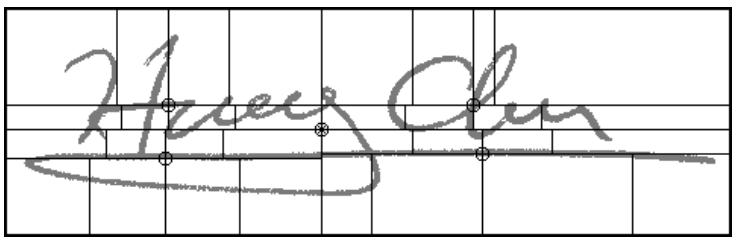


Figure 5.5: Example of regioning using the centre of mass grid

These distorted grids are termed irregular adaptive as each grid line has an irregular shape. Due to their nature, irregular adaptive grids are complex and difficult to construct. In literature (Chen and Srihari, 2006), there has only been one published irregular adaptive grid. This method was not implemented as there was not enough detail provided about the algorithm to construct it. It is unsure whether the irregular adaptive approach provides any benefit given the added complexity, however Section 15.3.1 will compare the results achieved by Chen and Srihari (2006) from its use with the methods implemented in this thesis.

Chapter 6

Spatial Pyramids

The splitting of a signature into individual sections via regioning sampling (Chapter 5) permits extracted features to have corresponding spatial information. The use of a spatial pyramid (Lazebnik et al., 2006) allows this information be improved upon for each feature by defining where they lie within the signature at different levels of granularity. By using spatial pyramids, the features from two signatures can be compared at different levels of granularity, allowing their similarity to be determined with greater accuracy.

6.1 Granularity

The construction of a spatial pyramid is based primarily off of the regioning method that has been chosen, and ultimately defines how each level will divide the signature. The initial number of horizontal and vertical divisions of the signature produce what will be the finest granularity of the signature, with each subsequent level being derived from these initial divisions and becoming more coarse, until the desired or global level is reached.

Each level is produced from the previous level by first dividing the signature with the chosen regioning method. The next consecutive level of the spatial pyramid is then found by halving the number of horizontal and vertical regions, effectively reducing the total number of regions by three quarters. This is continued until the chosen number of levels is reached, or no new levels can be created. The

instance in which no new levels can be created occurs when the number of horizontal or vertical levels can no longer be evenly divided by two. Figure 6.1 shows how the division lines will be placed on a signature which has had a three level spatial pyramid applied to it, if the number of regions for the finest level (a) is 8×4. The result of using differing amounts of horizontal and vertical cells is that the coarsest level will not be a single region, but can be achieved by using an equal amount of horizontal and vertical cells.

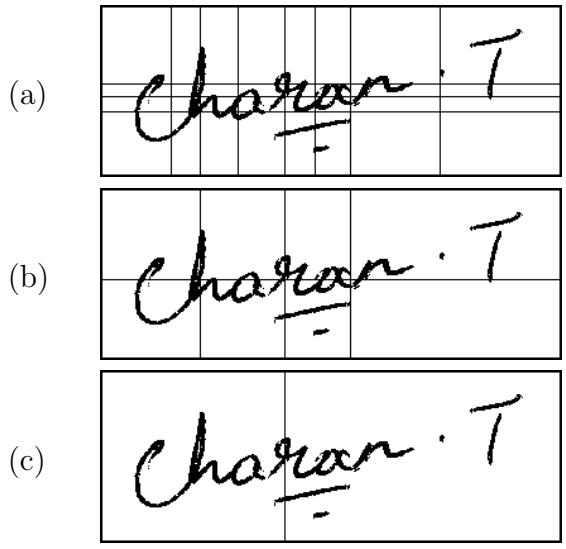


Figure 6.1: A three level spatial pyramid

This process of making each consecutive level based on the horizontal and vertical cells of the previous level does not apply to all varieties of regioning methods. For instance, ring regioning has a set number of circular regions based off of a single value. In this case, the manner in which the spatial pyramid is constructed remains essentially the same, expect only this single value is halved, determining the number of regions in the subsequent level. The construction of new spatial levels is stopped when the number of predefined levels is reached, or the number of regions is no longer divisible by two.

6.2 Weighting

Currently, when using a spatial pyramid to aid in determining the similarity between signatures, the impact that each level has on the final similarity is dependent upon the number of features that have been extracted from that level divided by the total number of features extracted from all levels. For example, if there are three levels in the spatial pyramid and the global level has 36 features, the middle level has 144 and the finest level has 576 features, the total number of extracted features will be 756. The weighting that the global level will have on the similarity measure will then be $\frac{36}{756} = 0.0476$ or 4.76%, and using this same calculation, the medium and finest levels have a 19.05% and a 76.19% weighting, respectively.

Using the number of extracted image features per level may not be the most ideal way of determining the impact that each level will have on the similarity. Because of this, changing the weighting of each level may improve the accuracy of the similarity measure. This weighting makes the calculated similarity for a level proportionate to the total number of features. Using the previous example, the number of features at the global level is 36, if this is adjusted so that it has a 25% total impact on the similarity measure, then the weighting for this level is a value that multiplies 36 so that it equals 25% of 756. This value would 5.25. Lazebnik et al. (2006) use a 25% weighting for the global level for their spatial pyramid, with the medium and fine levels having a 25% and 50% weighting respectively.

The manner in which the similarity score between two signatures is calculated requires that the chosen similarity measure (Section 9.3) be rearranged to manage multiple spatial levels, each of which has their own weighting. This modification can be seen in Equation 6.1, where w_i is the weighting value for level i and l is the total number of levels. In this equation, the function $f(V_{i1}, V_{i2})$ is a similarity measure that has had its denominator removed, where V_{i1} and V_{i2} are the binary feature vectors of length n_i for the two signatures at level i. Dividing by the vector length ensures that the final score is between 0 and 1.

$$score = \frac{\sum_{i=0}^{l} w_i \times f(V_{i1}, V_{i2})}{\sum_{i=0}^{l} n_i} \tag{6.1}$$

The effect that changing the weight for each level has on the similarity measure between two signatures will vary widely based on which level has the greatest impact on the similarity. The influence that the weighting of each level has on the final classification accuracy can not be determined without experimentation, and as such, Section 13.4 will cover how different weightings affect the results.

Chapter 7

Image Representation

When a signature is in a digital image format it is represented in a manner that is visually identifiable for humans, making it easier to comprehend what is being seen. This format is not an ideal choice for representing a signature for the purpose of classification, as the structural properties that characterise each signature can not easily be compared. The comparison of signatures can be improved upon by transforming them into a format that helps to clearly identify their structural properties. This chapter will look at two common methods for representing a signature.

7.1 Frequency Vector

In image classification, one of the most common methods used for representing an image is via a frequency vector, where each element in the vector measures the frequency of a particular structural property of the signature. A visual example can be seen in Figure 7.1, where each value is the frequency of a particular property.

136, 47, 89, 103, 24, 96, 118, 149, 6, 71, 153, 82, 192, 75, 84

Figure 7.1: A frequency vector

Equation 7.1 shows the mathematical structure of a frequency vector, where V is the vector, z is the frequency of a particular property and n is the number of elements.

$$V = (z_1,\ z_2,\ \cdots,\ z_n) \qquad z_i \in \mathbb{N} \qquad \forall i \in \{1, 2, \cdots, n\} \tag{7.1}$$

The advantage of transforming a signature into this form is that it allows two images to be compared based on the chosen structural properties. From this comparison, the similarity between two vectors and thus the two signatures can be easily computed.

Using the frequency of each image is not the most ideal method of representing the image, as the frequency is proportional to the size of the signature image. This means that if two signatures are of different sizes, the frequency of their structural properties will be skewed disproportionately in relation to each other. The manner in which this skew is overcome is to normalise the frequency in relation to the total count of image features that is possible. In most cases, this total is based on the number of pixels within the image. If the frequency of a structural property is divided by the total pixel count, then it will be converted into a percentage which is size invariant.

For example, if a particular property had a frequency of 162 and 374 for two signature images, and each of these signatures had a total pixel count of 625 and 1444 respectively, then the property would be $\frac{162}{625} = 0.2592$ or 25.92% for the first signature and $\frac{374}{1444} = 0.2590$ or 25.90% for the second. Even though the frequency of the structural property was different for the two signatures, proportionally they are very similar. If this process was applied to the frequency vector in Figure 7.1, each element in the normalised vector (as shown in Figure 7.2), would be proportional to the sum of all the frequencies, which in this case is 1,425.

.095, .033, .063, .072, .017, .067, .083, .105, .004, .05, .107, .058, .135, .053, .059

Figure 7.2: A normalised frequency vector

7.2 Binary Feature Vector

The previously described frequency vector gives an accurate measure of the individual properties that have been chosen to describe a signature. Using a binary feature vector (BFV) is an alternative method to this, where the frequency of particular property is replaced by either a 0 or a 1 based on a threshold. If the frequency of the property exceeds or is below the threshold, the bit is set to 1, otherwise it is set to 0. Figure 7.3 shows a visual example of what a binary feature vector might look like after thresholding.

1, 0, 1, 1, 0, 1, 0, 0, 0, 0, 1, 0, 1, 1, 1, 0, 1, 0, 0, 0, 1, 0, 1, 1, 0, 0, 0, 1, 0, 1

Figure 7.3: A binary feature vector

The mathematical structure that defines a BFV is shown in equation 7.2, where V is the BFV, z is a feature bit and n is the number of elements in the vector.

$$V = (z_1,\ z_2,\ \cdots,\ z_n) \qquad z_i \in \{0, 1\} \qquad \forall i \in \{1, 2, \cdots, n\} \tag{7.2}$$

Using a binary feature vector allows each feature bit to be set with a closer relation to the corresponding feature frequencies in the training signatures. Section 8.2 covers the construction of the thresholds that carry this out. Measuring the similarity between two signatures is also easier, as the feature bits either match or they do not; Section 9.3 discusses in more detail the methods used for measuring the similarity between binary feature vectors.

Chapter 8

Feature Thresholding

The creation of the binary vector from a feature frequency vector involves determining whether each feature element matches a certain criteria. If it does, the corresponding feature bit is set to on, otherwise it is left off. Feature thresholding is often carried out using a manually fixed threshold. However, this method often proves to be ineffective, as it requires human involvement to be set, and is not based on the training signatures. Because of this, new approaches were explored as part of the research conducted in this thesis. This chapter describes the evolution of these approaches and have been broken into the manual and automatic methods.

8.1 Manual

When thresholding a feature, a common approach is to use a manually set threshold (Favata, 2008) that determines how the corresponding feature in the binary feature vector is set. If the number of occurrences of this feature is greater than or equal to the threshold, then the criteria is met and the feature bit is turned on, otherwise it is left off. This method is rather rigid, as it is applied to the entire image. This makes it ineffective when used with region sampling (Chapter 5), as the same feature will differ percentage wise between regions. Because of this, the threshold needs to adapt to each region. So to begin for a particular region, the average frequency (μ) of the feature is found from each training signature. The formula for carrying this out is given in Equation 8.1, where T represents the feature frequency

for each training signature and n is the total number of training signatures.

$$\mu = \frac{1}{n} \sum_{i=1}^{n} T_i \tag{8.1}$$

The threshold can then be found by multiplying μ with a fixed value m, which either increases or decreases the threshold in relation to μ by having m either above or below 1. This adjustment to the threshold is carried out by Equation 8.2, and allows the threshold to be moved so that it more accurately captures genuine feature frequencies.

$$threshold = \mu \times m \tag{8.2}$$

The disadvantage of using a single threshold is that if a forgery has a feature count much greater than the threshold, instead of being relatively close to μ, it will still be accepted as a genuine. The solution to this is to have a lower and upper bound. These bounds are created using a method similar to the one previously described, but instead, a proportional value of μ is both added and subtracted from itself, with Equations 8.3 and 8.4 performing this respectively.

$$lower = (1 - m) \times \mu \tag{8.3}$$

$$upper = (1 + m) \times \mu \tag{8.4}$$

The feature count from an unknown signature will only then be accepted if it falls between the lower and upper bounds, making the accepted feature count more controlled.

The spread of feature counts about μ may vary in a fashion that is not normally distributed, as there is a variety of reasons (Section 1.1) that may cause a genuine signature to be inconsistent with other genuine signatures. So to improve the capture of features that are genuine, the distance that both the lower and upper bounds lie from μ is adjusted so that they are independent from each other. This is carried out using two multipliers, where m_L is the lower multiplier and m_U is the upper multiplier, as seen in Equations 8.5 and 8.6, which replace Equations 8.3

and 8.4 respectively.

$$lower = (1 - m_L) \times \mu \tag{8.5}$$

$$upper = (1 + m_U) \times \mu \tag{8.6}$$

Manually chosen thresholds often require a range of experiments to find the optimal value. These experiments use the test data to evaluate different thresholds. This usually results in the chosen threshold having been optimised for the test data instead of being found purely from the training data. The problem with this is that when used in practise, forgeries are generally non-existent, meaning that no adequate testing can be carried out to determine the optimal threshold. The solution to this significant problem is to derive a threshold from the training data only, which Section 8.2 covers in more depth.

8.2 Automatic

The advantage of using an automatic method for feature thresholding is that the threshold can be derived from *only the genuine training signatures and does not require a range of experiments for it to be optimised* (except for determining its performance once created). The simplest method of creating an automatic threshold is to just use the average feature count from the training signatures. Equation 8.7 shows the formula for this operation, where n is the number of training images and T is the count of a particular feature from the i-th training signature.

$$\mu = \frac{1}{n} \sum_{i=1}^{n} T_i \tag{8.7}$$

Because the average lies between the highest and lowest feature counts, its use as a threshold does not really work, as approximately half of the genuine feature counts will fall below this value and will be incorrectly set. To improve this, the threshold is instead set at one sample standard deviation less than the average,

where the sample standard deviation is found by Equation 8.8.

$$S = \sqrt{\frac{1}{n-1}\sum_{i=1}^{n}(T_i - \mu)^2} \tag{8.8}$$

The choice of using the sample standard deviation, is that one standard deviation away from the mean will capture approximately 68% of the genuine feature counts, making it more difficult for a feature count from a forgery to be accepted. The use of the sample standard deviation, instead of the standard deviation, is due to it being a population estimator, therefore, it is more representative of the entire population, not just the feature counts from the training signatures. The same idea of using both a lower and an upper bound from Section 8.1 is also incorporated, resulting in the upper and lower bounds being set by Equations 8.9 and 8.10 respectively, which replaces Equations 8.5 and 8.6 in the automatic case.

$$lower = \mu - S \tag{8.9}$$

$$upper = \mu + S \tag{8.10}$$

The manner in which this method converts the frequency of a feature into its corresponding feature bit can be seen in Figure 8.1. This figure shows how the frequencies of the feature taken from the training signatures will generally lie in a fashion that is normally distributed. If the frequency of the feature is between the lower and upper thresholds then the defined criteria is met and the feature bit is turned on. Otherwise, if the frequency is outside of these two thresholds, then it is deemed to be too far removed from the mean derived from the training signatures and as a result, the feature bit is set to off.

Calculating the lower and upper bounds separately will also help to further account for the way in which the feature frequencies from the training signatures are spread above and below the mean. This is done in a similar manner as above, except the lower bound is found by Equation 8.11, where S_L is the sample standard deviation of the values below μ and is calculated by Equation 8.13. The upper

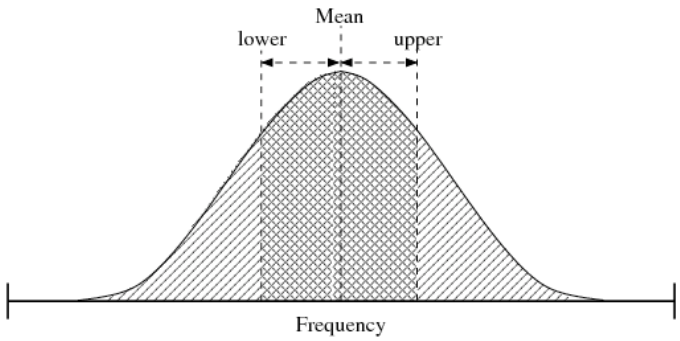


Figure 8.1: Normal curve showing lower and upper threshold placement

bound is found in a similar fashion by Equation 8.12, with S_U being calculated by Equation 8.14, which is the sample standard deviation of the values above μ.

$$lower = \mu - S_L \tag{8.11}$$

$$upper = \mu + S_U \tag{8.12}$$

$$S_L = \sqrt{\frac{1}{T_L - 1}\sum_{i=1}^{n}(T_i - \mu)^2} \qquad \forall T < \mu \tag{8.13}$$

$$S_U = \sqrt{\frac{1}{T_U - 1}\sum_{i=1}^{n}(T_i - \mu)^2} \qquad \forall T > \mu \tag{8.14}$$

The variable T_L is the number of training images that have a value less than μ, while T_U is the number that have a value greater than μ. By having the upper and lower bounds based off of separately calculated standard deviations, each feature frequency can be independently and adaptively thresholded, while taking into account possible outliers. This allows the corresponding feature bit in the binary feature vector to be set with a greater relation to the training data. Figure 8.2 shows how the feature frequency may be skewed due to an outlying signature. Outliers occur when the sample of training signatures is small, or if the subject writes a signature that differs substantially to normal (Section 1.1). To counteract this, the lower and upper thresholds are placed independently from each other. Because of this adaptability, this newly introduced method for feature thresholding

has been termed *adaptive feature thresholding* (Larkins and Mayo, 2008).

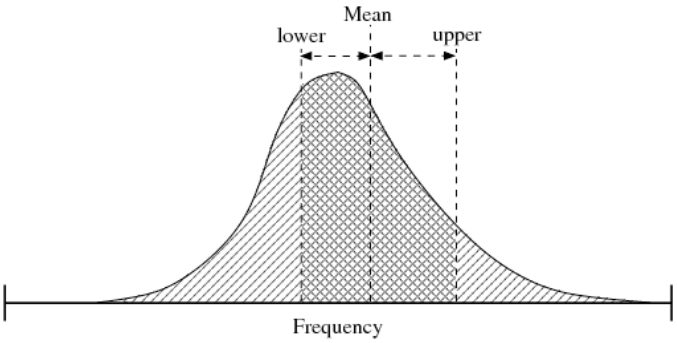


Figure 8.2: How the thresholds will be placed if the frequencies of a feature are skewed due to the training data

Chapter 9

One Class Classification

One class classification is utilised for classifying an unknown signature when only genuine training signatures are available to train with. In practise this is the realistic scenario for achieving signature verification. To achieve this, each signature in a set is converted to a binary feature vector. The next step is to use the chosen training signatures to build a classifier that will determine whether an unknown signature is genuine or forgery. In essence, this is accomplished by determining the similarity of this unknown signature in relation to the training signatures, where if the similarity is above a certain threshold, then the signature is classified as being genuine, otherwise it is deemed to be a forgery. This chapter will look at this approach in greater detail.

9.1 Distance Statistics

The method used for classifying an unknown signature begins by first comparing the binary feature vector (BFV) of the unknown signature with each of the training signatures, using one of the similarity measurements to be described in Section 9.3. This produces a set of similarity scores. The overall similarity between the unknown signature and the training signatures is then found as the mean score. Equation 9.1 shows this calculation, where the function S is the similarity measure and n is the number of positive training signatures for a particular subject; there will generally

be about 8 to 16 of these.

$$score = \frac{1}{n}\sum_{i=1}^{n} S(U, T_i) \tag{9.1}$$

Because each signature is classified based on its similarity score against the training signatures, this score will be a value between 0 and 1. The higher the similarity score, the greater the likelihood that the unknown signature is genuine. Only a single threshold is therefore needed for determining at what point on the similarity scale a score indicates that a signature is a genuine and not a forgery.

This threshold is found by first calculating the mean similarity when each training signature is compared with every other training signature. Equation 9.2 shows how this calculation is carried out.

$$\mu = \frac{2}{n^2 - n}\sum_{i=1}^{n-1}\sum_{j=i+1}^{n} S(T_i, T_j) \tag{9.2}$$

Using μ as the threshold will tend to classify half of the genuine signatures as forgeries, as it is the mean distance between the positive training examples. Instead, μ is used as a basis for calculating a new threshold that will classify an unknown signature. This new threshold is found as an offset of μ, which moves the threshold from μ down the similarity scale, as shown in Figure 9.1. Two methods were investigated for finding this offset, manual and automatic, both of which are described below.

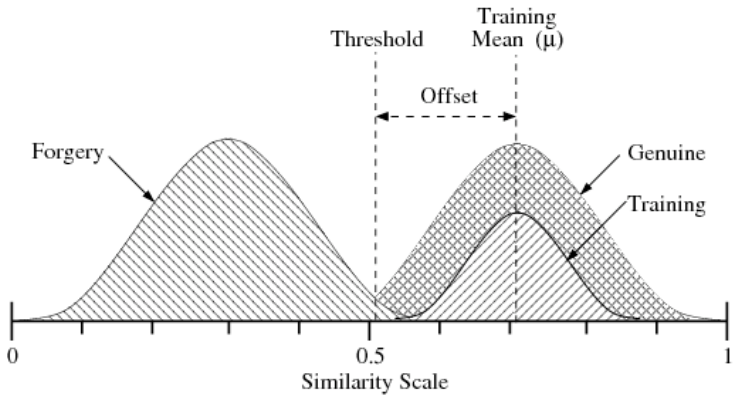


Figure 9.1: Threshold placement using the training mean and an offset

9.1.1 Manual Offset

The manual offset is a fixed value o which moves the threshold down the similarity scale. This offset is not designed to optimise the division between the genuine and forgery classes of each signature set, but is instead designed to minimise the false acceptance rate (FAR) and the false rejection rate (FRR) of the entire dataset and therefore will lie at the equal error rate (the point where the forgery and genuine curves cross). The way in which the threshold is moved down the similarity scale and used to classify an unknown signature can be seen in Equation 9.3.

$$class = \begin{cases} genuine & \text{if } score \geq \mu - o \\ forgery & \text{if } score < \mu - o \end{cases} \tag{9.3}$$

The manual offset is a common method (Kalera et al., 2004; Chen and Srihari, 2006; Srihari et al., 2008) used to calculate the classification ability of a technique in relation to an entire dataset. The disadvantage of using the manual method is that it can be fit to the test data indirectly as it requires a range of experiments, one for each possible threshold until the equal error rate is found. Therefore, the results may be overly optimistic. In relation to this, an alternative and automatic method is covered in the next section.

9.1.2 Automatic Offset

Using an automatic approach for calculating the offset removes the need for manual intervention in regards to the offset, and as such, it is a heuristic approach for finding where the threshold will lie on the similarity scale. There are potentially many ways in which the offset could be calculated, but for this research, only two were explored. These two approaches are the sample standard deviation S and the lower sample standard deviation S_L of similarity scores, when the training signatures are compared with each other. They differ in that the standard deviation S is calculated from all the training signature comparisons, while S_L is found solely from the comparisons below μ. The lower sample standard deviation is calculated

using the adaptive feature thresholding approach (Section 8.2). For the standard deviation, an unknown signature will then be classified using Equation 9.4, while the lower standard deviation is calculated in a very similar manner using Equation 9.5.

$$class = \begin{cases} genuine & \text{if } score \geq \mu - S \\ forgery & \text{if } score < \mu - S \end{cases} \tag{9.4}$$

$$class = \begin{cases} genuine & \text{if } score \geq \mu - S_L \\ forgery & \text{if } score < \mu - S_L \end{cases} \tag{9.5}$$

9.2 Random Subsets (Bagging)

The method described in Section 9.1 can be modified in a manner that will help deal with outliers that are potentially detrimental to the classification accuracy. This modification is known as bagging (Breiman, 1996) and involves taking randomly chosen subsets of the initial training signatures and using each of these subsets to produce an independent training mean. Taking the training mean from each subset and averaging them will allow an overall training mean to be produced, where this mean is the basis for calculating the threshold. Bagging is beneficial in the fact that it tries to remove to some degree the effect that possible outliers have on the classification accuracy. This is achieved as each subset mean will usually over-fit to the training signatures. By taking the average of these means, this new average is more stable and less over-fitting will occur.

9.3 BFV Similarity Measurement

The similarity between two binary feature vectors provides the main basis for classifying an unknown signature. Fortunately, the similarity between two BFVs is easily calculated. The methods used to calculate the similarity between two BFVs relies on the four values that are found when they are compared. These values are

the count of the four possible variations at each position in the vectors, and are defined as S_{ij} $(i, j \in 0, 1)$.

Using the two vectors in Figure 9.2 as an example, the first position in V_1 and V_2 is 0 and 1 respectively, and because of this combination, the variable S_{01} is incremented by 1. Applying this approach to the second position in these vectors will result in S_{11} being incremented. Repeating this process for all n positions will produce four variables that are the frequency of the four possible combinations.

$$V_1 = 0\ 1\ 0\ 1\ 1\ 0\ 1\ 0\ 0\ 0\ 1\ 0\ 1\ 1\ 1\ 0\ 0\ 1\ 0\ 1\ 0\ 0\ 1\ 0\ 1\ 1\ 1\ 1\ 0\ 0\ 1\ 1$$
$$V_2 = 1\ 1\ 0\ 0\ 1\ 1\ 0\ 1\ 0\ 1\ 1\ 0\ 1\ 0\ 0\ 1\ 0\ 1\ 1\ 1\ 1\ 0\ 1\ 0\ 1\ 1\ 0\ 0\ 1\ 0\ 1\ 1$$

Figure 9.2: A set of binary feature vectors of length 32

If the two vectors V_1 and V_2 from Figure 9.2 were compared and the four possible variations were counted for each position, the count for each variations would be $S_{11} = 11$, $S_{00} = 7$, $S_{01} = 8$, and $S_{10} = 6$.

Using a combination of these four values, the similarity of two vectors can be evaluated using a range of different measures. Of the different similarity measures that are available, the four listed in Table 9.1 will be tested.

Measure	$\mathbf{S}(V_1, V_2)$
GSC (Chen and Srihari, 2006)	$\frac{0.5 \times S_{00} + S_{11}}{n}$
Sokal-Michener (Zhang and Srihari, 2003)	$\frac{S_{11} + S_{00}}{n}$
Russell-Rao (Zhang and Srihari, 2003)	$\frac{S_{11}}{n}$
Rogers-Tanmoto (Zhang and Srihari, 2003)	$\frac{S_{11} + S_{00}}{S_{11} + S_{00} + 2S_{10} + 2S_{01}}$

Table 9.1: BFV Similarity Measures

Zhang and Srihari (2003) conclude that from these similarity measures, the Rogers-Tanmoto and the Sokal-Michener measurements had the best discriminative power, while the Russell-Rao measure had the worst power. The GSC measure was not covered in this literature, but was used by Chen and Srihari (2006). Each of these measures will always produce a value that ranges between 0 and 1, where if the value is 0, then the two vectors are completley dissimilar, while if the value is 1, then the two vectors are either very similar or the same. Each measure is designed to enhance a particular set of characteristics, by calculating the score based on certain attributes from the two vectors. The Russel-Rao measure is the most basic, as it only takes into account the variable S_{11} in relation to the entire vector length. The GSC measure takes into account all the positions that have the same value, but only gives S_{00} half as much weighting as S_{11}. Sokal-Michener is very similar to the GSC measure, except that it gives S_{00} the same weighting as S_{11}. Rogers-Tanmoto is the most complex of the four measures, instead of using n to measure the similarity against, it penalises the similarity score twice as much for each occurrence of positions with differing feature bits. The effect that these measures have on the similarity score for the two vectors in Figure 9.2 is shown in Table 9.2.

Measure	$\mathbf{S}(V_1, V_2)$		**Result**
GSC	$\frac{0.5 \times 7 + 11}{32}$	=	0.453
Sokal-Michener	$\frac{11 + 7}{32}$	=	0.563
Russell-Rao	$\frac{11}{32}$	=	0.344
Rogers-Tanmoto	$\frac{11 + 7}{11 + 7 + 2(6) + 2(8)}$	=	0.391

Table 9.2: Similarity Example Results

Chapter 10

Two Class Classification

Two class classification is an alternative to the one class approach and has the advantage of finding a more accurate division for separating the genuine and forgery classes, instead of estimating where this division should be solely from genuine signatures. Two class classification is only an option if there is a secondary (negative) class from which the classifier can be trained from. If the primary class consists of genuine signatures, then this secondary class must consist of skilled forgeries. Examples of forgeries for a particular subject will allow the best division between the two classes to be determined. Using skilled forgeries for this secondary class will provide the most refined classifier (as a negative class comprising of random and simple forgeries will most likely lead to skilled forgeries being classified as genuine). Machine learning algorithms were utilised for the two class classification approaches that were carried out.

10.1 Feature Differences

When training a machine learning classifier with two classes, the greater the number of training instances, the more accurate the division between the two classes should be. The disadvantage in this domain is that generally there will be few forgery signatures available, perhaps only 2–4, for example. To improve upon this, the difference between every pair of signatures from each class can be used to generate more data to training from, where this difference is calculated using the frequency

vectors that represent each signature.

The difference between two frequency vectors V_1 and V_2 of length n is found as a third vector D, where each element in the vector is determined by Equation 10.1.

$$D_i = |V_{1i} - V_{2i}| \quad i = 1, 2, \cdots, n \tag{10.1}$$

Finding the differences between the genuine signatures begins with obtaining the normalised frequency vector for each signature. The difference between every pair of genuine vectors is then found, and this will produce $\frac{g \times (g-1)}{2}$ new vectors, where g is the initial number of genuine signatures.

For example, if there were 10 genuine signatures, instead of using these 10 signatures to train with, the differences between each pair of signatures will allow $\frac{10 \times (10-1)}{2} = 45$ genuine instances to be created from which the classifier can be trained.

The forgery signatures are processed in a different manner. The difference between each forgery and each genuine signature is found by producing $f \times g$ difference vectors, where f and g are the number of forgery and genuine signatures respectively. Therefore, if there was 4 forgery and 10 genuines available to train with, this would allow $4 \times 10 = 40$ negative instances from which the forgery class in the classifier could be trained. Taking differences therefore increases the amount of training data from which a model can be built.

10.2 Machine Learning

Machine learning is a field of computer science which draws on the concepts and results from many other areas, including statistics, artificial intelligence, information theory and computational complexity (Mitchell, 1997). Machine learning algorithms are designed to find structural patterns from within data, and from this data, *learn* underlying information. Different algorithms have been invented that are effective for different types of learning tasks. For the classification of signatures, the effectiveness of the machine learning approach is tested with three significantly

different algorithms, Naïve Bayes, Random Forests and Support Vector Machines, each of which are described in Sections 10.2.1, 10.2.2, and 10.2.3, respectively.

described in this chapter.

These machine learning algorithms are generalised classification techniques, in which Support Vector Machines and Random Forests are state-of-the-art, while Naïve Bayes is used as a baseline, due to being simple but effective.

10.2.1 Naïve Bayes

The naïve Bayes classifier is a practical Bayesian learning method based on Bayes' theorem and is termed *naïve* because it relies on two important simplifying assumptions. The first assumption is that the predictive attributes are conditionally independent given the class. This means that the class is defined by the attributes that it is made up from and each of these attributes are considered independent given the class; if there is any correlation between attributes, this correlation is not taken into account. The second assumption is that there is no hidden or latent attributes that may influence the prediction process, therefore, classification is determined by using only the attributes about the class that are known. Using these assumptions, naïve Bayes calculates the independent probabilities for features that have been observed in a particular class, where these probabilities can then be used to form hypotheses about each class (Witten and Frank, 2005). These hypotheses can then be used to estimate the probability that an unknown instance belongs to this class instead of making an outright prediction.

Figure 10.1 is a graphical depiction of a naïve Bayes classifier, where all arcs are directed from the class attribute to each observable and predictive attributes X, where k is the number of attributes. The class attribute C is the prediction that an unknown instance belongs to a particular class.

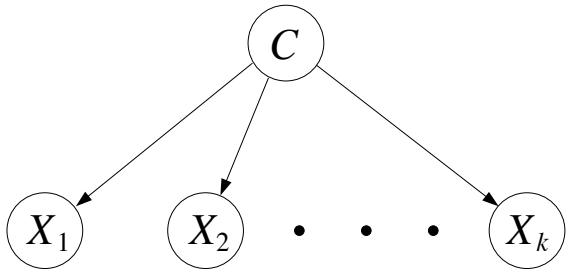


Figure 10.1: A graphical depiction of a naïve Baysian classifier

The creation of a naïve Bayes classifier first requires training data from which it will be built. Table 10.1 contains a mock-dataset as an example, where each signature is described by basic, high level attributes, where these attributes are width, height and density. Each of these attributes contain one of three possible features that describe the shape of the signature.

Signature	Class	Width	Height	Density
1	Genuine	Thin	Medium	Medium
2	Genuine	Thin	Medium	Dense
3	Genuine	Medium	Short	Dense
4	Genuine	Medium	Tall	Medium
5	Genuine	Wide	Tall	Sparse
6	Genuine	Wide	Tall	Dense
7	Forgery	Thin	Medium	Sparse
8	Forgery	Medium	Short	Dense
9	Forgery	Wide	Short	Sparse
10	Forgery	Wide	Short	Medium

Table 10.1: Example signature dataset with basic high level features

When each of these features are measured in relation to the signature class, their probability of occurring can be computed from their frequency using Bayes' rule and in turn, used to form a naïve Bayes classifier. Bayes' rule determines the correct probability P of a feature occurring (F), given its observed frequency for either the genuine or forgery class (C), with this rule being shown in Equation 10.2.

$$P(F|C) = \frac{P(C|F) \times P(F)}{P(C)} \tag{10.2}$$

Using the frequency data in Table 10.1 as the basis, the probability of each feature

occurring conditionally can be estimated. For example, the probability of the density being sparse given that the class is genuine can be calculated as

$$\begin{aligned} P(Sparse|Genuine) &= \frac{P(Genuine|Sparse) \times P(Sparse)}{P(Genuine)} \\ &= \frac{0.333 \times 0.3}{0.6} \\ &= 0.167. \end{aligned}$$

This method of calculating the feature probability is used to find the probabilities for all attributes and their corresponding features in relation to both classes, and is used to construct the probability tables shown in Figure 10.2.

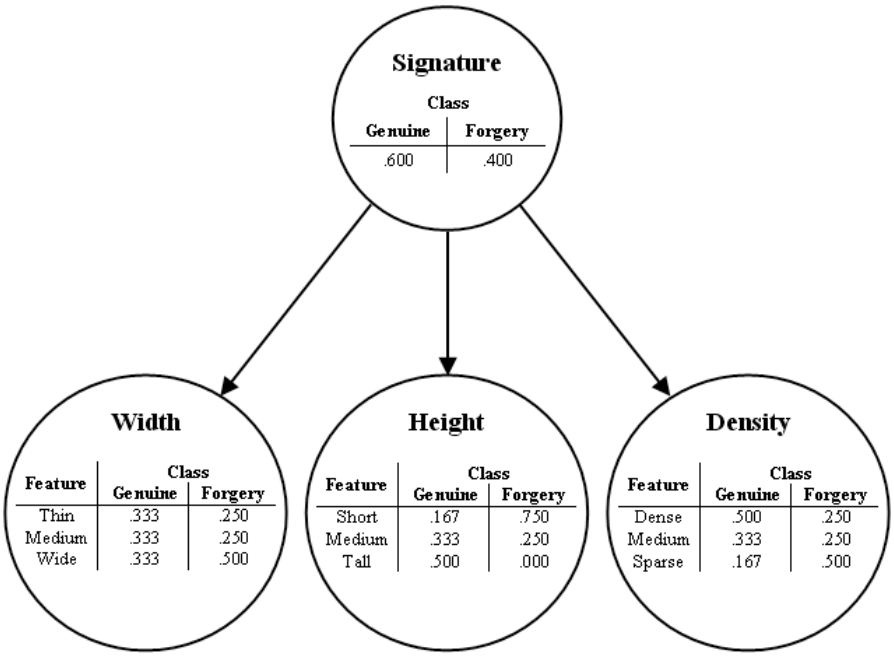


Figure 10.2: A naïve Baysian classifier modelled off of the data in Table 10.1

Using the feature probabilities from Figure 10.2, an unknown signature can then be classified by determining how likely it is to belong to either the genuine or forgery classes. For example, if a signature has the following characteristics:

Class	Width	Height	Density
?	Thin	Short	Sparse,

the first step is to find the probability of each respective feature occurring in relation to both the genuine and forgery classes.

Using the probabilities from Figure 10.2, their joint probability is calculated by multiplying them together, whilst also taking into account the overall probability of each class. The formula used to carry this out is shown in Equation 10.3, where n is the number of features.

$$P(C, F_1, F_2, \cdots, F_n) = P(C) \prod_{i=1}^{n} P(F_i|C) \tag{10.3}$$

The result of this multiplication for the genuine class is then $.6 \times (.333 \times .167 \times .167) = 0.0056$. Repeating this for the forgery class creates the probability $.4 \times (.25 \times .75 \times .5) = 0.0375$. The conditional probability of each class is found by dividing these values with their sum, which is $0.0056 + 0.0375 = 0.0431$. Therefore the probability that this example is genuine would be $\frac{0.0056}{0.0431} = 0.1299$, while its probability of being a forgery is $\frac{0.0375}{0.0431} = 0.8701$. Having followed this, the signature is deemed to belong to the class with the greater probability, which in this case is forgery.

10.2.2 Random Forests

The random forest classifier is an ensemble of decision trees built from randomly chosen attributes taken from the training dataset. This section will first describe how a decision tree is constructed starting at the root node, then the information gain method used for deciding which node to have at each level is explained, and finally how a random forest is built and used to classify an unknown instance.

Root Node

The construction of a decision tree for use in a random forest firstly requires a set of training instances from which it will be modeled, where these instances are randomly selected from a training dataset. For example, if the mock data in Table 10.1 was a collection of random signatures chosen from a training dataset and

the three attributes were only a randomly chosen subset of all possible attributes, a decision tree can be constructed by first forming each attribute into a node. The branches that extend from each node are formed from the features that this attribute could potentially be.

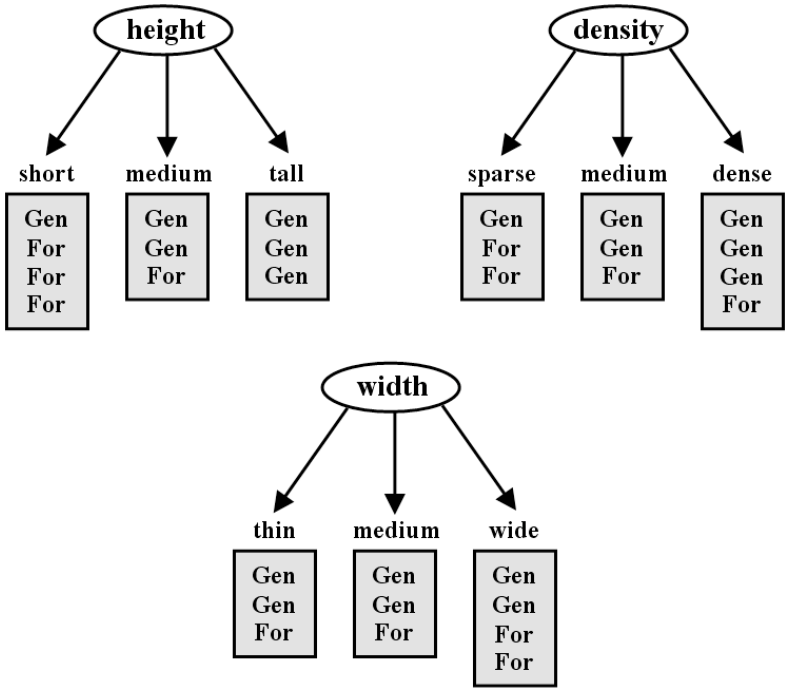


Figure 10.3: The three possible nodes, where **Gen** is genuine and **For** is Forgery

Using this selected sample, the three possibilities that the root node could be are shown in Figure 10.3. The decision of which node will be the root is found using the information gain method, and is described with greater detail later in this section. The calculated information gain for these nodes is

$$\begin{aligned} \text{gain}(\textit{height}) &= 0.1117, \\ \text{gain}(\textit{density}) &= 0.0287, \\ \text{gain}(\textit{width}) &= 0.006. \end{aligned}$$

From these values, the best attribute to choose as the root node is the height attribute, as it produces the greatest information gain.

Having found the root node, the remaining nodes of the tree are then found, a

process that is described next.

Decision Tree Construction

The construction of a decision tree is carried out in a recursive manner, where the choice of nodes at each level is found using the same process to choose the root node. The difference though, is that the information gain for each child node is calculated using only the instances that came from the parent branch. For example, if the root node was height, then there are four instances used to select the next node for the *short* branch. This recursion is stopped once the data can no longer be split. Using the three nodes from earlier, the decision tree that is constructed is shown in Figure 10.4.

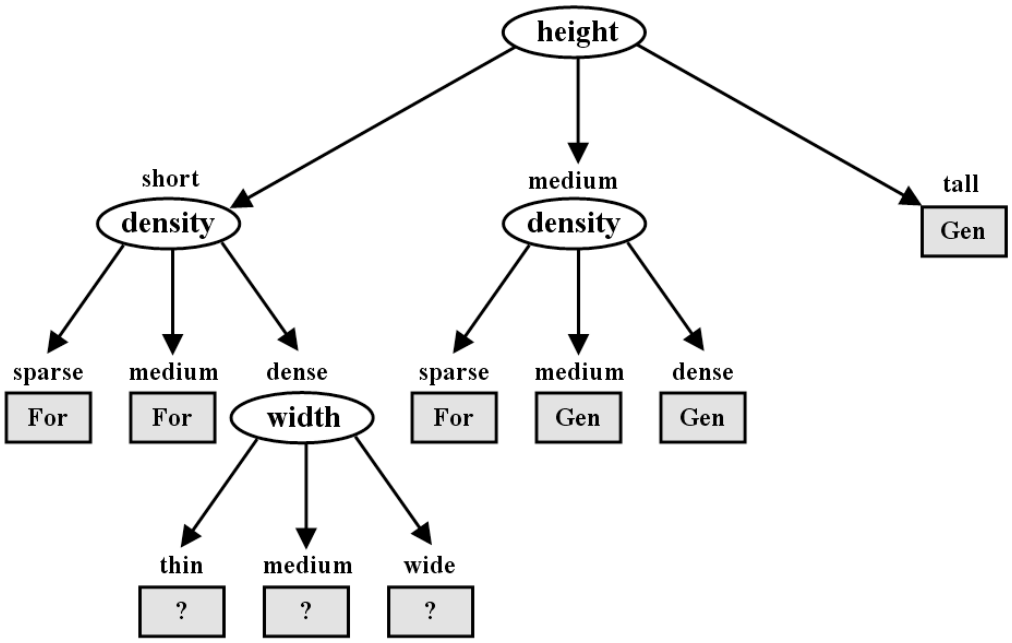


Figure 10.4: The built decision tree based on the selected sample data

Ideally, a decision tree will have leaf nodes that are all *pure*, that is, they contain instances of only one class, an example of this can be seen for the *tall* branch for the height node in Figure 10.4. Unfortunately, this is not always the case, as indicated in Figure 10.4 by the leaves with a '**?**'. These missing classifications are primarily caused by missing values, which may prevent an instance from being classified.

There is also the possibility of multiple instances having the same features, yet being from different classes. Solutions for both of these situations exist, but their scope is outside of this discussion; Witten and Frank (2005) outline this with more detail.

Information Gain

Determining which node should be chosen for each level of the decision tree is carried out using the information gain method (Witten and Frank, 2005), where information gain evaluates the purity of the node in being able to provide information that will allow an unknown signature to be classified. This measure of expected information conforms to two initial premises:

1. If the frequency of either class is zero, the information value is zero.
2. If the frequency of both classes is equal, the information value is at a maximum.

The way in which the information gain is calculated for an individual leaf is shown by Equation 10.4, where a and b are the frequency of *genuine* and *forgery* classes at the leaf node respectively. Generally the information gain is calculated in base 2, with the resulting value being in fractions of bits, but for the sake of simplification, base 10 will be used.

$$\text{info}([a,b]) = -\frac{a}{a+b} \times log\frac{a}{a+b} - \frac{b}{a+b} \times log\frac{a}{a+b} \tag{10.4}$$

Using the height attribute from Figure 10.3 as an example, the frequency of each class for the leaf nodes is [1,3], [2,1] and [3,0] for the short, medium and tall features respectively. The information gain that each leaf node provides is shown in Figure 10.5.

$$\text{info}([1,3]) = -\tfrac{1}{4} \times log\tfrac{1}{4} - \tfrac{3}{4} \times log\tfrac{3}{4} = 0.2442,$$

$$\text{info}([2,1]) = -\tfrac{2}{3} \times log\tfrac{2}{3} - \tfrac{1}{3} \times log\tfrac{1}{3} = 0.2764,$$

$$\text{info}([3,0]) = 0.$$

Figure 10.5: Information gain provided by each leaf node for the height attribute

Having calculated the information gain for each leaf, the average information gain that the attribute node provides towards the classification is found by Equation 10.5. This equation is indicative of a two branch node, but can have as many branches as necessary. The variable c is the total number of instances.

$$\text{info}([a_1, b_1],[a_2, b_2]) = \frac{a_1 + b_1}{c} \times \text{info}([a_1, b_1]) + \frac{a_2 + b_2}{c} \times \text{info}([a_2, b_2]) \qquad (10.5)$$

Following on from the previous example, the height node consists of 10 instances, with four of these going to the short feature, while the other two branches have three each. Using these values, the average information gain that the height node achieves is shown in Figure 10.6.

$$\text{info}([1,3],[2,1],[3,0]) = \tfrac{4}{10} \times 0.2442 + \tfrac{3}{10} \times 0.2764 + \tfrac{3}{10} \times 0 = 0.1806.$$

Figure 10.6: Average information gain for the height features

Before the gain provided by this node can be calculated, the total information value of the selected sample needs to be determined, so seeing as the sample comprised of six genuine and four forgery signatures, the information gain for the sample is info([6,4]) = 0.2923. From this, the information gain of the height node is shown in Figure 10.7.

$$\text{gain}(\textit{height}) = \text{info}([6,4]) - \text{info}([1,3],[2,1],[3,0]) = 0.2923 - 0.1806 = 0.1117.$$

Figure 10.7: Information gain for the height node

This process of calculating the information gain is then repeated for each of the other attributes, with the attribute that has the most information gain always

being the one to split on.

Random Forests

Using only one tree for classification is ineffective, as each tree is constructed from a random subset of attributes taken from instances which have been randomly selected from a training dataset. This method of attribute selection causes each node in a tree to be split based on suboptimal criterion (Topi et al., 2005; Witten and Frank, 2005). However, if many trees are grown using this process, each tree will contributes a small amount of detail about each class as a whole, improving classification. For an unknown instance, this is achieved by having each tree in the forest vote as to which class the instance should belong. The mode of these votes is used to determine what the final classification should be.

10.2.3 Support Vector Machines

The support vector machine (SVM) is a linear classifier that when given a set of training data that has N attributes will construct an $N-1$-dimensional maximum margin hyperplane that optimally separates this data. This optimal hyperplane location is found by first constructing a boundary that passes through the instances that form the tightest enclosing convex polygon, with the instances that are the closest to the opposing class boundary being called *support vectors*. Both classes will have at least one support vector each, though often there will be more. It is on the section of boundary defined by these support vectors that the shortest line between the two classes will lie. By perpendicularly bisecting this line, the hyperplane will maximise the margin between the two classes. This is seen in Figure 10.8, where the shortest distance between the two boundaries is indicated with a dashed line.

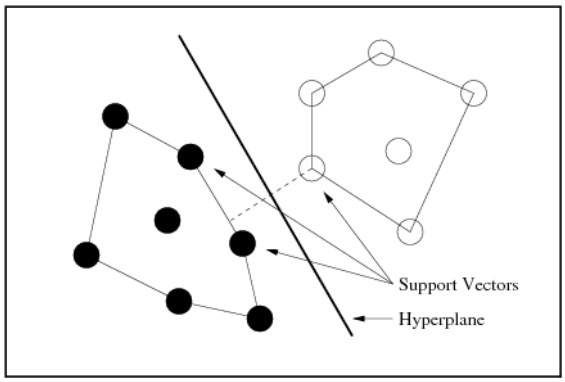


Figure 10.8: A hyperplane splitting two classes in the input space

Often though, the input space of the instances will not be ideal for constructing a hyperplane, due to the fashion in which they are situated within the input space. An example of this situation can be seen in Figure 10.9.

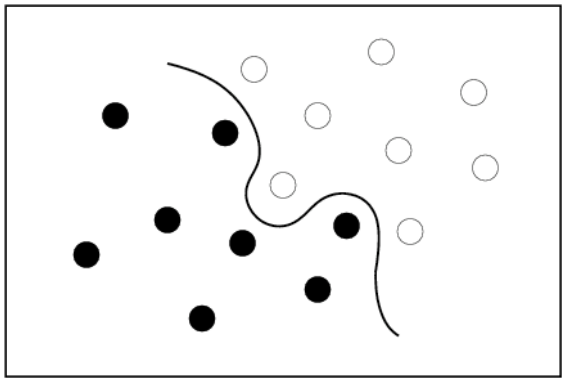


Figure 10.9: Two classes in the input space that a hyperplane cannot separate

In the cases where a hyperplane cannot split the data, the instances in the input space are mapped to a different feature space that allows the two classes to be linearly separated. This mapping is carried out using mathematical functions known as kernels, such as the radial basis function (RBF) and the sigmoid kernel. The mapping of these instances to a new feature space requires the solution to a very large quadratic programming (QP) optimisation problem, which these kernels can solve. The manner in which these kernels solve the QP problem is outside the scope of discussion here, though more information can be found in (Witten and Frank, 2005; Platt, 1999).

Chapter 11

Signature Datasets

A signature dataset is a collection of signatures containing both genuine and forgeries for a range of different people. Having both genuine and forgeries allows for algorithms to be tested for the purpose of determine whether they can correctly classify the signature class. Using the same database allows for different algorithms to be tested against each other, so that from this, the most effective algorithm can be determined by comparing the produced classification accuracies. For off-line signature verification there is a limited number of datasets available, the result of this is that for the evaluation experiments, only two datasets could be obtained, with these being CEDAR and GPDS.

11.1 CEDAR

The Center of Excellence for Document Analysis and Recognition (CEDAR) signature dataset[1] (Kalera et al., 2004) is a commonly used dataset for off-line signature verification. The CEDAR dataset consists of 55 signature sets, with each set being composed by one writer. Each writer provided 24 samples of their signature, where these samples constitute the genuine portion of the dataset. The forgeries for this dataset were obtained by asking arbitary people to skillfully forge the signatures of the previously mentioned writers. In this fashion, 24 forgery samples were collected per writer from about 20 skillful forgers. In total, this dataset contains 2,640

[1]The CEDAR dataset can be found at http://www.cedar.buffalo.edu/NIJ/publications.html (December 2008)

signatures, built from 1,320 genuine signatures and 1,320 forgeries, which are of the skilled variety (Section 1.5). The writers of both the genuine and forgery signatures were asked to sign in a predefined space measuring 2×2 inches. The input signatures were scanned at 300 dpi in 8-bit grey scale (which produces 256 shades of grey) and were stored in the Portable Network Graphics (PNG) format. Figure 11.1 and 11.2 show 10 examples for both the genuine and forgery signatures for one writer respectively. Figure 11.3 shows an example of each signatures in their binarised state.

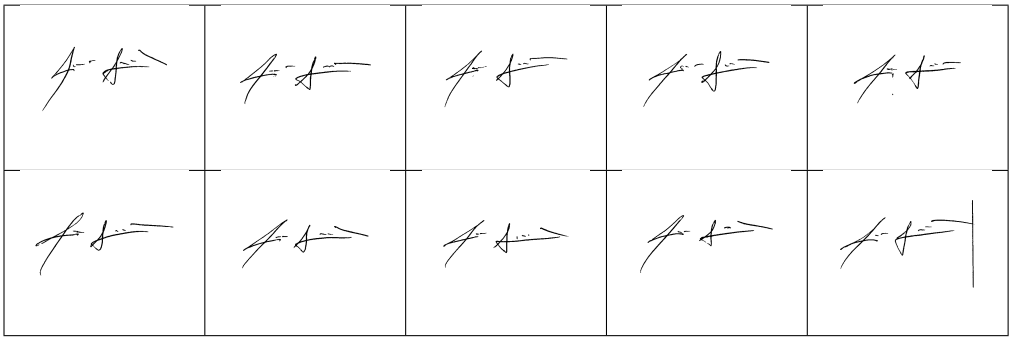


Figure 11.1: CEDAR genuine samples for one writer

Figure 11.2: CEDAR forgery samples for the same writer in Figure 11.2

Figure 11.3: Sample of each signature in the CEDAR dataset

11.2 GDPS

The Grupo de Procesado Digital de Senales (GPDS) signature dataset[2] is another dataset that has been used in literature (Martinez et al., 2004; Armand et al., 2006). This dataset consists of 39 signature sets, where each set has been composed by one writer, who provided 24 samples of their signature for the genuine component of the dataset. The forgeries were written by three forgers, each of whom were allowed to practise the signature for as long as they wish, with these forgers then imitated five genuine signatures three times. This produced 30 skilled forgeries for each set, which came from 10 forgers. In total, the GPDS dataset contains 2,106 signatures, which is built from 936 genuine signatures and 1170 forgeries. At the time this dataset was obtained, only 39 signature sets were available, since then, it has grown to have 300 signature sets. For testing, only the original 39 sets will be used. Figure 11.4 and 11.5 show 10 examples of both the genuine and forgery signatures for one writer respectively. Figure 11.6 shows examples of signatures from this dataset in their unmodified state.

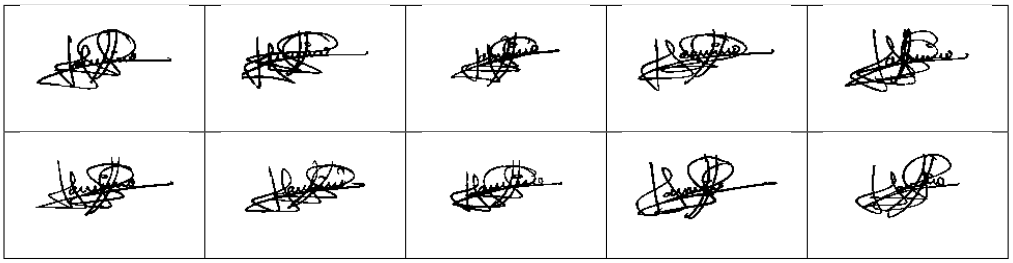


Figure 11.4: GPDS genuine samples for one writer

Figure 11.5: GPDS forgery samples for the same writer in Figure 11.4

[2]The GPDS database can be found at http://www.gpds.ulpgc.es/download/index.htm (April 2008)

Figure 11.6: Sample of each signature in the GPDS dataset

Chapter 12

Evaluation: Preprocessing

When classifying signatures, there are a variety of preprocessing techniques that can be applied to each signature image before features are extracted. This chapter will evaluate each preprocessing technique described in Chapter 3, and will determine the impact that they have on classification accuracy. This evaluation of these techniques is carried out with feature extraction and classification remaining fixed, with only the preprocessing techniques being varied.

12.1 Experiment Design

The evaluation of a preprocessing technique begins by first applying it to the entire signature dataset. Having carried this out, each signature is then cropped so that there is no excess padding. The other parameters that influence the classification will remain fixed, ensuring that the accuracy is fairly compared between preprocessing techniques. The progression from the first experiment to each consecutive one will utilise the most effective methods based on the achieved results, with the goal of finding the most advantageous combination of parameters.

The process that is carried out for each experiment begins with taking signatures (the number is dependent on the dataset) from a particular set to use as training. Following this, a single image feature is then extracted from each training and testing signature in the set. For the preprocessing evaluation four features will be tested, these features are mass, gradient with 18 segments, standard local

binary patterns with a radius of 1, using 8 points and concavity features with diagonal opening concavities. The choice of these features is due to their similarity with the remaining features, which should indicate the affect that preprocessing techniques have on image features in general. In relation to feature extraction, an equimass grid of 8×4 in size was chosen without the use of spatial pyramids as this combination was used by Chen and Srihari (2006). Having extracted the chosen image feature, it is then represented by a normalised frequency histogram.

The conversion of the frequency histogram to a binary feature vector is carried out using the adaptive feature thesholding method described in Section 8.2. Each signature is then classified using one class classification with a manual offset, while the GSC distance measure is used as the similarity score (Section 9.3).

This experiment process is carried out on each CEDAR signature set by randomly choosing 16 signatures for training, while the remaining 8 genuine and 24 forgery signatures will be used for testing. These experiments are then repeated 10 times each to determine a reliable estimate of the accuracy resulting from one of the preprocessing techniques being tested. Each accuracy is found as the middle point between the genuine and forgery classification accuracies and is calculated by Equation 12.1, where FAR is the false acceptance rate and FRR is the false rejection rate.

$$accuracy = \frac{(1 - FAR) + (1 - FRR)}{2} \tag{12.1}$$

The GPDS dataset will not be tested in the preprocessing evaluation, as it is already in a binarised format with each signature having been correctly orientated.

12.2 Image Binarisation

The manner in which a signature is binarised from grey-scale to binary can effect the signature structure. Because of this, four different binarisation methods will be tested to determine how they impact the classification accuracy. The methods that will be tested are fixed, iterative, Otsu and local iterative, each of which are

described in Section 3.1. The fixed threshold is being used as a baseline and will have its threshold set at 220. The choice of 220 is that from observation, it tends to binarise the CEDAR dataset reasonable well.

12.2.1 Results

The testing of these binarisation methods followed the experimental design as described in Section 12.1. From the results in both Table 12.1 and the corresponding graph in Figure 12.1, there is no distinct binarisation method that consistently achieves greater results for each image feature. The fixed method was the most effective for both the mass and gradient features, while Otsu and local iterative produced the best results for the LBP and concavity features respectively. The iterative method did not produce the best results for any of the image features, but tended to have a decent accuracy for each feature.

Method	Mass	Gradient	LBP	Concavities	Average
Fixed	**75.28** ± 0.71	**88.34** ± 0.87	73.96 ± 1.11	85.48 ± 1.01	**80.77**
Iterative	72.28 ± 1.00	87.57 ± 0.57	75.80 ± 0.68	85.39 ± 0.86	80.26
Otsu	72.61 ± 0.92	88.12 ± 0.57	**75.98** ± 0.94	85.86 ± 0.90	80.64
Local Iterative	71.33 ± 0.70	84.79 ± 0.76	75.31 ± 1.24	**85.91** ± 0.86	79.33

Table 12.1: Image binarisation results

The accuracy achieved by each feature has a tendency of varying based on different binarisation methods. For example, using mass with fixed binarisation has a 2.67% lead over the next best method, while using fixed with LBPs had the opposite effect, with it being 1.35% lower than the next highest result. If the accuracy from each image feature is averaged, then the fixed method achieved the highest result at 80.77%, though this is only marginally higher than the other three methods, with Otsu and iterative placing second and third.

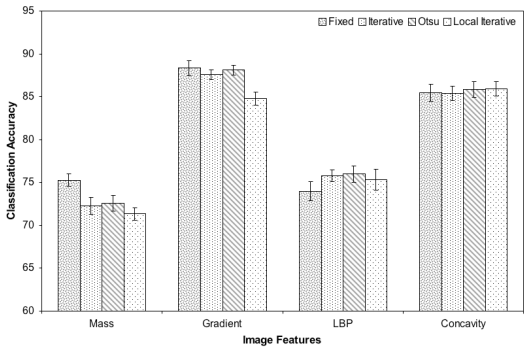


Figure 12.1: Graphed results for image binarisation

From the results provided here, it is shown that the binarisation methods tend to vary based on the image features, with no method being consistently better. Because of this, each method will be utilised and evaluated under signature reconstruction, as just using the thresholded image is not enough to determine their impact upon the classification accuracy.

12.3 Signature Reconstruction

Often when the signature is converted from grey scale to binary, sections of the signature will be incorrectly binarised. To deal with this, four signature reconstruction techniques were investigated. These four techniques are the square and plus shaped median filters (SSF and PSF respectively), dilate and erode (DE), and region growing (RG) each of which are described in Chapter 3. These techniques will be applied to the four binarisation methods used previously. The purpose of this is to determine both the effectiveness of each reconstruction method as well as the ability of the binarisation methods. From these reconstruction and binarisation methods, the four combinations that fair the best will be tested in conjunction with rotation normalisation. In regards to this, fixed binarisation will not be included, as in practise, it will only be effective for signatures that are always captured in the same manner, as a slight shift in image brightness will result in the image being incorrectly binarised.

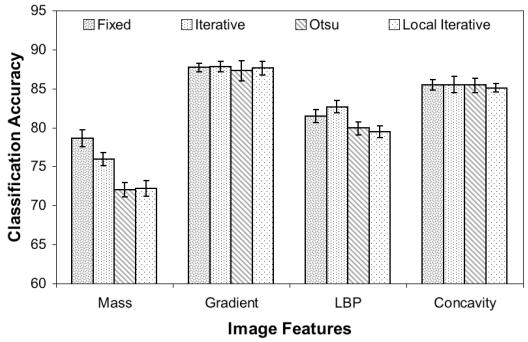
12.3.1 Results

Following the experimental design of section 12.1, signature reconstruction was tested using the combination of each binarisation method with the four reconstruction methods, creating 16 variations of the CEDAR set that will be tested initially. The results of these 16 experiments are shown in both Table 12.2 and the graphs in Figure 12.2. Each accuracy is the average of the four image features that were tested. Appendix A shows the full results in Tables 16.1, 16.2, 16.3 and 16.4.

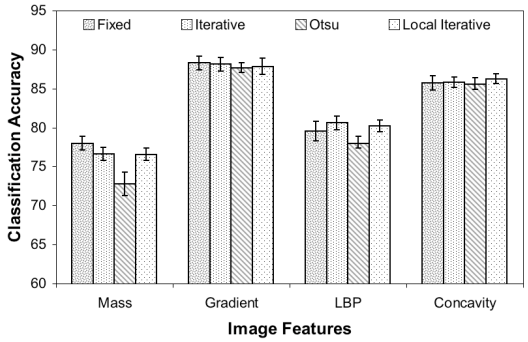
	Fixed	Iterative	Otsu	Local Iterative	Average
SSF	**83.33**	82.98	81.19	81.10	82.15
PSF	82.88	82.80	81.01	82.72	**82.35**
DE	82.71	82.94	82.43	80.27	82.09
RG	80.96	80.81	80.70	77.87	80.09
Average	**82.47**	82.38	81.33	80.49	

Table 12.2: Results from reconstruction and binarisation combination

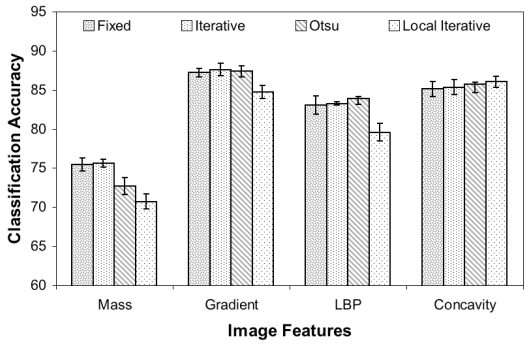
These results show that on average the plus shaped filter produces the best accuracy, though in general, the square shaped filter performs better, except in regards to local iterative. Dilate and erode also fair exceptionally well in regards to SSF and PSF, though region growing on the other hand is below dilate and erode by 2% at 80.09%. In terms of the binarisation capability, the fixed method surprisingly achieved an accuracy that is greater than the other three methods. The validity of the fixed method though is undetermined, as in practise there is no way of guaranteeing that each signature will be captured in the same manner, as factors such as brightness may change, causing the signature to be binarised incorrectly. From the other three binarisation methods, iterative faired the best, followed by Otsu and then local iterative. There is approximately a 1% difference between each of these results.



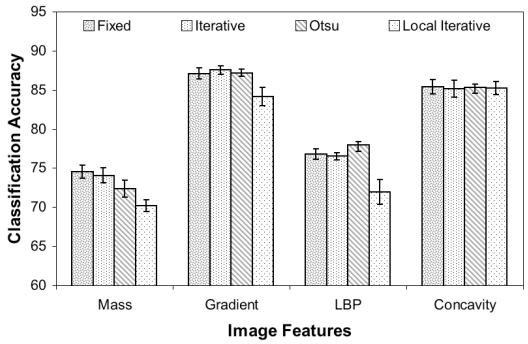
(a) Square shaped median filter



(b) Plus shaped median filter



(c) Dilate and erode



(d) Region growing

Figure 12.2: Graphed results for the binarisation and reconstruction techniques

Local iterative so far has produced results that are lower than expected. This is possibly because local iteration tends to produce noise around the signature, adversely affecting the ability of DE and RG. To combat this, the two median filters are tested in combination with either DE or RG are tested. The results of testing these methods together are shown in Table 12.3 and in Figure 12.3. The full result table is available under Appendix A in Table 16.5.

	Dilate and Erode	Region Growing
Square Shaped Filter	82.54	81.00
Plus Shaped Filter	82.64	81.18

Table 12.3: Results from local iterative using dual reconstruction

By applying dual reconstruction to local iterative, the results are improved in regards to DE and RG, but in general, they do not produce any improvement over

just SSF and PSF.

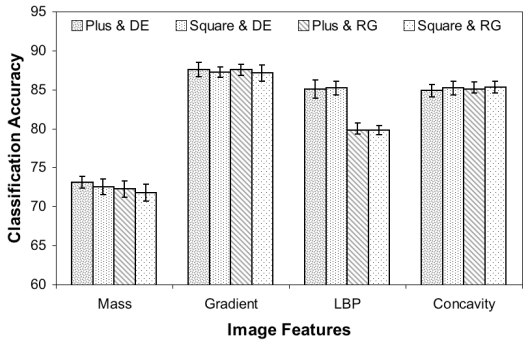


Figure 12.3: Local iterative using dual techniques

From the produced results, the reconstruction and binarisation techniques are selected based on the average ability of the four features. Using this average helps ensure that the chosen method is robust, instead of performing well for a single feature. When the average ability is ranked in order from the highest to lowest, the most effective combination can be found. Table 12.4 shows the top six combinations, with the full list being available in Table 16.6 found in Appendix A.

Reconstruction	Threshold	Mass	Gradient	LBP	Concavities	Average
Square	Fixed	**78.65**	87.70	81.47	85.49	**83.33**
Square	Iterative	75.95	87.82	82.65	85.50	82.98
DE	Iterative	75.62	87.58	**83.23**	85.34	82.94
Plus	Fixed	77.96	**88.29**	79.54	85.73	82.88
Plus	Iterative	76.63	88.14	80.61	85.81	82.80
Plus	Local Iterative	76.57	87.83	80.24	**86.25**	82.72

Table 12.4: Top results from reconstruction and binarisation combination

Because the fixed method is only a baseline, it will not be considered, therefore, the remaining four methods are found to be the best combination of preprocessing techniques, and as such will be test in the following experiments. These methods are iterative binarisation with each reconstruction type except region growing, and local iterative with the plus shaped median filter.

12.4 Rotation Normalisation

When signatures are written on paper, they be written at different orientations. The result of this, is that when features are extracted from each signature, their values may differ substantially, unless the features are rotation invariant. Rotation normalisation is designed to re-orientate each signature in a set, so that they each have the same alignment. The two tested methods are the axis of least inertia and region rotation, with both being described in Section 3.3.

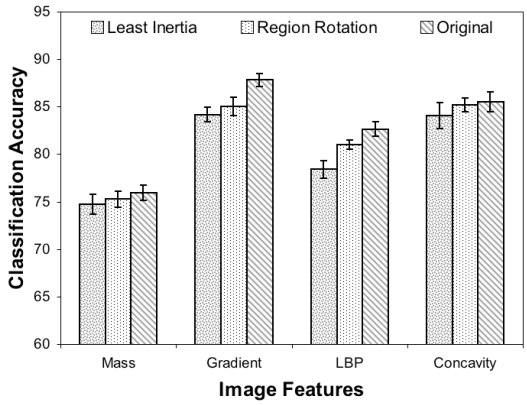
12.4.1 Results

Applying rotation normalisation to the four reconstruction methods will produced eight additional variations to the CEDAR dataset. The effect that using rotation normalisation can be seen in Table 12.5 and Figure 12.4. The full results from these experiments can be found in Tables 16.7, 16.8, 16.9 and 16.10 in Appendix A.

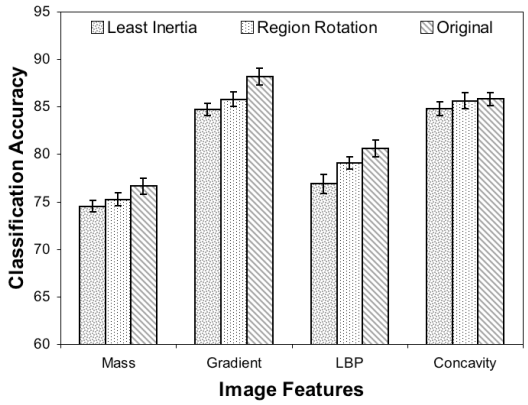
	Iterative			Local Iterative
Method	SSF	PSF	DE	SSF
Original	**82.98**	**82.80**	**82.94**	**82.72**
Least Inertia	80.33	80.22	80.66	76.62
Region Rotation	81.61	81.43	78.17	78.78

Table 12.5: Average rotation normalisation results

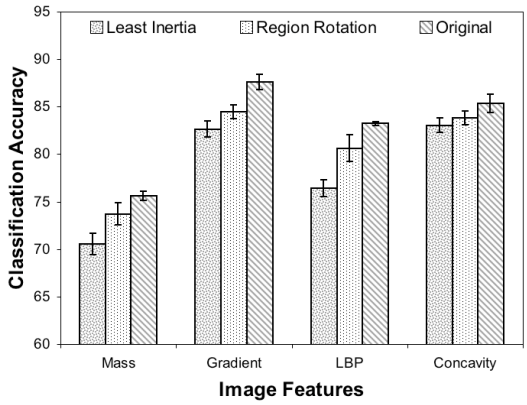
The two rotation normalisation methods, as shown in these results, do not increase the classification accuracy under any of the tested circumstances. In general, the unchanged or original dataset produces results that have an improvement of 1% to 4% over the corresponding rotated datasets. In regards to their ability, region rotation faired better than least inertia, though it does have the disadvantage of being computationally more complex.



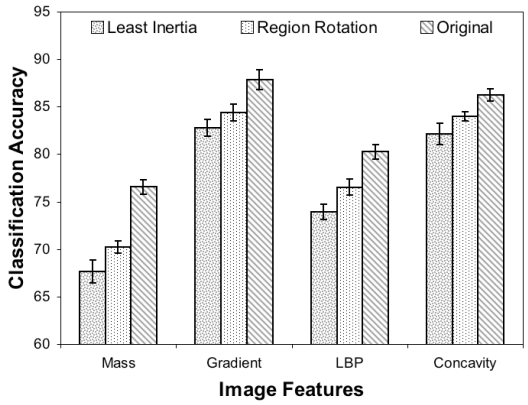
(a) Iterative with square filter



(b) Iterative with plus filter



(c) Iterative with dilate and erode



(d) Local Iterative with plus filter

Figure 12.4: Graphed results for rotation normalisation

The reason why rotation normalisation had a drop in accuracy is hard to explain, as there are individual signature sets that benefit from this rotation. Essentially, there are two possibilities, firstly, rotating of the signatures may damage the underlying structure of a signature, making it more difficult to differentiate from other signatures. The second is that the orientation of the signature could be an attribute of how an individual writes, and as such, this attribute helps to uniquely distinguish the genuine signatures in a set from the forgeries. Because of this decrease in accuracy, rotation normalisation will not be applied to the CEDAR dataset in subsequent experiments.

Chapter 13

Evaluation: Feature Extraction

The extraction of features from each signature is a necessary step in the classification process, as features help to uniquely differentiated signatures from each other. The disadvantage of many features though is that they do not provide any spatial information in regards to where they were taken from within the image; this is where region sampling and spatial pyramids should prove to be beneficial. This chapter will evaluate the image features described in Chapter 4 as well as determining the impact that region sampling and spatial pyramids have on the final classification accuracy.

13.1 Experiment Design

The evaluation of the image features and the spatial techniques will be carried out using the best preprocessing techniques found in the previous chapter. Following on from the experiment design described in Section 12.1, the manner in which each experiment is carried out will remain fixed, while only the image features will vary. To reiterate this experimental design, each consecutive experiment will utilise the most effective parameters found from testing. Classification will be carried out in the same manner as the preprocessing evaluation, with each feature being represented as a normalised frequency histogram. The conversion of this frequency histogram to a binary feature vector is carried out by adaptive feature thesholding (Section 8.2). Each signature is then classified using one class classification with

a manual offset, while the GSC distance measure (Section 9.3) is used for the similarity score. This experiment process will be carried out on each CEDAR and GPDS signature set, where CEDAR will use 16 randomly chosen signatures for training and GPDS will use 12. The remaining signatures for each particular set will be used for testing. The choice of 16 training signatures for CEDAR is to make the results comparable to the work by Chen and Srihari (2006); Srihari et al. (2004) and Chen and Srihari (2005). As for GPDS, the choice of 12 training signatures is to make the results comparable to Tian et al. (2007). This experiment process is repeated 10 times to determine a reliable estimate of the accuracy.

13.2 Image Features

The experiments in this section will determine the effectiveness of six different types of image features, with these being, mass, gradient, local binary patterns, concavities, structural rules and long strokes. The gradient feature will be tested using 12 and 18 segments. The choice of 12 segments is common in literature (Favata and Srikantan, 1996; Kalera et al., 2004; Srihari et al., 2004), but 18 segments have also been used Srikantan et al. (1996). Local binary patterns will be tested with all three variations; these being, standard, rotation invariant and uniform. The concavity feature will be tested with and without concavities opening in the diagonal directions. In total, ten features will be tested.

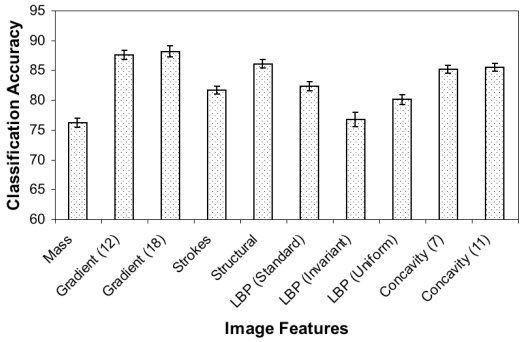
13.2.1 CEDAR Results

Testing of these image features was carried out using the experimental design described in section 13.1. CEDAR will be tested using the four binarisation and reconstruction combinations that were selected in Section 12.3.1. The results that were produced using these combinations can be seen in both Table 13.1 and Figure 13.1.

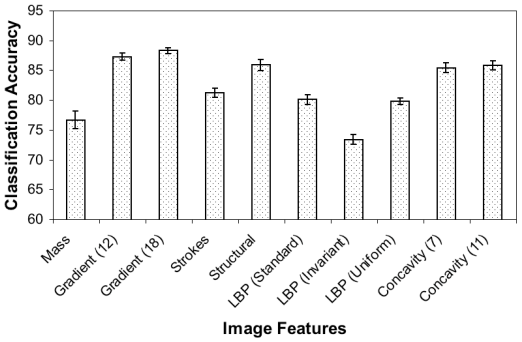
Feature	Iterative SSF	Iterative PSF	Iterative DE	Local Iterative PSF	Average
Mass	76.22 ± 0.76	76.67 ± 1.50	74.39 ± 1.05	76.00 ± 1.24	75.82
Gradient (12)	87.57 ± 0.72	87.28 ± 0.64	86.38 ± 0.82	87.80 ± 0.83	87.26
Gradient (18)	**88.14** ± 0.90	**88.28** ± 0.46	**87.40** ± 0.68	**88.20** ± 0.89	**88.01**
Strokes	81.68 ± 0.66	81.22 ± 0.72	85.29 ± 0.55	81.29 ± 0.68	82.37
Structural	86.06 ± 0.69	85.89 ± 0.91	84.94 ± 0.99	85.77 ± 0.72	85.67
LBP (Standard)	82.29 ± 0.79	80.09 ± 0.83	83.13 ± 1.00	79.91 ± 0.65	81.36
LBP (Invariant)	76.75 ± 1.20	73.39 ± 0.85	75.48 ± 0.43	73.35 ± 1.24	74.74
LBP (Uniform)	80.09 ± 0.80	79.80 ± 0.56	76.78 ± 1.09	79.87 ± 0.86	79.14
Concavity (7)	85.16 ± 0.64	85.41 ± 0.82	84.82 ± 0.96	84.95 ± 0.62	85.09
Concavity (11)	85.49 ± 0.63	85.79 ± 0.79	85.42 ± 0.73	85.58 ± 0.54	85.57
Average	**82.95**	82.38	82.40	82.27	

Table 13.1: CEDAR results when tested with different image features

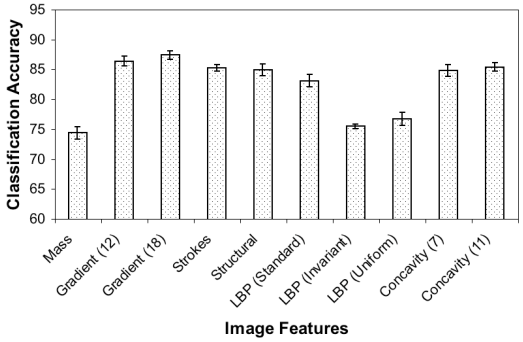
Each image feature that was tested utilises different aspects of the signature allowing them to be differentiated from one another, with the classification ability being dependent upon these aspects. From the results that are shown, there is a large variation of 14% between the lowest ranked feature (LBP Invariant) and the highest ranked (Gradient with 18 segments). From this ranking the four features that are the most effective for binary signatures in the CEDAR dataset can be seen, with Gradient (18) achieving an accuracy of 88.01%, followed by Concavity (11), Structural and Strokes. These four features will be used for the following experiments that will be performed on the CEDAR dataset. Gradient (12) and Concavity (7) are not included as they are slight variations on Gradient (18) and Concavity (11) respectively. Gradient (12) assigns each gradient direction to one of 12 segments, while Concavity (7) does not include diagonal open concavities. Of the four binarisation techniques, the combination of iterative and square shaped median filter produced the highest result, and because of this, iterative SSF will be used for all subsequent testing that is performed on the CEDAR dataset.



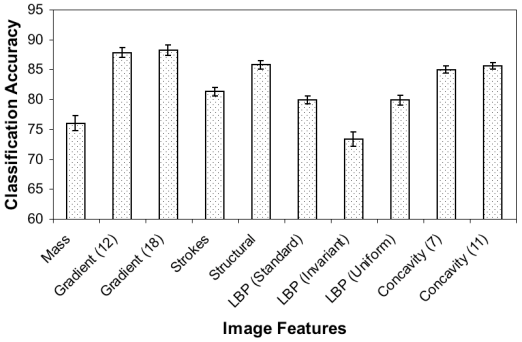
(a) Iterative with SSF



(b) Iterative with PSF



(c) Iterative with DE



(d) Local iterative with PSF

Figure 13.1: Binarisation and reconstruction for the CEDAR dataset

13.2.2 GPDS Results

Testing of the GPDS dataset was followed in the same fashion as CEDAR, though GPDS was tested in its original form instead of having any preprocessing methods applied. The ability of each image feature in regards to the GPDS dataset is shown in Table 13.2 and Figure 13.2.

Feature	GPDS
Mass	80.04 ± 1.40
Gradient (12)	82.20 ± 0.58
Gradient (18)	82.44 ± 0.48
Strokes	82.44 ± 0.54
Structural	81.92 ± 0.64
LBP (Standard)	81.83 ± 1.07
LBP (Invariant)	80.96 ± 1.13
LBP (Uniform)	84.18 ± 1.02
Concavity (7)	86.01 ± 0.87
Concavity (11)	**88.59** ± 0.55
Average	83.11

Table 13.2: GPDS feature results

In relation to the CEDAR dataset, each image feature tends to be ranked the same in distinguishing ability, though there are some variations. Concavity features replace gradient as the highest ranked, with LBP uniform also achieving a greater result than the gradient feature. The variability has diminished between the highest (Concavity 11) and lowest (Mass) ranked features, with this being approximately 8%. It is uncertain why there was a change in the feature ranking, but was not investigated due to there being a variety of possible causes.

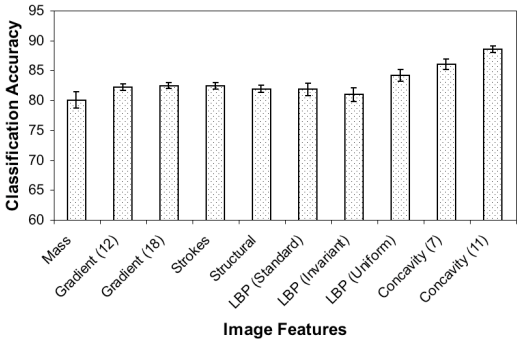


Figure 13.2: Comparison of each image feature for the GPDS dataset

From these features, Concavity (11), Gradient (18), Strokes and Structural are the four unique features that were chosen to be used for the remaining experiments. LBP uniform though achieved approximately 2% higher than gradient (18) making

it the better choice. The decision to use gradient over LBP uniform is to keep the features the same between the datasets, and also due to the GSC ensemble utilising the gradient direction instead of LBP.

13.3 Region Sampling

Region sampling was used in conjunction with feature extraction allowing spatial information to be associated with each feature. There are several methods of applying region sampling to an image, with these being the uniform, equimass and the centre of mass (COM) methods, each of which are described in Chapter 5. As well as this, ring region sampling is introduced. This alternative form of regioning is designed to allow the spatial properties to be rotation invariant.

13.3.1 CEDAR Results

The CEDAR dataset was tested with the two different region sizes, 8×4 and 4×4; the choice of these sizes is due to both having been used in literature (Chen and Srihari, 2006; Kalera et al., 2004). The first value corresponds to the number of regions across, while the second is the number of regions down. The effect that region sampling has on the accuracy is shown by contrasting it with the accuracy produced when no spatial information is used. These results are shown in Table 13.3 and Figure 13.3. The testing that was carried out also utilised spatial pyramids, the results of which will be discussed in Section 13.4.

In general, the ability of uniform regioning is an improvement over not regioning, with there being as much as a 2% increase without spatial pyramids. Both the equimass and COM sampling techniques have almost a 10% increase over uniform, automatically making it irrelevant for all future experiments. On average, equimass has a slightly higher accuracy than COM for each of the tested region and pyramid sizes. The grid size used also has an effect on the final result, with 8×4 generally producing an increase of about 1% over 4×4 for the equimass and COM sampling regions. Based on this, an equimass spatial pyramid using an 8×4 grid will be

utilised for the remaining experiments.

Region	P	Size	Gradient	Structural	Strokes	Concavities	Average
N.A.	1	1×1	79.09 ± 0.90	74.12 ± 1.06	66.16 ± 0.77	73.66 ± 0.87	73.26
Uniform	1	4×4	74.46 ± 1.01	74.09 ± 0.59	68.82 ± 0.86	73.42 ± 1.17	72.70
		8×4	77.20 ± 0.76	75.93 ± 0.81	72.72 ± 0.72	77.19 ± 0.89	75.76
	3	4×4	78.02 ± 0.79	77.34 ± 1.04	71.66 ± 0.97	76.57 ± 0.90	75.90
		8×4	80.11 ± 1.12	78.68 ± 0.96	75.85 ± 1.23	79.57 ± 1.07	78.55
Equimass	1	4×4	87.38 ± 0.95	85.75 ± 1.13	78.95 ± 1.10	84.97 ± 0.76	84.26
		8×4	87.48 ± 0.98	85.77 ± 0.85	81.83 ± 1.05	85.34 ± 0.99	85.11
	3	4×4	87.93 ± 0.79	86.35 ± 0.96	81.03 ± 0.59	85.81 ± 1.03	85.28
		8×4	**89.04** ± 0.56	87.00 ± 0.50	**84.17** ± 0.79	**86.48** ± 0.80	**86.67**
COM	1	4×4	87.41 ± 0.63	84.92 ± 0.87	80.47 ± 0.96	84.71 ± 0.83	84.38
		8×4	88.79 ± 0.69	86.27 ± 0.65	81.18 ± 1.34	85.67 ± 0.94	85.48
	3	4×4	87.94 ± 0.75	85.89 ± 0.59	81.60 ± 0.90	84.95 ± 0.79	85.10
		8×4	88.91 ± 1.10	**87.39** ± 0.61	83.41 ± 1.03	86.08 ± 1.10	86.45

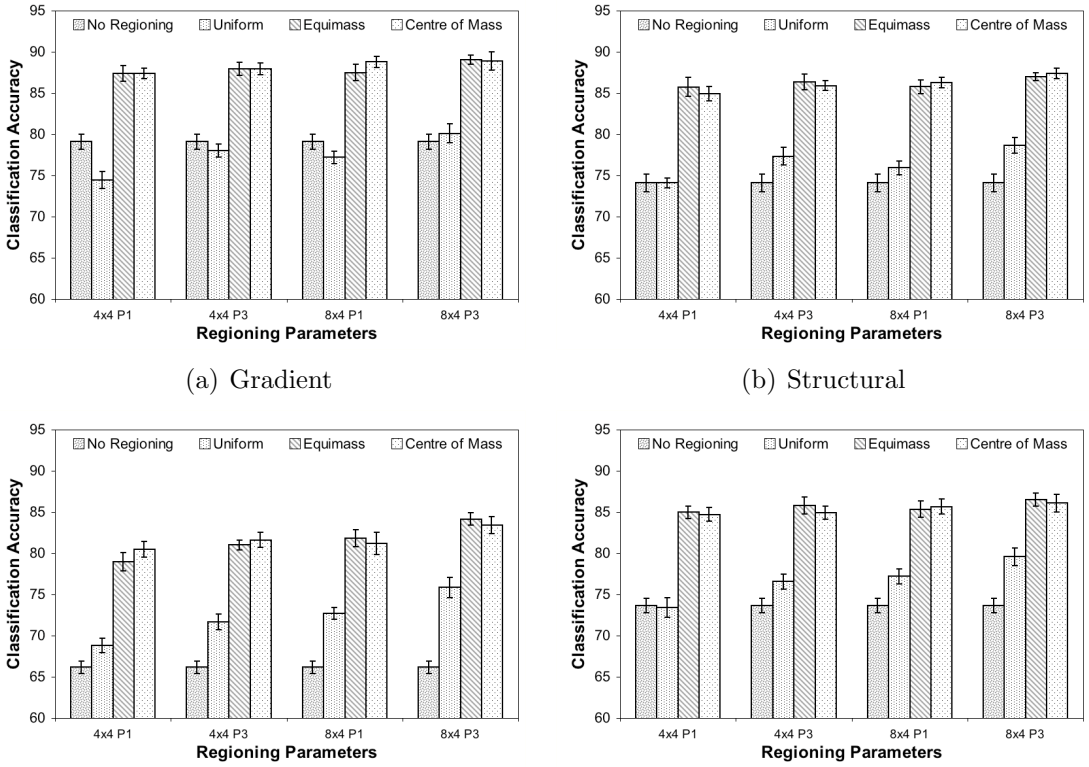
Table 13.3: CEDAR region sampling results

An alternative method of region sampling is to use concentric circles instead of a rectangular grid. The use of ring region sampling is described with greater detail in Sections 5.1.2 and 5.2.2. For this method of sampling, the most ideal features are rotation invariant LBPs, as this allows them to be unaffected by differently orientated signatures. Therefore, in regards to this, LBP uniform will be tested, as it performed significantly better than LBP invariant. The ability of ring region sampling combined with LBP uniform can be seen in Table 13.4 and Figure 13.4, where R is the number of regions and P is the number of levels in the spatial pyramid.

Ring Region	R8 P4	R4 P3	R8 P1	Average
Uniform	77.06 ± 1.04	**76.92** ± 1.20	75.15 ± 0.93	76.38
Equimass	**77.78** ± 0.70	74.92 ± 1.48	**76.76** ± 0.95	**76.49**

Table 13.4: Results for ring region sampling

These results show there is an insignificant difference between the uniform ring and equimass ring techniques. Varying the number of regions or pyramid levels



(a) Gradient

(b) Structural

(c) Stroke

(d) Concavity

Figure 13.3: CEDAR results for variations in regioning and spatial pyramid size in relation to four image features. The abbreviations 8×4 stands for 8 regions across 4 regions down and P3 stands for a pyramid with 3 levels. P1 indicates that spatial pyramids were not used

does impact the produced results, but in general, this variations is in the range of 1% to 2%. From these results, it can be ascertained that there is no benefit of using the ring regioning techniques over the rectangular grid based techniques.

13.3.2 GPDS Results

Testing of the GPDS dataset was carried out in the same fashion as the CEDAR. Based on the CEDAR results, uniform sampling will not be tested. The equimass and COM techniques will be contrasted against the achieved classification accuracy when no region sampling is used. The results from this testing can be seen in Table 13.5 or alternatively in Figure 13.5.

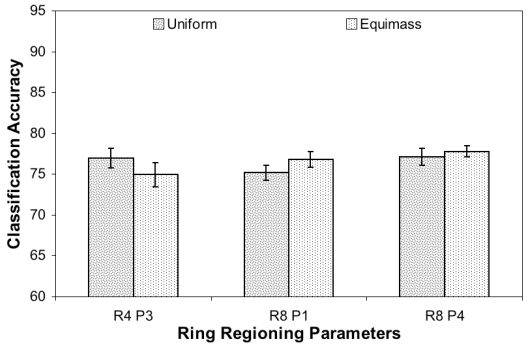
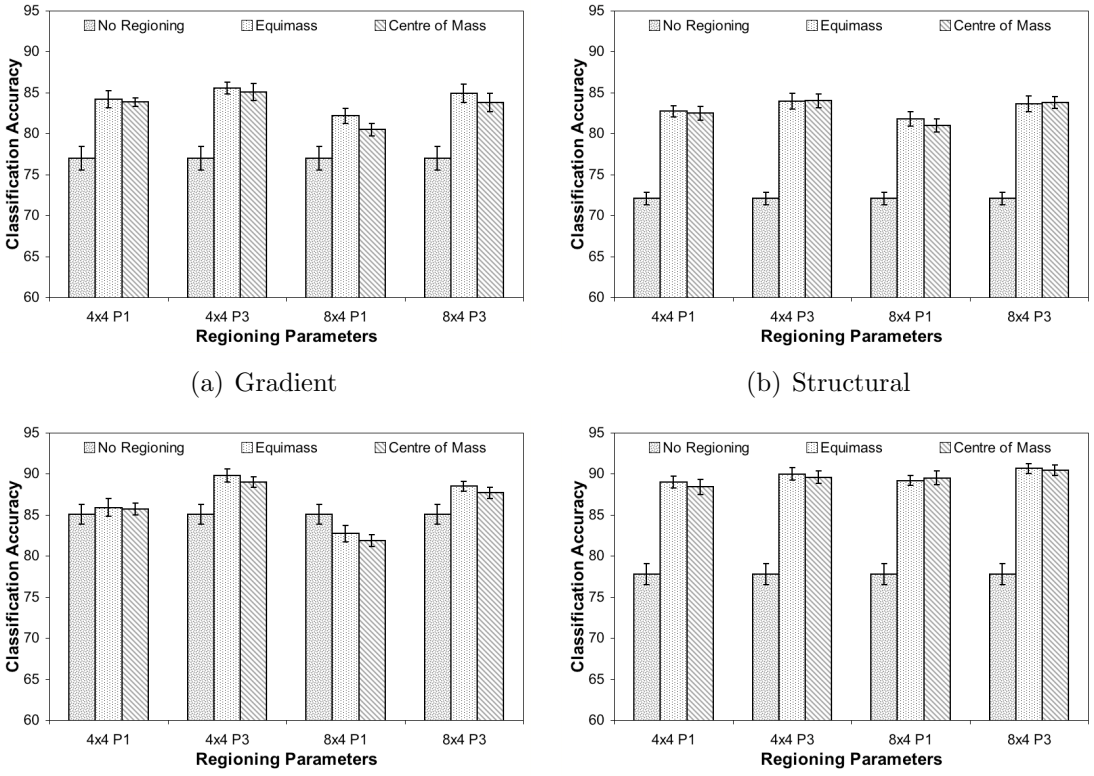


Figure 13.4: Graphed results for ring region sampling

Region	P	Size	Gradient	Structural	Strokes	Concavities	Average
N.A.	1	1×1	77.00 ± 1.43	72.08 ± 0.76	85.04 ± 1.19	77.75 ± 1.28	77.97
Equimass	1	4×4	84.19 ± 1.07	82.71 ± 0.68	85.90 ± 1.08	88.99 ± 0.71	85.45
		8×4	82.17 ± 0.91	81.79 ± 0.87	82.72 ± 0.98	89.16 ± 0.59	83.96
	3	4×4	**85.55** ± 0.73	83.94 ± 0.96	**89.78** ± 0.78	89.97 ± 0.75	**87.31**
		8×4	84.92 ± 1.11	83.63 ± 0.99	88.49 ± 0.62	**90.66** ± 0.61	86.93
COM	1	4×4	83.86 ± 0.52	82.50 ± 0.84	85.72 ± 0.75	88.39 ± 0.91	85.12
		8×4	80.46 ± 0.73	80.99 ± 0.79	81.87 ± 0.74	89.50 ± 0.82	83.21
	3	4×4	85.04 ± 1.03	**84.00** ± 0.82	88.99 ± 0.62	89.58 ± 0.77	86.90
		8×4	83.79 ± 1.09	83.78 ± 0.73	87.69 ± 0.70	90.43 ± 0.66	86.42

Table 13.5: GPDS region sampling results

These results show that using region sampling has a significant advantage when classifying signatures, with a substantial increase of approximately 10%. Following the same trend as CEDAR, equimass on average has a marginal advantage over COM, though this advantage is generally under 1%. The accuracy also changes in regards to the grid size, with 4×4 having a slight advantage over 8×4. The reason for this may be due to the GPDS dataset being smaller in size, meaning that the regions are too small for 8×4. The concavity feature contrasts with the other features as an 8×4 grid improves the results. Because concavity achieves the highest classification, the 8×4 grid will be utilised rather than the 4×4 grid, even though on average 4×4 is more effective. The secondary reason for this choice, is that it keeps the parameters between the two datasets equivalent.



(a) Gradient

(b) Structural

(c) Stroke

(d) Concavity

Figure 13.5: GPDS results for variations in regioning and spatial pyramid size in relation to four image features. The abbreviations 8×4 stands for 8 regions across 4 regions down and P3 stands for a pyramid with 3 levels. P1 indicates that spatial pyramids were not used

13.4 Spatial Pyramids

The previous section showed that region sampling increased the accuracy significantly. Spatial pyramids build upon region sampling by extracting the image features at different granularities, Section 6 describes this in greater detail. The experiments that are described here follow on from those in Section 5, with three level pyramids being contrasted against. In relation to this, it will be determined whether level weightings improve the classification.

13.4.1 CEDAR Results

The manner in which a spatial pyramid is formed is based on the initial grid size of the region sampling technique, with the number of horizontal and vertical regions

halving at each successive level. This means that a grid size of 4×4 will result in a different spacial pyramid than 8×4, as there will be a differing number of regions at each corresponding level. The effect that spatial pyramids has on the classification accuracy was shown previously in Table 13.3 and Figure 13.3 under Section 13.3.1. These results are presented again in Table 13.6 in a reduced form.

Region	P	Size	Feature Average
Equimass	1	4×4	84.26
		8×4	85.11
	3	4×4	85.28
		8×4	**86.67**
COM	1	4×4	84.38
		8×4	85.48
	3	4×4	85.10
		8×4	86.45

Table 13.6: CEDAR spatial pyramid results

The improvement that is gained from using spatial pyramids can be seen in these results, with an increase of approximately 1% being achieved in all cases. The use of an 8×4 grid is also shown to be more effective than 4×4 in regards to a three level pyramid. This is most likely the result of the signature being broken up into finer regions, though the effect of using 8×4 is that the global level has two regions instead of only one.

Each level in its current state has a weighting based on the number of features that can be extracted from it, where these weightings influence the final accuracy. The testing carried out here will determine whether two alternative schemes will be more effective. The first scheme is the standard weighting of a spatial pyramid (Lazebnik et al., 2006), which at its finest level will be assigned half of the overall weighting, while the remain two levels account for a quarter each. The second scheme tested gave each level an equal weighting. The results for both of these are shown in Table 13.7 and Figure 13.6.

Weighting	Gradient	Structural	Strokes	Concavities	Average
Unmodified	**89.05** ± 0.73	**87.29** ± 0.39	**84.28** ± 0.86	**86.51** ± 0.77	**86.78**
$(\frac{1}{2}, \frac{1}{4}, \frac{1}{4})$	88.62 ± 0.37	87.20 ± 0.75	81.67 ± 0.61	85.64 ± 0.65	85.78
$(\frac{1}{3}, \frac{1}{3}, \frac{1}{3})$	87.66 ± 0.75	86.01 ± 0.60	79.02 ± 0.71	84.79 ± 1.35	84.37

Table 13.7: CEDAR results for different weighting schemes

The results show that the unmodified scheme gives the best results, with at least a 1% gain over the other two schemes that were tried. It is possible that an alternative weighting may prove to be more advantageous, but based on these results, the unmodified scheme will be utilised.

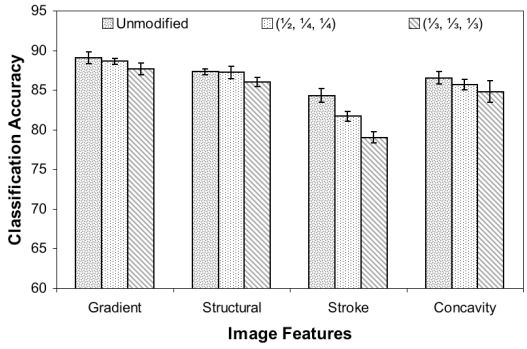


Figure 13.6: Graphed results for the CEDAR weighting schemes

13.4.2 GPDS Results

The testing of spatial pyramids in regards to GPDS was carried out in the same manner as CEDAR. The effect that they have on the accuracy was shown previously in Table 13.5 and Figure 13.5 under Section 13.3.2. Table 13.8 presents these results again in a reduced form.

Region	P	Size	Feature Average
Equimass	1	4×4	85.45
		8×4	83.96
	3	4×4	**87.31**
		8×4	86.93
COM	1	4×4	85.12
		8×4	83.21
	3	4×4	86.90
		8×4	86.42

Table 13.8: GPDS spatial pyramid results

Like CEDAR, the use of spatial pyramids with GPDS improved the achieved results by up to 3% in all cases, and dropped the difference between the 8×4 and 4×4 grids from 2% down to 0.5%. Due to the added benefit of spatial pyramids, they will be utilised for all following experiments.

The weighting schemes that were tested on CEDAR were also tested on GPDS, with three weighting schemes being tried. The produced results can be seen in either Table 13.9 or Figure 13.7.

Weighting	Gradient	Structural	Strokes	Concavities	Average
Unmodified	85.22 ± 0.85	**84.21** ± 0.74	88.18 ± 0.85	**90.55** ± 0.58	**87.04**
$(\frac{1}{2}, \frac{1}{4}, \frac{1}{4})$	**85.58** ± 0.57	83.44 ± 0.88	**89.66** ± 0.52	89.26 ± 0.78	86.99
$(\frac{1}{3}, \frac{1}{3}, \frac{1}{3})$	84.79 ± 0.88	82.65 ± 0.79	88.87 ± 0.94	88.59 ± 0.76	86.23

Table 13.9: GPDS results for different weighting schemes

The GPDS dataset differs from CEDAR as both the gradient and stroke features have an increase in accuracy when the $(\frac{1}{2}, \frac{1}{4}, \frac{1}{4})$ weighting scheme was used. On average, the unmodified scheme has a very slight advantage over the other two, though this advantage is less than 1%. Because the unmodified scheme produces the highest results, it will be used exclusively for the remaining experiments.

From these weighting experiments, as well as the other experiments that were conducted throughout this chapter, conclusions can be drawn to which combination of feature extraction techniques worked the best. For the features, the four with

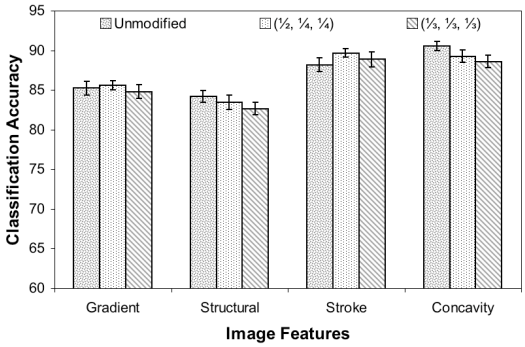


Figure 13.7: Graphed results for the GPDS weighting schemes

the highest results were chosen, with these being Gradient, Structural, Strokes and Concavities. The combination of equimass regioning and unweighted spatial pyramids using an 8×4 grid were found to be the best method for extracting features whilst providing spatial information and as such will be utilised for the remaining experiments.

Chapter 14

Evaluation: Classification

In the previous evaluation chapters, the ability of preprocessing and feature extraction techniques were established, with each consecutive experiment building on the previous. From the preprocessing experiments it was concluded that the combination of iterative binarisation and square shaped median filters were the best choice. The feature extraction experiments found that the most effective features were gradient, structural, strokes and concavities when used in combination with an 8×4 equimass grid and a three level spatial pyramid. This chapter will continue this process by determining the ability of the classifiers described in Chapters 9 and 10, including the impact that other variables have on the accuracy.

14.1 Experiment Design

Having carried out the experiments to find the most effective preprocessing and feature extraction techniques, the final step is to determine which classification methods allow for the best differentiation between the genuine and forgery signatures. This is determined by continuing the same methodology as the two previous evaluation chapters. Both the preprocessing and feature extraction techniques will remain fixed, while the classifiers are varied. The preprocessing and feature extraction techniques used are those that produced the best results previously. Each experiment will then be repeated 10 times for the purpose of determining how much the overall accuracy of the dataset fluctuates.

14.2 Comparison of Feature Thresholds

The conversion of a frequency vector to a binary feature vector requires each element to be thresholded to determine whether it will be a 0 or a 1. The are many ways of calculating this threshold, both manual and automatic. Chapter 8 describes a variety of these. This section will look at the ability of Adaptive Feature Thresholding (AFT) in relation to the other automatic thresholding techniques to determine its effectiveness. The thresholds come in two varieties, single threshold or an upper and lower threshold. The single threshold techniques consist of the mean and the mean minus the standard deviation (Mean-Stdev). The upper and lower thresholds will be tested with the standard deviation (Stdev), the sample standard deviation (Sample Stdev) and AFT, each of which are described in Section 8.2.

14.2.1 CEDAR Results

The impact that the five feature thresholding techniques have on CEDAR is expected to vary quite substantially, especially between the single and dual thresholds. The results that are produced can be seen in Table 14.1 as well as Figure 14.1.

Threshold	Gradient	Structural	Strokes	Concavities	Average
Mean	73.84 ± 0.66	76.64 ± 0.80	64.22 ± 1.05	76.83 ± 1.21	72.88
Mean-Stdev	83.32 ± 1.22	84.56 ± 0.92	68.58 ± 1.18	82.25 ± 0.79	79.68
Stdev	87.96 ± 0.72	86.30 ± 0.97	84.26 ± 0.62	85.55 ± 0.84	86.02
Sample Stdev	88.58 ± 1.06	86.09 ± 0.86	**84.62** ± 0.86	85.36 ± 0.98	86.16
AFT	**89.13** ± 0.52	**87.02** ± 0.86	84.48 ± 0.77	**86.22** ± 0.69	**86.71**

Table 14.1: CEDAR results for feature thresholding

Out of the five thresholds, the mean produced the lowest average, this is because the mean is found based on the training signatures and as such will tend to classify half of the genuine features incorrectly. By offsetting the threshold by the standard deviation improves this classification by almost 7%. When the upper threshold is also added via the standard deviation, there is another 7% increase to 86.02%. The use of the sample standard deviation only has a marginal increase of 0.14%, while

AFT has approximately a 0.5% improvement over this.

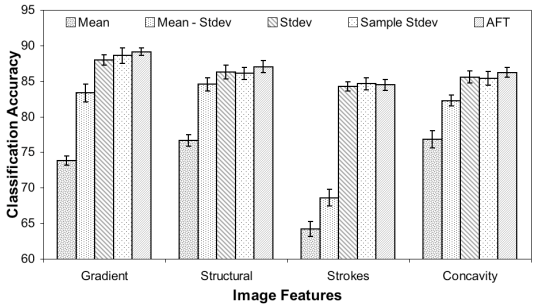


Figure 14.1: Graphed CEDAR results for feature thresholding

From these results, it is shown that AFT consistently achieves the highest accuracy for each of the features, except for strokes, where the sample standard deviation approach is higher by 0.14%. Due to the ability of AFT, it will be used for all remaining experiments.

14.2.2 GPDS Results

The GPDS dataset was also tested using these five feature thesholds, with the experiments being carried out in the same manner as CEDAR. The results that GPDS achieves can be seen in both Table 14.2 and Figure 14.2.

Threshold	Gradient	Structural	Strokes	Concavities	Average
Mean	66.34 ± 1.65	62.83 ± 1.15	89.04 ± 0.99	82.90 ± 0.99	75.28
Mean-Stdev	76.33 ± 0.52	74.07 ± 0.97	**91.36** ± 0.45	87.16 ± 1.02	82.23
Stdev	84.03 ± 1.01	84.40 ± 0.64	86.04 ± 0.76	90.78 ± 0.70	86.31
Sample Stdev	84.52 ± 0.61	**84.57** ± 0.81	86.38 ± 0.85	**91.19** ± 0.67	86.67
AFT	**84.88** ± 0.76	84.54 ± 0.81	88.57 ± 1.02	90.76 ± 0.62	**87.19**

Table 14.2: GPDS results for feature thresholding

From these results it can be seen that each threshold follows a similar trend as CEDAR, though there are some differences that occur. The first is that the stroke feature set achieves an incredible score of 91.36%, which surprisingly is with a single threshold. It is not clear why this score is so high, but in general the use of a single threshold makes the results susceptible to accepting forgeries as genuine.

AFT once again had the highest average classification, but the sample standard deviation did better for both the structural and concavity features.

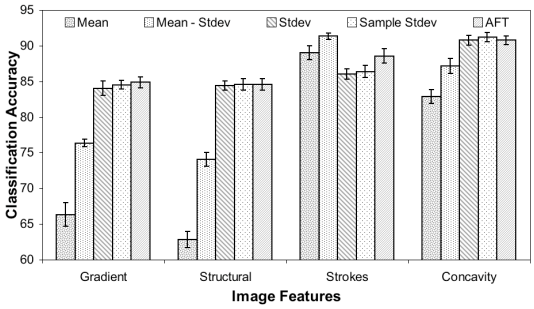


Figure 14.2: Graphed GPDS results for feature thresholding

Unlike CEDAR, AFT was not the highest for the majority of the features in the GPDS dataset, though like CEDAR, it did produced the best average accuracy. Because of this, AFT will be used for the remaining experiments.

14.3 Training with Subsets

Training a signature classifier begins with the random selection of signatures from the genuine set. The use of bagging attempts to improve the training by using randomly selected subsets of training signatures. This section will determine the effect that bagging has on the classification accuracy, where 100 randomly selected subsets are taken with the purpose of alleviating the impact that outliers have on the classification. Table 14.3 and Figure 14.3 show the results for the CEDAR dataset, where four subset sizes are tested and compared against the results without bagging.

Subset Size	Gradient	Structural	Strokes	Concavities	Average
N.A.	89.08 ± 0.52	86.92 ± 0.63	84.22 ± 0.74	86.14 ± 0.57	86.59
8	89.23 ± 0.57	87.03 ± 1.13	84.39 ± 0.95	86.34 ± 0.78	86.75
10	**89.26** ± 0.74	87.01 ± 0.95	**85.05** ± 0.92	**86.64** ± 0.83	**86.99**
12	89.13 ± 0.52	**87.21** ± 0.65	84.41 ± 0.56	86.41 ± 0.79	86.79
14	89.06 ± 0.58	87.06 ± 0.70	84.82 ± 0.79	86.22 ± 1.10	86.79

Table 14.3: CEDAR results when subsets are used

From these results it can be seen that the incorporation of training subsets improves the classification accuracy. This average accuracy peaks when the subset size is 10, though the structural feature had a 0.2% improvement over this when the subset size was 12. The difference between the average ability when no subsets are used and subsets of size 10 is 0.4%.

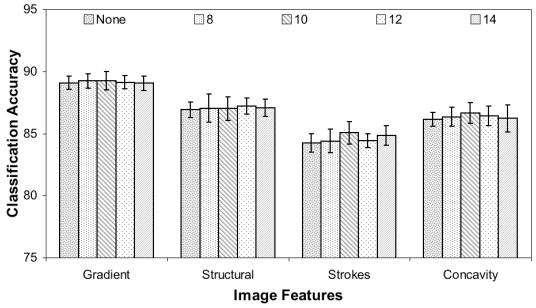


Figure 14.3: Graphed results from the CEDAR dataset when subsets are used

From the results that are shown, the use of subsets has a marginal increase in accuracy, though on average, this increase is less than 0.5% in all cases. Even though subsets increase the accuracy, they will not be utilised for the GPDS dataset or the remaining experiments, as to minimise the computational time of the overall approach.

14.4 Distance Measures

Once each signature has been converted into a binary feature vector, their similarity with one another can be easily determined. There are a variety of ways in which this similarity can be found, with four measures explored, GSC, Rogers-Tanamoto, Russell-Rao and Sokal-Michener, each of which are described in Section 9.3.

14.4.1 CEDAR Results

The ability of each similarity measure was tested using the experimental design described in Section 14.1. The results that were produced are shown in both Table 14.4 and Figure 14.4.

Measure	Gradient	Structural	Strokes	Concavities	Average
GSC	88.82 ± 0.59	**87.32** ± 0.72	84.34 ± 0.87	**86.41** ± 0.74	**86.72**
Rogers-Tanamoto	89.22 ± 0.68	87.02 ± 0.23	**84.69** ± 0.87	85.48 ± 0.67	86.60
Russell-Rao	88.20 ± 0.74	86.96 ± 0.93	84.14 ± 0.91	86.16 ± 0.82	86.37
Sokal-Michener	**89.53** ± 0.43	87.18 ± 0.85	84.23 ± 0.91	85.34 ± 0.67	86.57

Table 14.4: CEDAR results with different distance measures

These results show that the ability of each similarity measure has a tendency of fluctuating based on the feature that it is used with. The highest average classification accuracy is achieved using the GSC measure, though this average is only 0.12% above Rogers-Tanamoto. The lowest scoring measure was Russell-Rao, which produced an accuracy 0.35% below GSC.

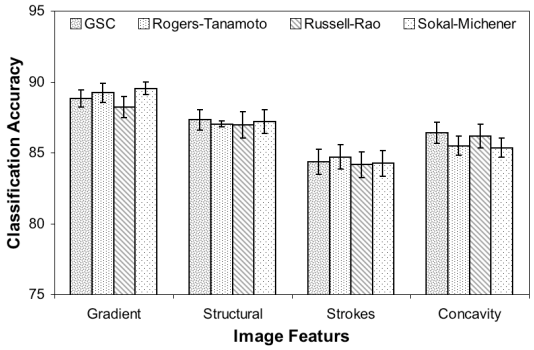


Figure 14.4: Graphed CEDAR results for each distance measure

The similarity measure that is used to differentiate between two binary feature vectors does influence the classification accuracy. From the experiments that were carried out on the CEDAR dataset, this influence on average appears to be minimal, with each feature being impacted differently. Based on these results, the GSC measure will be utilised for the CEDAR dataset.

14.4.2 GPDS Results

Testing of the four distance measures with the GPDS dataset was carried out in the same fashion as CEDAR. Table 14.5 and alternatively Figure 14.5 show the produced results.

Measure	Gradient	Structural	Strokes	Concavities	Average
GSC	84.94 ± 0.90	83.99 ± 1.11	**88.74** ± 0.95	90.62 ± 0.86	87.07
Rogers-Tanamoto	**85.31** ± 0.88	**84.71** ± 0.53	87.69 ± 0.60	**91.04** ± 0.60	**87.19**
Russell-Rao	84.73 ± 0.82	83.53 ± 0.43	88.30 ± 0.74	90.14 ± 0.65	86.68
Sokal-Michener	85.23 ± 0.77	84.57 ± 0.92	87.69 ± 0.82	90.60 ± 0.46	87.02

Table 14.5: GPDS results with different distance measures

The shown results differ between the GPDS and CEDAR datasets when the similarity measure is varied. The GSC measure drops below Rogers-Tanamoto by 0.12%, with the lowest measure once again being Russel-Rao at 86.68%, 0.51% below Rogers-Tanamoto. On average, the difference between each of these measures is minimal, even though the Rogers-Tanamoto measure achieved the highest accuracy for three of the features.

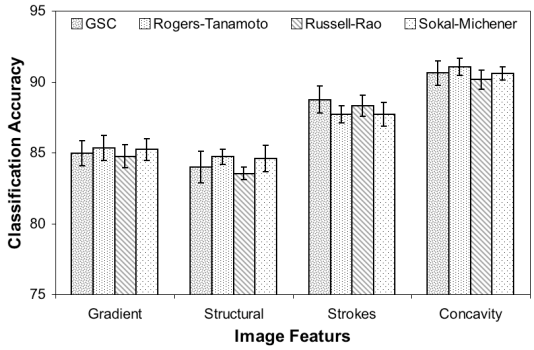


Figure 14.5: Graphed GPDS results for each distance measure

The GPDS results differ from CEDAR in that Rogers-Tanamoto has an increased tendency to produce a greater differentiation between the two signature classes. The choice of using GSC over Rogers-Tanamoto for the remaining experiments is so that the feature parameters between the two datasets remain the same, though in regard to this, the difference between GSC and Rogers-Tanamoto is only 0.12%.

14.5 Automatic Classification

In all previous experiments, each signature was classified using the manual offset described in Section 9.1.1, and is used to adjust the false acceptance rate and the false rejection rate ensuring that they both lie at the equal error rate. By doing this, the accuracy of a method can be accurately compared with that of another, and is a common approach in literature (Kalera et al., 2004; Chen and Srihari, 2006; Srihari et al., 2008). In practise though, the manual offset is not a viable option as normally there will be no negative training data available. Because of this, an automatic approach is required to offset the threshold from the mean. This automatic approach is described in Section 9.1.2 with its ability being determined in this section for both the CEDAR and GPDS datasets.

14.5.1 CEDAR Results

For automatic classification, two methods were explored, both of which change how the threshold is offset. Essentially they move the threshold down the similarity scale based on the sample standard deviation or by the lower sample standard deviation. The ability of these automatic methods can be seen in either Table 14.6 or Figure 14.6.

Offset	Gradient	Structural	Strokes	Concavities	Average
Manual	**89.01** ± 0.71	**87.23** ± 0.74	**84.05** ± 0.83	**86.32** ± 0.58	**86.65**
Lower Stdev	86.17 ± 1.01	83.99 ± 0.94	80.62 ± 0.49	82.30 ± 1.03	83.27
Stdev	87.30 ± 0.81	85.33 ± 0.64	80.89 ± 0.64	83.61 ± 0.52	84.28

Table 14.6: CEDAR results for automatic classification

In all cases, the manual method out performs both the sample standard deviation (Stdev) and the lower sample standard deviation (Lower Stdev) by over 2%. This decrease though was to be expected, as the false acceptance rate (FAR) and the false rejection rate (FRR) will not be the same. The difference between the two automatic offsets is about 1%, with the standard deviation being the better choice of the two. This is most likely due to the standard deviation moving the

threshold down the similarity scale by a smaller amount, decreasing the FAR.

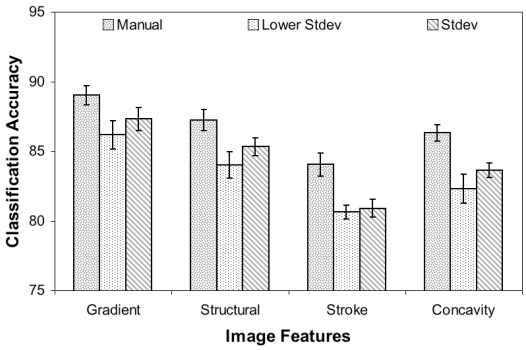


Figure 14.6: Graphed CEDAR results for automatic classification

Because the offset is found automatically, it is very unlikely to be at the equal error rate between the FAR and the FRR. In general, both of these automatic methods will have a higher FAR, with there being approximately a 15% to 20% difference between FAR and FRR. Having a higher FAR will classify more genuine signatures correctly, but will also allow a higher number of forgeries to be incorrectly classified.

14.5.2 GPDS Results

Following on from the CEDAR dataset, GPDS was tested in the same manner using automatic classification. The results that GPDS produced can be found in Table 14.7 or in Figure 14.7.

Offset	Gradient	Structural	Strokes	Concavities	Average
Manual	**84.95** ± 0.67	**84.61** ± 1.10	**88.58** ± 1.23	**90.82** ± 0.58	**87.24**
Lower Stdev	82.32 ± 0.97	81.00 ± 0.78	87.53 ± 1.11	87.34 ± 1.08	84.55
Stdev	82.81 ± 0.81	81.73 ± 0.80	87.37 ± 0.69	87.53 ± 0.75	84.86

Table 14.7: GPDS results for automatic classification

The GPDS results show that automatic classification follows the same trend as CEDAR, with the sample standard deviation having an average accuracy of 84.86%, 0.31% higher than the lower sample standard deviation. Like CEDAR, the manual method has an increase of about 2% over these methods.

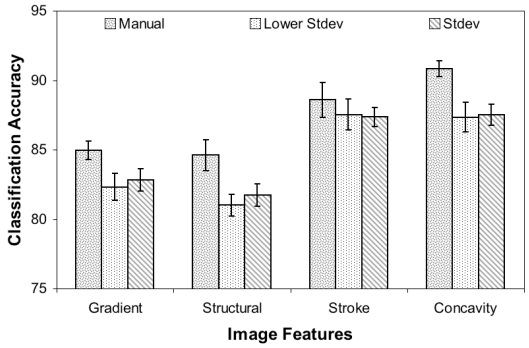


Figure 14.7: Graphed GPDS results for automatic classification

The decrease in performance between the automatic and manual methods is due to the difference in FAR and FRR. Like the CEDAR dataset, this difference is in the range of 15% to 20%, with the FAR being higher than the FRR. Even though there is this difference, the overall accuracy has not dropped by a significant amount.

According to Judd (2008), having the FAR higher than the FRR is beneficial in practise as *'ultimately any system could only identify a potential issue and the number of false:positives becomes an issue. Any possible fraud situation would still require a "human" to work through'.* This means that every suspect signature would require investigation, requiring time and money to perform.

14.6 Machine Learning

In all the previous evaluation sections only one class classification was tested. This section will look at two class classification in relation to three machine learning algorithms. These algorithms are naïve Bayes (NB), random forest (RF) and support vector machines (SVM). The implementation of these algorithms comes from the Waikato Environment for Knowledge Analysis (WEKA), which is a machine learning workbench written in the Java language (Witten and Frank, 2005). The two class experiments will use the final parameters decided upon from the previous tests, with the sole difference being in the classification step. The way in which the classification is carried out is outlined in Section 14.1.

14.6.1 CEDAR Results

The two class experiments that will be carried out on the CEDAR dataset will use the three machine learning algorithms described in Section 10.1. The training of these algorithms will use the method from Section 10.2, using three different combinations of training signatures, each of which are shown in Table 14.8.

Combination	Genuine	Forgery
1	12	4
2	12	12
3	16	16

Table 14.8: Combinations of genuine and forgery signatures used for training

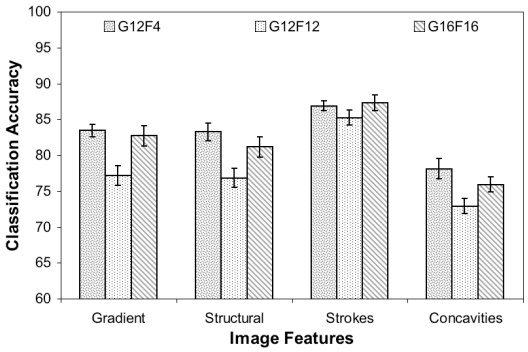
The first choice of using 12 genuine (G) and 4 forgery (F) signatures is that in practise the genuine signatures will be easier to obtain. The other two combinations are optimistic amounts designed to determine the effect of using an increased number of either genuine or forgery signatures. The results that are produced for the CEDAR dataset are shown in both Table 14.9 and Figure 14.8.

ML	G	F	Gradient	Structural	Strokes	Concavities	Average
NB	12	4	83.44 ± 0.89	83.26 ± 1.20	86.89 ± 0.69	78.12 ± 1.40	82.93
	12	12	77.17 ± 1.35	76.85 ± 1.28	85.23 ± 1.02	72.92 ± 1.08	78.04
	16	16	82.70 ± 1.45	81.18 ± 1.42	87.31 ± 1.12	75.93 ± 1.05	81.78
RF	12	4	85.08 ± 0.85	85.91 ± 0.57	85.56 ± 1.06	82.28 ± 1.03	84.71
	12	12	94.38 ± 0.57	93.87 ± 0.65	92.56 ± 1.04	94.05 ± 0.71	93.72
	16	16	94.91 ± 0.91	94.85 ± 0.59	**93.28** ± 0.90	**94.95** ± 0.77	94.50
SVM	12	4	90.10 ± 0.65	89.95 ± 0.68	86.60 ± 0.67	85.39 ± 0.58	88.01
	12	12	95.81 ± 0.38	95.41 ± 0.34	92.36 ± 0.51	91.47 ± 0.57	93.76
	16	16	**96.35** ± 0.52	**96.18** ± 0.59	93.10 ± 0.80	93.02 ± 0.56	**94.66**

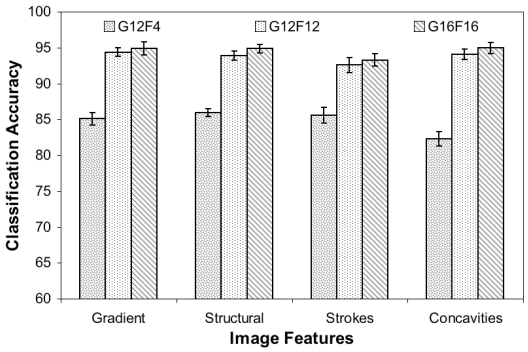
Table 14.9: CEDAR machine learning results

These results show that for the first combination there is a difference in ability between each of the three algorithms, with NB being lower than the other two at 82.93%, while SVM produced the highest accuracy at 88.01%. Surprisingly, when

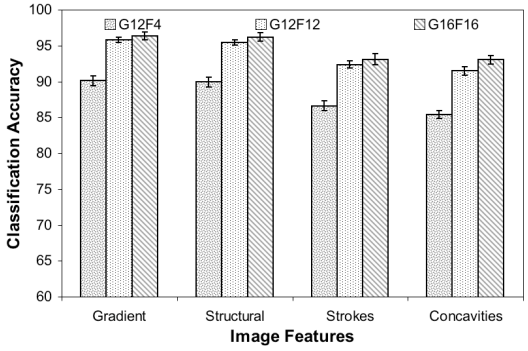
a large number of forgery signatures are used the NB accuracy drops. This is not the case for either RF or SVM which increase in accuracy to over 93% when 12 genuine and forgery signatures are used to train with. When 16 signatures are used for both classes, the average accuracy is over 94%.



(a) Naïve Bayes



(b) Random Forest



(c) SVM

Figure 14.8: Comparison of machine learning algorithms for the CEDAR dataset

The way each algorithm differentiates between the genuine and forgery signatures varies based on the number of training signatures. The result of this is that the FAR and FRR are not usually the same. The difference between the FAR and the FRR tends to vary based on three aspects, the machine learning algorithm, the feature and the number of training signatures that are used. When the number of training signature is higher the difference between the FAR and FRR for NB tends to increase, while for RF and SVM, it decreases. This difference is less than 1% for the gradient feature when SVM is used with 16 genuine and forgery signatures; this explains why the accuracy achieved over 96%.

14.6.2 GPDS Results

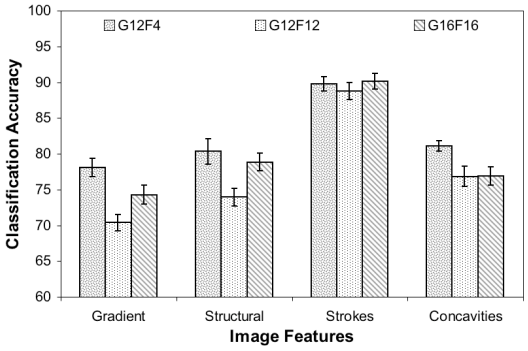
The manner in which the GPDS dataset was tested follows the same procedure as that of CEDAR, with the same combination of genuine and forgery signatures being used. The results that the two class approach achieves for the GPDS dataset are shown in Table 14.10 and Figure 14.9.

ML	G	F	Gradient	Structural	Strokes	Concavities	Average
NB	12	4	78.09 ± 1.31	80.34 ± 1.80	89.77 ± 0.99	81.11 ± 0.71	82.33
	12	12	70.37 ± 1.16	73.93 ± 1.25	88.79 ± 1.20	76.85 ± 1.45	77.49
	16	16	74.28 ± 1.32	78.84 ± 1.25	90.16 ± 1.10	76.89 ± 1.24	80.04
RF	12	4	74.13 ± 1.25	75.74 ± 1.43	87.88 ± 0.70	78.94 ± 1.40	79.17
	12	12	88.72 ± 0.46	89.91 ± 1.00	92.65 ± 0.86	93.34 ± 0.80	91.16
	16	16	90.01 ± 0.87	91.37 ± 1.16	**93.59** ± 0.85	**94.41** ± 0.56	92.35
SVM	12	4	82.30 ± 0.74	83.23 ± 1.09	88.51 ± 0.70	85.99 ± 0.66	85.01
	12	12	92.66 ± 0.38	92.48 ± 0.88	92.12 ± 0.71	91.82 ± 0.95	92.27
	16	16	**94.06** ± 0.93	**93.74** ± 0.95	93.51 ± 0.75	92.88 ± 0.61	**93.55**

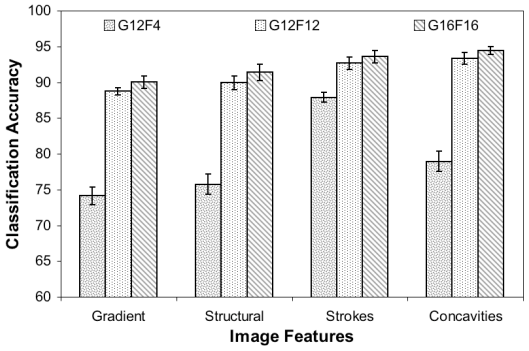
Table 14.10: GPDS machine learning results

These results show that the ability of each machine learning algorithm when using a particular combination of genuine and forgery signatures follows a trend that is much the same as CEDAR. The difference between each average accuracy has dropped, with the change from combination 2 to combination 3 (Table 14.8) only increasing the score by about 1%. The highest average was again produced by SVM when 16 genuine and forgery signatures were used.

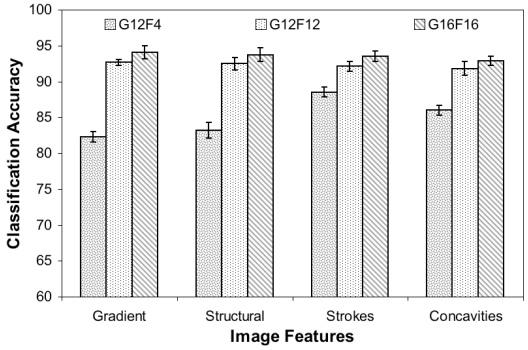
Like the CEDAR dataset, the difference between the FAR and the FRR for each GPDS experiment is similar. When a greater number of genuine and forgery signatures are used to train with this difference decreases. The highest result achieved for GPDS is once again SVM when it is trained with the gradient feature using 16 genuine and forgery signatures.



(a) Naïve Bayes



(b) Random Forest



(c) SVM

Figure 14.9: Comparison of machine learning algorithms for the GPDS dataset

Chapter 15

Evaluation: Summary

In the three previous chapters, the effectiveness of different techniques for processing, defining and classifying signatures were evaluated, and was carried out in regards to both the CEDAR and GPDS datasets. This chapter summarises these techniques with respect to the three main steps used to carry out the evaluation. Due to the large variety of techniques, there was a number of combinations that were not tested, as this would have required an infeasible number of experiments. To ensure that the most effective methods were found, each experiment determined which techniques achieved the greatest results. Following this, the combination of these most effective techniques will be compared definitively against other verification approaches published in literature.

15.1 Final Evaluation Configuration

The final configuration of techniques that were chosen for classifying signatures is described in this section with respect to the three main evaluation steps. These steps were preprocessing, feature extraction and classification.

The preprocessing of each signature began with its binarisation followed by its reconstruction. It was determined that the best combination for achieving this was iterative binarisation and a square shaped median filter. The use of rotation normalisation was found to detract from the classification accuracy.

Using the preprocessed signatures, a variety of features were extracted and

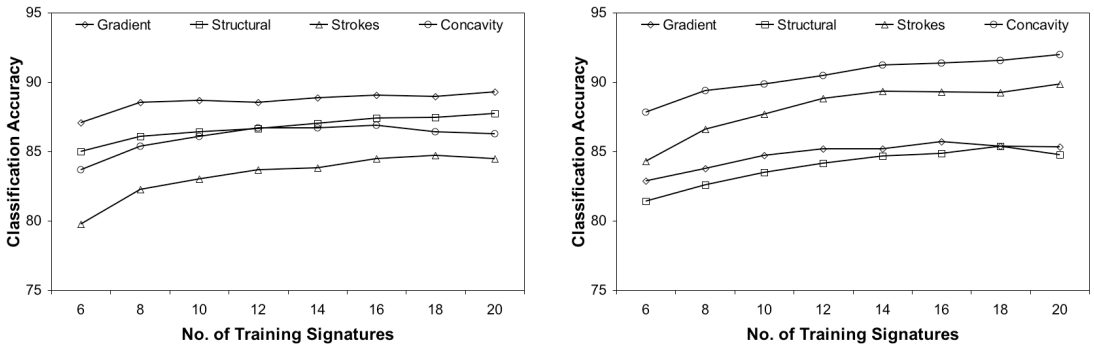
tested, where these feature are used to uniquely define each signature. The gradient, structural rules, long strokes and concavity features when tested were determined to achieve the highest accuracy. Additionally their combination with an 8×4 equimass grid and a three level spatial pyramid further boosted the accuracy.

Classification is the final step in the verification process, and is built from a range of individual components, where these components are feature thresholding, distance measures, and automatic one class and two class classification. For feature thresholding, it was found that adaptive feature thresholding was the best method for converting a feature frequency vector to a binary feature vector. The distance measure that produced the best distinction between pairs of binary feature vectors was the GSC measure. Using this combination, automatic classification was carried and is shown to produce the best classification when the standard deviation of the signature similarity scores is used to calculate the offset. The experiments conducted for two class classification found that support vector machines achieved the best result out of the three machine learning algorithms.

15.2 Count of Training Signatures

The experiments that were performed for both datasets in the previous evaluation chapters used a fix number of genuine training signatures. For CEDAR this was 16, while GPDS used 12. The reason behind this choice of training signatures was to ensure that the achieved accuracy is comparable to approaches previously published in literature, as they also use this number.

The stability of the produced accuracy when the number of training signatures is varied between six and twenty was investigated. The results of this variation can be seen in Figure 15.1(a) and 15.1(b) for CEDAR and GPDS respectively. They can also be found in a tabulated form in Tables 16.11 and 16.12 in Appendix A.



(a) CEDAR (b) GPDS

Figure 15.1: Classification accuracy when the number of training signatures are varied

The results show that there is an increase in accuracy when additional training signatures are used for both datasets, though this is less substantial with CEDAR. For CEDAR, the best feature was gradient, while for GPDS, this was concavity, both of which remained consistent in all cases. The stability of other approaches in literature are not often tested in this manner, as the usual procedure is to use a fixed number of training signatures. The combination of techniques that have been used here shows that the accuracy remains remarkably resilient even when there is only six training signatures.

15.3 Comparison of Verification Approaches

The verification approach that was implemented in this thesis is only one of many that have been tried in regards to off-line signature verification. This section will take the final one and two class classification approaches that were devised in the evaluation chapters and compare them against methods that have been proposed in previous literature. Each of these methods are described in greater detail in Chapter 2, with the comparison of each being carried out in regards to both the GPDS and CEDAR datasets.

15.3.1 One Class

The most effective one class approach determined in Section 15.1 will be evaluated against other approaches presented in literature. This method will use 16 training signatures for CEDAR and 12 for GPDS, as this appears to be standard in literature. For CEDAR the gradient feature will be used as it achieved the best classification, while the concavity feature will be used for GPDS. The automatic method for classifying signatures is not included here as the methods that are evaluated against utilise the manual approach to ensure that the results are at the equal error rate.

The highest accuracy that was achieved by Srinisvasan et al. (2005) was 82.44% using the Kolmogorov-Smirnov and Kullback-Leibler classifiers togethor, which is over 6% less than AFT Gradient at 88.68%. Table 15.1 shows the accuracy of three additional approaches that have been proposed. The prefix AFT indicates the results achieved by research conducted in this thesis. As well as this, the classification accuracy when the signatures are not normalised will be tested, as the size of the signature maybe an attribute of how a person writes. Finally the gradient, structural, stroke and concavity features will be combined to determine what improvement is achieved over single features.

Method	1-FAR (%)	1-FRR (%)	Accuracy (%)
GSC (Kalera et al., 2004)	80.50	77.55	78.50
Zernike Moments (Chen and Srihari, 2005)	83.70	83.40	83.60
Graph Matching (Chen and Srihari, 2006)	91.80	92.30	**92.10**
AFT Gradient	89.02	88.34	88.68
AFT GSC	88.58	89.14	88.86
AFT Gradient NN	90.69	90.66	90.67

Table 15.1: CEDAR Comparison Results, where NN stands for No Normalisation

The implemented one class approach used in this thesis is based heavily on the GSC procedure by Kalera et al. (2004) which achieved 78.50%, over 10% less than when the Adaptive Feature Thresholding (AFT) technique is included. The GSC ensemble only has a marginal increase over the gradient, while the inclusion

of the size attribute adds 2% to the accuracy. The graph matching approach is approximately 2% higher than AFT Gradient NN, though the literature in which the graph matching method is presented does not mention whether any normalisation or signature cropping was carried out. Therefore, if no cropping was applied to the AFT Gradient NN method the result is 92.16% and is equivalent to graph matching. If the centre of mass regioning is used instead of equimass, the accuracy further increases to 93.28%.

The same experiments that were conducted on the CEDAR dataset were also tested on GPDS, though concavity was used instead of gradient due to being more effective. The achieved results are shown in Table 15.2 where they are compared against the seven methods presented by Tian et al. (2007).

Method	1-FAR (%)	1-FRR (%)	Accuracy (%)
HMM	87.40	85.90	86.65
SVM Linear	81.46	78.94	80.20
SVM Poly	84.36	84.59	84.47
SVM RBF	86.88	84.59	85.73
Euclidean	84.34	83.79	84.06
Random	82.08	80.69	81.39
Optimal	88.11	86.74	87.43
AFT Concavity	90.68	90.60	90.64
AFT GSC	89.26	89.97	89.56
AFT Concavity NN	91.19	91.58	**91.39**

Table 15.2: GPDS comparison with Tian et al. (2007)

The ability of AFT in regards to the GPDS dataset is shown to have increased results over the other methods listed in Table 15.2. The AFT concavity method had approximately a 3% improvement over the optimal method presented in Tian et al. (2007). The GSC ensemble produces lower accuracy than when a single feature is used, and in this regard differs from the CEDAR dataset. Once again, if the size of each signature is taken into account the accuracy increases marginally by 0.75%.

From the comparisons shown previously, AFT has a substantial improvement

over the other methods presented for both the CEDAR and GPDS datasets. In relation to graph matching, there is no evidence that the signatures were either cropped or normalised. When this is applied to AFT the same accuracy is achieved. It was shown that if centre of mass regioning was used instead of equimass when no cropping or normalisation occurs, the result further increased. The accuracy of the GSC ensemble fluctuated in comparison with both the gradient and concavity features, but in practise would be the better choice as it should remain more stable.

15.3.2 Two Class

Two class classification trains with signatures from both the genuine and forgery classes, because of this, these approaches can not be fairly compared with the one class approach. This section will determine the effectiveness of the two class approaches implemented in this thesis and compare them against those proposed in literature. This will be carried out for both the CEDAR and GPDS datasets.

The classification accuracy that was achieved by Srihari et al. (2004) can be seen in Table 15.3 where two combinations of genuine (G) and forgery (F) signatures are used to train with. The methods identified with a '⋆' were produced from the research conducted in this thesis.

	Method	G	F	1-FAR (%)	1-FRR (%)	Accuracy (%)
	Distance Statistics	16	16	77.90	78.70	78.30
	Naïve Bayes	16	16	75.90	77.10	76.50
	Distance Statistics	16	5	79.30	82.40	80.80
	Naïve Bayes	16	5	87.00	90.05	88.55
	SVM	16	5	89.90	91.50	90.70
⋆	Naïve Bayes	16	16	97.95	50.61	74.28
⋆	SVM	16	16	94.05	94.07	**94.06**
⋆	Naïve Bayes	16	5	96.90	76.07	86.48
⋆	SVM	16	5	85.09	97.68	91.39

Table 15.3: CEDAR comparison with Srihari et al. (2004)

The results in Table 15.3 show that the approaches introduced by Srihari et al.

(2004) when combined with different classifiers produce a maximum accuracy of 90.70%. The naïve Bayes classifier that was tested in this thesis did not fair that well as it had a 2% decrease in accuracy from its counterpart by Srihari et al. (2004) for both combinations of training signatures. When the number of forgeries are decreased for Naïve Bayes, the accuracy is shown to increase, the reason for this has not been identified but is likely correlated with the high false rejection rate. In regards to both combinations of training signatures, the SVM classifier used in this thesis produced the best results.

Following the CEDAR dataset, the same experimental setup was tested with GPDS. The difference though is that the number of training signatures differs to allow each approach to be fairly compared against. Table 15.4 shows the results that are achieved when the resilient backpropagation (RBP) and radial basis function (RBF) neural networks are used. These two neural network classifiers are trained using all four features, described by Armand et al. (2006). SVM was used to compare against these two classifiers as it produced the highest result.

	Method	G	F	1-FAR (%)	1-FRR (%)	Accuracy (%)
	RBP	18	22	Not Provided	Not Provided	85.90
	RBF	18	22	Not Provided	Not Provided	91.12
⋆	SVM	18	22	95.77	92.69	**94.23**
⋆	SVM	9	11	93.23	88.65	90.94

Table 15.4: GPDS comparison with Armand et al. (2006)

The results that are shown in Table 15.4 indicate that both RBP and RBF are not as effective as SVM for classifying signatures. To further show the ability between these two neural networks and SVM, the number of training signatures was halved when training the SVM. The difference in accuracy between RBF and SVM in this case is only 0.18%, effectively meaning that the two classifier are equivalent when RBF has twice the number of training signatures.

The results that were shown for both the CEDAR and GPDS comparisons indicate that when the training method outlined in Section 10.1 is used in conjunction with the WEKA work bench, the produced results tend to be better than those

previously proposed.

Chapter 16

Conclusion

Signature verification is one of the most important and common tasks in today's society. Signatures are the most accepted form of identity verification, however forgers can often exploit signatures to impersonate an individual, giving them the authority to carry out tasks that they are not permitted to. Generally, verification to combat forgers is carried out by forensic document examiners. An alternative and cheaper approach is automatic signature verification, in which a computer determines whether the signature is a genuine or a forgery.

This thesis explored each of the three main steps required for automatic off-line signature verification, where these steps were preprocessing, feature extraction and classification. For each step, well-established techniques were tested and compared against novel techniques introduced in this thesis. The best achieved accuracy was then evaluated against published literature to determine the improvement.

16.1 Explored Techniques and Their Ability

The results show that our method out-performs the majority of approaches previously published. Signature preprocessing methods generally remains the same throughout literature, though the choice of using rotation normalisation varies. We show that there is a significant increase in accuracy when signatures are left at their original orientation. This indicates that the orientation of a signature is a potential attribute which is often overlooked.

Additionally, the use of normalisation to ensure that each signature is the same size is commonly applied, so that no bias is introduced. However, as with orientation, the size of a signature is also a potential attribute. Our results show that when the size is taken into account there is a distinct performance increase.

Feature extraction is normally carried out using a variety of features, as multiple features often boost the ability to differentiate between signatures. The introduction of Adaptive Feature Thresholding (AFT) when tested using the GSC ensemble achieve 88.86%, which is a substantial gain over the manual threshold at 78.50%. The best result of 93.28% was attained using AFT with centre of mass regioning.

The testing of newly published techniques often utilises manual classification approaches to ensure that the false acceptance and false rejection rates are equal. By doing this, the achieved results can be compared more accurately. The disadvantage though is that this method of classification is not feasible in practise, as there is no guarantee that the false acceptance and false rejection rates will remain equal. As well as this, each approach is usually trained with more signatures than what would be realistic in practise. This thesis introduced a fully automatic one class classification scheme that is shown to produce classification accuracies that differ from the manual accuracy by a minimal amount. The final classification approach chosen is shown to remain relatively stable when the number of training signatures is varied, with 6 training signature getting 87.08% and 20 attain 89.29% when the gradient feature is used.

16.2 Contributions

From the research that was carried out in this thesis, the techniques that have the greatest impact were identified. This extensive examination of each technique has not been carried out in previous work, and as such allowed insights into their ability. This thesis also introduced new techniques, some of which proved to be beneficial, whilst others unfortunately were not. The following is a list of contributions made to research, listed roughly in order of importance:

1. I introduced Adaptive Feature Thresholding, which vastly improves the conversion of a feature frequency vector to a binary feature vector by conforming more closely with the training signatures.

2. Demonstrated that the novel combination of region sampling and spatial pyramids boosts the classification ability.

3. Identified that adding additional features to uniquely distinguish a signature does not necessarily boost the classification accuracy, as good single features such as gradients and concavity features can be better than ensembles.

4. Illustrated that rotation normalisation decreases accuracy.

5. Demonstrated that simple median filters are superior to region growing.

6. Introduced an algorithm for a fully automatic approach to one class signature classification.

7. Improved the concavity feature by allowing it to take into account a greater variety of concavities, increasing the achieved results.

8. Demonstrated that size and rotation are potentially powerful, yet underutilised attributes.

9. Introduced the centre of mass grid and ring region sampling.

10. Adapted iterative thresholding into a local style binarisation method.

11. Introduced region rotation, which improves on using the axis of least inertia for rotation normalisation.

From these, the most significant contribution that this thesis made to the task of signature verification was the introduction of Adaptive Feature Thresholding. Essentially this approach ensures that the transformation of a feature frequency vector to a binary feature vector is properly carried out, using only a single positive class. This is a significant achievement.

Bibliography

Al-Shoshan, A. (2006). Handwritten Signature Verification Using Image Invariants and Dynamic Features. *Proceedings of the International Conference on Computer Graphics, Imaging and Visualisation*, pages 173–176.

Armand, S., Blumenstein, M., and Muthukkumarasamy, V. (2006). Off-line Signature Verification based on the Modified Direction Feature. *Proceedings of the 18th International Conference on Pattern Recognition (ICPR'06)-Volume 04*, pages 509–512.

Bernsen, J. (1986). Dynamic Thresholding of Grey-Level Images. In *Proceedings of the 8th International Conference on Pattern Recognition (ICPR)*, pages 1251–1255.

Blayvas, I., Bruckstein, A., and Kimmel, R. (2006). Efficient Computation of Adaptive Threshold Surfaces for Image Binarization. *Pattern Recognition*, 39(1):89–101.

Breiman, L. (1996). Bagging Predictors. *Machine Learning*, 24(2):123–140.

Chen, S. and Srihari, S. (2005). Use of Exterior Contours and Shape Features in Off-line Signature Verification. *8th International Conference on Document Analysis and Recognition*, pages 1280–1284.

Chen, S. and Srihari, S. (2006). A New Off-line Signature Verification Method Based on Graph Matching. *Pattern Recognition. ICPR 2006. 18th International Conference on*, 2:869–872.

Coetzer, J., Herbst, B., and du Preez, J. (2004). Offline Signature Verification Using the Discrete Radon Transform and a Hidden Markov Model. *EURASIP Journal on Applied Signal Processing*, 4:559–571.

Deng, P., Liao, H., Ho, C., and Tyan, H. (1999). Wavelet-Based Off-Line Signature Verification. *Computer Vision and Image Understanding*, 76(3):173–190.

Fang, B., Leung, C., Tang, Y., Tse, K., Kwok, P., and Wong, Y. (2003). Off-Line Signature Verification by the Tracking of Feature and Stroke Positions. *Pattern Recognition*, 36(1):91–101.

Favata, J. (2008). personal communication.

Favata, J. and Srikantan, G. (1996). A Multiple Feature/Resolution Approach to Handprinted Digit and Character Recognition. *International Journal of Imaging Systems and Technology*, 7(4):304–311.

Impedovo, S. and Pirlo, G. (2007). Verification of Handwritten Signatures: an Overview. *Image Analysis and Processing, 2007. ICIAP 2007. 14th International Conference on*, pages 191–196.

Jain, A., Griess, F., and Connell, S. (2002). On-line signature verification. *Pattern Recognition*, 35(12):2963–2972.

Jain, A., Ross, A., and Prabhakar, S. (2004). An introduction to biometric recognition. *Circuits and Systems for Video Technology, IEEE Transactions on*, 14(1):4–20.

Judd, J. (2008). personal communication.

Justino, E., 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*, 1034.

Justino, E., Bortolozzi, F., and Sabourin, R. (2005). A comparison of SVM and HMM classifiers in the off-line signature verification. *Pattern Recognition Letters*, 26(9):1377–1385.

Kalera, M., Zhang, B., and Srihari, S. (2004). Offline Signature Verification and Identification Using Distance Statistics. *International Journal of Pattern Recognition and Artificial Intelligence*, 18(7):1339–1360.

Larkins, R. and Mayo, M. (2008). Adaptive Feature Thresholding for Off-line Signature Verification. In *Image and Vision Computing New Zealand, 2008. IVCNZ 2008. 23rd International Conference*. IEEE Xplore.

Lazebnik, S., Schmid, C., and Ponce, J. (2006). Beyond Bags of Features: Spatial Pyramid Matching for Recognizing Natural Scene Categories. In *Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition-Volume 2*, pages 2169–2178. IEEE Computer Society Washington, DC, USA.

Lutterbeck, B., Ishii, K., and Gehring, R. (2000). Governing Legal Identities Lessons from the History of Seals and Signatures. *Information Security Solutions Europe (ISSE) Conference, Barcelona, Spain*, 28:2000–09.

Ma, Z., Zeng, X., Zhang, L., Li, M., and Zhou, C. (2007). A Novel Off-Line Signature Verification Based on Adaptive Multi-resolution Wavelet Zero-Crossing and One-Class-One-Network. *LECTURE NOTES IN COMPUTER SCIENCE*, 4493:1077.

Mäenpää, T., Ojala, T., Pietikäinen, M., and Soriano, M. (2000). Robust Texture Classification by Subsets of Local Binary Patterns. In *Proceedings of the 15th International Conference on Pattern Recognition - Volume 03*, pages 947–950.

Majhi, B., Reddy, Y., and Babu, D. (2006). Novel Features for Off-line Signature Verification. *International Journal of Computers, Communications & Control*, 1(1):17–24.

Martinez, L., Travieso, C., Alonso, J., and Ferrer, M. (2004). Parametrization of a forgery Handwritten Signature Verification using SVM. *IEEE 38th Annual 2004 International Carnahan Conference on Security Technology*, pages 193–196.

Mitchell, T. (1997). *Machine learning*. McGraw-Hill.

Mizukami, Y., Yoshimura, M., Miike, H., and Yoshimura, I. (2002). An Off-Line Signature Verification System Using an Extracted Displacement Function. *Pattern Recognition Letters*, 23(13):1569–1577.

Ojala, T., Pietikäinen, M., and Mäenpää, T. (2002). Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. *IEEE transactions on pattern analysis and machine intelligence*, 24(7):971–987.

Oliveira, L., Justino, E., Sabourin, R., and Bortolozzi, F. (2008). Combining Classifiers in the ROC-Space for Off-line Signature Verification. *Journal of Universal Computer Science*, 14(2):237–251.

Otsu, N. et al. (1979). A Threshold Selection Method from Gray-Level Histograms. *IEEE Transactions on Systems, Man, and Cybernetics*, 9(1):62–66.

Özgündüz, E., Şentürk, T., and Karslıgil, M. (2005). Off-Line Signature Verification and Recognition by Support Vector Machine. In *European Signal Processing Conference*.

Pietikäinen, M., Ojala, T., and Xu, Z. (2000). Rotation-invariant texture classification using feature distributions. *Pattern Recognition*, 33(1):43–52.

Platt, J. (1999). Sequential Minimal Optimization: A Fast Algorithm for Training Support Vector Machines. *Advances in Kernel Methods-Support Vector Learning*, pages 185–208.

Ridler, T. and Calvard, S. (1978). Picture Thresholding Using an Iterative Selection Method. *IEEE Transactions on Systems, Man and Cybernetics*, 8(8):630–632.

Sabourin, R. (1997). Off-Line Signature Verification: Recent Advances and Perspectives. In *Advances in Document Image Analysis: First Brazilian Symposium, BSDIA'97, Curitiba, Brazil, November 2-5, 1997: Proceedings*. Springer.

Shi, Z. and Govindaraju, V. (1996). Character Image Enhancement by Selective Region-Growing. *Pattern Recognition Letters*, 17(5):523–527.

Srihari, S., Srinivasan, H., Chen, S., and Beal, M. (2008). Machine Learning for Signature Verification. *Intelligence (SCI)*, 90:387–408.

Srihari, S., Xu, A., and Kalera, M. (2004). Learning Strategies and Classification Methods for Off-line Signature Verification. *Frontiers in Handwriting Recognition, 2004. IWFHR-9 2004. Ninth International Workshop on*, pages 161–166.

Srikantan, G., Lam, S., and Srihari, S. (1996). Gradient-based contour encoding for character recognition. *Pattern Recognition*, 29(7):1147–1160.

Srinisvasan, H., Beal, M., and Srihari, S. (2005). Machine Learning approaches for Person Identification and Verification. In *SPIE Conference on Homeland Security*, pages 574–586.

Tian, W., Qiao, Y., and Ma, Z. (2007). A New Scheme for Off-line Signature Verification Using DWT and Fuzzy Net. In *Proceedings of the Eighth ACIS International Conference on Software Engineering, Artificial Intelligence, Networking, and Parallel/Distributed Computing (SNPD 2007)-Volume 03*, pages 30–35. IEEE Computer Society Washington, DC, USA.

Topi, G., Tomislavmuc, Z., and Skala, K. (2005). Reimplementation of the Random Forest Algorithm. *Parallel Numerics 2005, Theory and Applications*.

Trier, O. and Jain, A. (1995). Goal-Directed Evaluation of Binarization Methods. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17:1191–1201.

White, P. (2004). *Crime Scene To Court: The Essentials Of Forensic Science*. Royal Society of Chemistry.

Witten, I. and Frank, E. (2005). *Data Mining: Practical Machine Learning Tools and Techniques*. Morgan Kaufmann Publishers Inc. San Francisco, CA, USA.

Zhang, B. (2006). Off-Line Signature Recognition and Verification by Kernel Principal Component Self-Regression. *Proceedings of the 5th International Conference on Machine Learning and Applications*, pages 28–33.

Zhang, B. and Srihari, S. (2003). Properties of Binary Vector Dissimilarity Measures. *Proc. of the JCIS CVPRIP*, 2003:26–30.

Zhang, J., Zeng, X., Lu, Y., Zhang, L., and Li, M. (2007). A Novel Off-line Signature Verification Based on One-class-one-network. In *Proceedings of the Third International Conference on Natural Computation (ICNC 2007)-Volume 02*, pages 590–594. IEEE Computer Society Washington, DC, USA.

Zimmerman, T., Russell, G., Heilper, A., Smith, B., Hu, J., Markman, D., Graham, J., and Drews, C. (2004). Retail applications of signature verification. *Proceedings of SPIE*, 5404:206–214.

Appendix A

Result Tables

The experiments performed in the evaluation chapters produced a large number of results, while the majority of these results are tabulated with the evaluations, some did not fit. The following tables are those that did not fit.

Method	Mass	Gradient	LBP	Concavity	Average
Fixed	**78.65** ± 1.07	87.70 ± 0.56	81.47 ± 0.81	85.49 ± 0.64	**83.33**
Iterative	75.95 ± 0.80	**87.82** ± 0.68	**82.65** ± 0.80	**85.50** ± 1.03	82.98
Otsu	72.02 ± 0.91	87.28 ± 1.28	79.95 ± 0.90	**85.50** ± 1.03	81.19
Local Iterative	72.22 ± 1.01	87.64 ± 0.87	79.44 ± 0.74	85.09 ± 0.55	81.10

Table 16.1: Results for the square shaped median filter

Method	Mass	Gradient	LBP	Concavity	Average
Fixed	**77.96** ± 0.87	**88.29** ± 0.89	79.54 ± 1.28	85.73 ± 0.90	**82.88**
Iterative	76.63 ± 0.85	88.14 ± 0.88	**80.61** ± 0.86	85.81 ± 0.70	82.80
Otsu	72.78 ± 1.49	87.69 ± 0.59	77.99 ± 0.63	85.56 ± 0.67	81.01
Local Iterative	76.57 ± 0.76	87.83 ± 1.05	80.24 ± 0.76	**86.25** ± 0.63	82.72

Table 16.2: Results for the plus shaped median filter

Method	Mass	Gradient	LBP	Concavity	Average
Fixed	75.46 ± 0.84	87.20 ± 0.54	83.07 ± 1.17	85.12 ± 0.95	82.71
Iterative	**75.62** ± 0.51	**87.58** ± 0.79	83.23 ± 0.21	85.34 ± 0.96	**82.94**
Otsu	72.72 ± 1.08	87.36 ± 0.72	**83.92** ± 0.81	85.73 ± 1.12	82.43
Local Iterative	70.73 ± 0.99	84.76 ± 0.84	79.58 ± 1.14	**86.03** ± 0.74	80.28

Table 16.3: Results for dilate and erode

Method	Mass	Gradient	LBP	Concavity	Average
Fixed	**74.55** ± 0.86	87.10 ± 0.68	76.79 ± 0.70	**85.40** ± 0.94	**80.96**
Iterative	74.06 ± 0.94	**87.54** ± 0.55	76.50 ± 0.44	85.16 ± 1.06	80.82
Otsu	72.34 ± 1.09	87.19 ± 0.48	**77.95** ± 0.85	85.33 ± 0.80	80.70
Local Iterative	70.18 ± 0.75	84.14 ± 1.13	71.94 ± 1.59	85.23 ± 0.87	77.87

Table 16.4: Results for region growing

Method	Mass	Gradient	LBP	Concavity	Average
Plus & DE	**73.10** ± 0.78	**87.55** ± 0.91	85.05 ± 1.14	84.86 ± 0.78	**82.64**
Square & DE	72.53 ± 0.97	87.24 ± 0.70	**85.20** ± 0.87	85.20 ± 0.87	82.54
Plus & RG	72.25 ± 1.02	**87.55** ± 0.72	79.84 ± 0.51	85.09 ± 0.56	81.18
Square & RG	71.79 ± 1.09	87.11 ± 1.04	79.79 ± 0.57	**85.31** ± 0.72	81.00

Table 16.5: Results for local iterative binarisation when the two median filters are combined with dilate and erode (DE) and region growing (RG)

Reconstruction	Threshold	Mass	Gradient	LBP	Concavity	Average
Square	Fixed	**78.65**	87.70	81.47	85.49	**83.33**
Square	Iterative	75.95	87.82	82.65	85.50	82.98
DE	Iterative	75.62	87.58	83.23	85.34	82.94
Plus	Fixed	77.96	88.29	79.54	85.73	82.88
Plus	Iterative	76.63	88.14	80.61	85.81	82.80
Plus	Local Iterative	76.57	87.83	80.24	**86.25**	82.72
DE	Fixed	75.46	87.20	83.07	85.12	82.71
Plus & DE	Local Iterative	73.10	87.55	85.05	84.86	82.64
Square & DE	Local Iterative	72.53	87.24	**85.20**	85.20	82.54
DE	Otsu	72.72	87.36	83.92	85.73	82.43
Square	Otsu	72.02	87.28	79.95	85.50	81.19
Plus & RG	Local Iterative	72.25	87.55	79.84	85.09	81.18
Square	Local Iterative	72.22	87.64	79.44	85.09	81.10
Plus	Otsu	72.78	87.69	77.99	85.56	81.01
Square & RG	Local Iterative	71.79	87.11	79.79	85.31	81.00
RG	Fixed	74.55	87.10	76.79	85.40	80.96
RG	Iterative	74.06	87.54	76.50	85.16	80.81
N.A.	Fixed	75.28	**88.34**	73.96	85.48	80.77
RG	Otsu	72.34	87.19	77.95	85.33	80.70
N.A.	Otsu	72.61	88.12	75.98	85.86	80.64
DE	Local Iterative	70.73	84.76	79.58	86.03	80.27
N.A.	Iterative	72.28	87.57	75.80	85.39	80.26
N.A.	Local Iterative	71.33	84.79	75.31	85.91	79.33
RG	Local Iterative	70.18	84.14	71.94	85.23	77.87

Table 16.6: Comparison of reconstruction and binarisation methods, ordered by the Average of the four image features

Method	Mass	Gradient	LBP	Concavity	Average
Original	**75.95** $\pm$ 0.80	**87.82** $\pm$ 0.68	**82.65** $\pm$ 0.80	**85.50** $\pm$ 1.03	**82.98**
Least Inertia	74.69 $\pm$ 1.05	84.17 $\pm$ 0.76	78.39 $\pm$ 0.95	84.05 $\pm$ 1.35	80.33
Region Rotation	75.25 $\pm$ 0.88	85.02 $\pm$ 0.96	80.98 $\pm$ 0.49	85.19 $\pm$ 0.75	81.61

Table 16.7: Results for rotation normalisation using iterative with square shaped median filter

Method	Mass	Gradient	LBP	Concavity	Average
Original	**76.63** ± 0.85	**88.14** ± 0.88	**80.61** ± 0.86	**85.81** ± 0.70	**82.80**
Least Inertia	74.51 ± 0.63	84.70 ± 0.62	76.87 ± 1.02	84.78 ± 0.71	80.22
Region Rotation	75.23 ± 0.68	85.77 ± 0.76	79.08 ± 0.68	85.62 ± 0.85	81.43

Table 16.8: Results for rotation normalisation using iterative with plus shaped median filter

Method	Mass	Gradient	LBP	Concavity	Average
Original	**75.62** ± 0.51	**87.58** ± 0.79	**83.23** ± 0.21	**85.34** ± 0.96	**82.94**
Region Rotation	73.70 ± 1.17	84.47 ± 0.74	80.62 ± 1.40	83.83 ± 0.74	80.66
Least Inertia	70.55 ± 1.15	82.64 ± 0.86	76.42 ± 0.91	83.05 ± 0.77	78.17

Table 16.9: Results for rotation normalisation using iterative with dilate and erode

Method	Mass	Gradient	LBP	Concavity	Average
Original	**76.57** ± 0.76	**87.83** ± 1.05	**80.24** ± 0.76	**86.25** ± 0.63	**82.72**
Least Inertia	67.65 ± 1.21	82.80 ± 0.89	73.93 ± 0.80	82.11 ± 1.11	76.62
Region Rotation	70.22 ± 0.63	84.40 ± 0.89	76.52 ± 0.83	83.96 ± 0.48	78.78

Table 16.10: Results for rotation normalisation using iterative with square shaped median filter

Training No.	Gradient	Structural	Strokes	Concavities	Average
6	87.08 ± 0.97	85.00 ± 1.06	79.76 ± 0.82	83.67 ± 0.67	83.88
8	88.56 ± 0.80	86.08 ± 0.60	82.27 ± 0.96	85.37 ± 0.71	85.57
10	88.70 ± 0.72	86.44 ± 0.55	83.02 ± 0.99	86.08 ± 0.50	86.06
12	88.52 ± 0.70	86.66 ± 0.75	83.70 ± 0.46	86.69 ± 0.54	86.39
14	88.89 ± 0.74	87.02 ± 1.00	83.83 ± 0.81	86.72 ± 0.67	86.62
16	89.05 ± 1.01	87.42 ± 0.56	84.46 ± 0.85	**86.89** ± 0.86	**86.96**
18	88.95 ± 0.80	87.44 ± 0.63	**84.73** ± 1.07	86.43 ± 0.88	86.89
20	**89.29** ± 0.91	**87.75** ± 1.23	84.48 ± 1.22	86.28 ± 1.06	86.95

Table 16.11: CEDAR results when the training count is varied

Training No.	Gradient	Structural	Strokes	Concavity	Average
6	82.85 ± 1.28	81.41 ± 1.13	84.30 ± 1.37	87.81 ± 0.71	84.09
8	83.77 ± 1.05	82.59 ± 1.21	86.60 ± 1.15	89.40 ± 0.66	85.59
10	84.71 ± 0.67	83.50 ± 1.02	87.67 ± 0.76	89.86 ± 0.66	86.44
12	85.19 ± 0.88	84.13 ± 0.65	88.81 ± 0.76	90.47 ± 0.85	87.15
14	85.19 ± 1.01	84.66 ± 1.03	89.35 ± 0.74	91.21 ± 0.82	87.60
16	**85.69** ± 1.00	84.88 ± 1.21	89.30 ± 0.89	91.38 ± 0.85	87.81
18	85.36 ± 1.52	**85.40** ± 0.91	89.24 ± 1.13	91.54 ± 0.82	87.89
20	85.35 ± 1.05	84.77 ± 1.44	**89.88** ± 0.96	**92.00** ± 0.64	**88.00**

Table 16.12: GPDS results when the training count is varied

Appendix B

IVCNZ Conference Paper

The following paper, *Adaptive Feature Thresholding for Off-line Signature Verification* (Larkins and Mayo, 2008), is based on the research conducted in this thesis, and was presented at the Image and Vision Computing New Zealand (IVCNZ) conference in November 2008.

Adaptive Feature Thresholding for Off-line Signature Verification

Robert Larkins, Michael Mayo

Machine Learning Group, Department of Computer Science, University of Waikato, New Zealand.
Email: {rll6,mmayo}@cs.waikato.ac.nz

Abstract

This paper introduces Adaptive Feature Thresholding (AFT) which is a novel method of person-dependent off-line signature verification. AFT enhances how a simple image feature of a signature is converted to a binary feature vector by significantly improving its representation in relation to the training signatures. The similarity between signatures is then easily computed from their corresponding binary feature vectors. AFT was tested on the CEDAR and GPDS benchmark datasets, with classification using either a manual or an automatic variant. On the CEDAR dataset we achieved a classification accuracy of 92% for manual and 90% for automatic, while on the GPDS dataset we achieved over 87% and 85% respectively. For both datasets AFT is less complex and requires fewer images features than the existing state of the art methods, while achieving competitive results.

Keywords: off-line signature verification, person-dependent, feature thresholding, spatial pyramid

1 Introduction

Hand written signatures have a well-established and accepted place in society as a formal means of personal verification, for both the identification and the intent of the signatory. Because of this, signatures are the most accepted method of verification [1] and are used in government, legal and commercial transactions. A result of this is that signatures are often forged for the purpose of feigning the authenticity of a document. This leads to the problem of being able to correctly verify whether a signature is a genuine or a forgery.

Many different approaches have been employed for accurately classifying whether a signature is a genuine or a forgery. These approaches are split into two categories: on-line and off-line.

Online approaches use a digitising surface to capture dynamic features about how a signature is written. These are features such as pressure, speed and direction, which allow online classification to achieve accuracies of over 95% [2].

Off-line verification deals with signatures that have been written on paper and scanned in to the computer. Because of this, they are unable to use dynamic features. This means that the signatures can only be distinguished from each other by what is visually available.

978-1-4244-2582-2/08/$25.00

Person-dependent classification is a commonly used method that is designed to train with genuine signatures from only one person. Two methods that have achieved high person-dependent classification accuracies are graph matching [3] and the Discrete Wavelet Transform (DWT) [4].

The approach that we use is similar to the Gradient, Structural and Concavity (GSC) method [5], except we only use the gradient direction. The improvement that dramatically increased classification accuracy was the way in which we implemented the thresholding of each image feature count. This enhanced the creation of the binary feature vector [6] by adaptively restricting which feature bits are set to 1. In GSC, thresholding is carried out with a fixed value that is manually chosen, with its capability being determined by experimentation, where if the count of an image feature is above this value, then the feature bit is set to 1, otherwise it is set to 0. An adaptation is then made to our thresholding method to achieve automatic classification. As well as these methods, a novel combination of spatial pyramids [7] and equimass sampling grids [8] is also introduced to help boost the classification accuracy.

2 Signature Representation

Each signature, before it is processed, is in the form of a binarised digital image (see Figure 1). This format does not describe the individual aspects of

the signature in a manner that makes it feasible for comparing it to other signatures. This is because digital images are designed to be visually identifiable to humans. As a result, each signature needs to be converted into a format that will allow the similarity of it and another signature to be easily computed.

This section details the implementation of the novel thresholding method, AFT, that this paper presents. Essentially, this method ensures that the comparative similarity of a signature is more accurately represented in contrast to the training signatures when it is converted from a digital image to a binary feature vector.

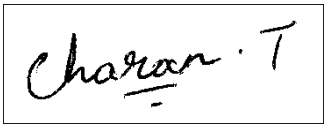


Figure 1: A signature that has been binarised.

2.1 Binary Feature Vector

A binary feature vector is a method of representing a signature by indicating whether a particular feature matches a certain criteria by turning the corresponding feature bit on. The vector structure is shown in equation (1), where V is the feature vector, z is a feature bit and k is the number of elements in the vector.

$$V = (z_1,\ z_2,\ \cdots,\ z_k) \qquad z_i \in \{0, 1\} \qquad (1)$$

2.2 Gradient Direction Extraction

The creation of a binary feature vector is based solely on the gradient direction of each pixel from across a signature. This direction θ of a pixel at coordinates x and y is found by equation (2), where G_x is the Sobel kernel for horizontal change and G_y is the kernel for vertical change.

$$\theta(x, y) = \tan^{-1}\left(\frac{G_y}{G_x}\right) \qquad (2)$$

The resulting direction is a value that ranges from 0.0 to 2π radians. This range can then be split into 18 non-overlapping segments based on $\frac{2\pi}{18}$ radians, allowing a gradient direction histogram to be created from the count of each direction. For the experiments conducted in this paper, 18 segments were chosen because this value proved effective in initial tests and is also used in [9].

2.3 Equimass Spatial Pyramids

Using the gradient direction in its current state only expresses a signature at the global level. This can be improved upon through the use of a spatial pyramid, which defines the signature at increasingly finer levels of granularity, improving the ability to distinguish a signature in relation to other signatures. This is normally achieved through the use of different types of image features which capture particular properties of the signature, but the use of spatial pyramids helps to overcome this need for multiple features.

The levels of granularity are produced by splitting a signature up into increasingly smaller regions, which is usually carried out with a uniform grid. The disadvantage of a uniform grid is that it does not capture the same structural properties of corresponding regions between signatures. A novel and effective approach that improved the capture of these structural properties was achieved by combining spatial pyramids and equimass sampling grids.

Equimass is an adaptive grid based on the number of black pixels or mass M of a signature, where the grid lines are found at the equimass divisions of the horizontal and vertical mass histogram. That is, where the masses between all adjacent points on either the x-axis or the y-axis are equal. This average mass M_A is found by equation (3), where r is the number of horizontal or vertical regions.

$$M_A = \frac{M}{r} \qquad (3)$$

Figure 2 shows how the grid lines will be placed for each of the three levels of the spatial pyramid if the number of regions for the finest level (c) is 8×4. An example of this calculation is if Figure 2(a) has mass $M = 2709$, then the single vertical line will be placed where the number of black pixels in both regions is $\frac{2709}{2} = 1354$.

(a)

(b)

(c)

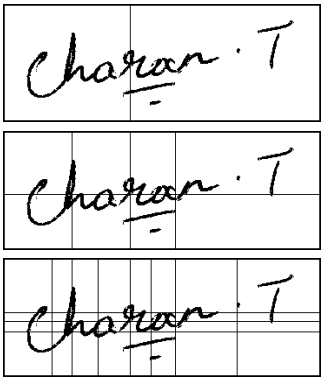


Figure 2: A three level spatial pyramid.

2.4 Novel Feature Thresholding

When thresholding a particular gradient direction of a region, the criteria for determining whether the corresponding feature bit will be 0 or 1 is dependent upon the training signatures, where if the direction count c of a set of pixels abides by $\tau_1 \leq c < \tau_2$,

then the feature bit is set to 1, otherwise it is set to 0.

The lower threshold, τ_1 and the upper threshold, τ_2 are found at one sample standard deviation S either side of μ, where μ is the mean of D, in which D is the count of a particular direction from the same region across each training signature. The problem with using S to calculate both τ_1 and τ_2 is that any possible skew is not taken into account, as the values from D may be spread about μ in a fashion that is not normally distributed. So to adjust for the skew, τ_1 is calculated by the equation (4) and τ_2 is calculated by (5).

$$\tau_1 = \mu - S_L \quad (4)$$

$$\tau_2 = \mu + S_U \quad (5)$$

The variable S_L is the average distance that values below μ lie from μ and is found by equation (6). S_U is found in a similar fashion but for values above μ, this is calculated by equation (7).

$$S_L = \sqrt{\frac{1}{D_L - 1}\sum_{i=1}^{n}(D_i - \mu)^2} \quad \forall D < \mu \quad (6)$$

$$S_U = \sqrt{\frac{1}{D_U - 1}\sum_{i=1}^{n}(D_i - \mu)^2} \quad \forall D > \mu \quad (7)$$

D_L is the number of values in D which are less than μ, D_U is the number that have a value greater than μ and n is the size of D.

For example, if $D = \{36, 47, 54, 59, 63, 81\}$ then μ would equal the mean value of D which is 56.67. S_L then equals 16.24 and is calculated by equation (6) using the values 36, 47 and 54, the values from D which are less than μ. S_U equals 17.86 and is found by equation (7), using the values 59, 63 and 81, the values greater than μ. τ_1 then equals 56.67 - 16.24 = 40.43 and τ_2 would equal 56.67 + 17.86 = 74.53. Therefore the corresponding feature bit will be 1 if $40.43 \leq c < 74.53$, otherwise it will be 0.

Using this method, each direction in a gradient direction histogram can be thresholded independently and used to produce one bit of the feature vector. This is then repeated for the histogram of each region across each spatial level for all signatures. Using this thresholding method on the signature in Figure 2 would produce a feature vector that is 756 feature bits in length, consisting of 36 bits for (a), 144 bits for (b) and 576 bits for (c).

3 Signature Classification

The classification of an unknown signature is based heavily on the similarity score of two feature vectors. The comparison of these two vectors produces

$V_a = 0\,1\,0\,1\,1\,0\,1\,0\,0\,0\,1\,0\,1\,1\,1\,0\,0\,1$
$V_b = 1\,1\,0\,0\,1\,1\,0\,1\,0\,1\,1\,0\,1\,0\,0\,1\,0\,1$
$V_c = 0\,0\,1\,0\,0\,1\,0\,1\,1\,1\,1\,0\,1\,0\,1\,0\,0\,1$
$V_u = 0\,1\,0\,0\,1\,1\,0\,1\,1\,0\,1\,0\,1\,0\,0\,1\,1\,1$

Figure 3: A set of binary feature vectors of length 18

four values based on the sum of the four possible variations at each position in the vectors. These four values are defined by equation (8), where C is the count of times that each of the four outcomes occur, i is the feature bit value for the first vector V_1, and j is the value for the second vector V_2.

$$C_{ij}\ (i, j \in \{0, 1\}) \quad (8)$$

The similarity between two vectors, that both have the length k, is calculated by the function f and produces a score between 0 and 1, where if the score is 0, the two vectors are completely different, while 1 means they are the same. This scoring method is used in [3] and is calculated by equation (9). The calculation of f only requires C_{00} and C_{11}, as it is designed to take into consideration only the positions where the feature bits are both the same. For example, if the feature vectors V_a and V_b in Figure 3 were compared, the result would be that $C_{00} = 4$ and $C_{11} = 5$. The similarity of these two vectors would be calculated by equation (9) and would equal 0.389.

$$f(V_1, V_2) = \frac{0.5 \times C_{00} + C_{11}}{k} \quad (9)$$

The similarity score between an unknown vector U and the set of training vectors T is then found as the mean similarity when U is compared to each signature in T. For example, if the vectors V_a, V_b and V_c from Figure 3 are used as the set T, and V_u is an unknown vector, the similarity score between T and V_u would be .491.

The threshold that will be used to determine the classification is produced from the average similarity score μ of the training vectors when they are compared to each other. This method of finding μ is outlined in [3], and [10]. Carrying on from the previous example, if T comprises of the first three vectors from Figure 3, μ would be .398.

Because μ is the mean similarity score of the training vectors, it will tend towards classifying half of the genuine signatures as forgeries, therefore its use as the threshold is not ideal. So to create the threshold, an offset is required to move the mean down the similarity scale, with the intent being to maximise the classification accuracy of both the genuine and forgery signatures. This is shown in Figure 4, where the threshold is the offset of the

mean. Two methods were investigated for finding this offset, these were manual and automatic, both of which are described below.

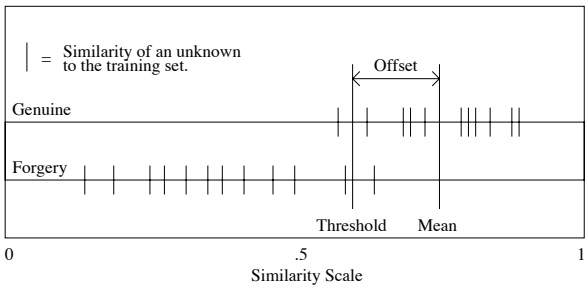


Figure 4: Threshold creation.

3.1 Manual Classification

Manual classification uses a fixed offset o to move μ down the similarity scale. This offset does not compute a threshold directly, but instead is used to generate a false acceptance rate (FAR) versus false rejection rate (FRR) curve, from which the best threshold (found at the equal error rate) is computed. This offset is then used for all signature sets, with the class of an unknown signature being determined by equation (10).

$$class = \begin{cases} genuine & \text{if } score \geq \mu - o \\ forgery & \text{if } score < \mu - o \end{cases} \quad (10)$$

The disadvantage of the manual method is that it requires a range of experiments to find a value which minimises the FAR and the FRR. The use of this method was to make AFT comparable to the graph matching technique [3], which also uses this manual offset.

3.2 Automatic Classification

Automatic classification is a heuristic method for finding an offset based on the training signatures. The offset is found in the same manner that AFT uses for calculating the lower sample standard deviation S_L, where only the values below μ are used. These values are the similarity scores from equation (9) when all training signatures are compared to each other. The class of an unknown signature is then determined by equation (11).

$$class = \begin{cases} genuine & \text{if } score \geq \mu - S_L \\ forgery & \text{if } score < \mu - S_L \end{cases} \quad (11)$$

4 Experiments

The evaluation of AFT was carried out on two datasets of signatures, both of which comprise of skilled forgeries. A skilled forgery is one in which the forger has both seen and practised writing a genuine signature, making it visually similar to the original. Because of this visual similarity, there is a substantial increase in difficulty of being able to classifying an unknown signature, as opposed to signatures of the random or simple variations [11].

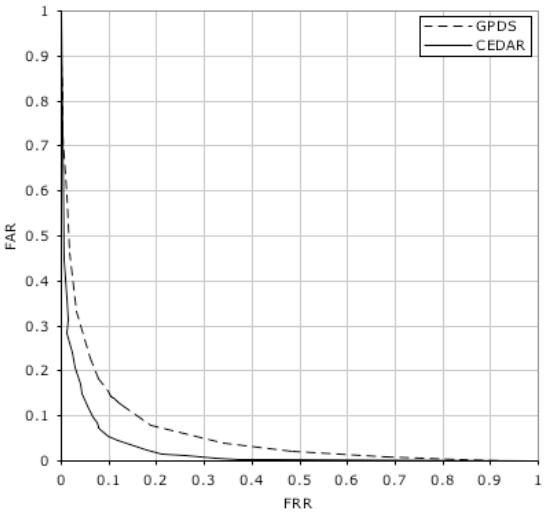


Figure 5: Correlation between FAR and FRR

The first dataset, CEDAR [12], is made up of 55 signature sets, where each set consists of 24 genuine signatures and 24 forgeries. Initially this dataset was in grey-scale, but was converted to binary using a classic iterative thresholding method [13], which was chosen for its simplicity and robustness. The second dataset was GPDS [14], which was already binarised. This dataset contains 39 signature sets, where each set consists of 24 genuine signatures and 30 forgeries.

The experiments for both of these datasets were carried out using three spatial pyramid levels and an 8×4 grid for the finest level of region sampling. The final accuracy is calculated by equation (12) and is the middle point between the genuine and forgery classification accuracies.

$$accuracy = \frac{(1 - FAR) + (1 - FRR)}{2} \quad (12)$$

4.1 CEDAR Results

For each signature set in CEDAR, 16 signatures were randomly selected as training samples, while the remaining 8 genuine signatures along with the 24 forgeries were used for testing. The use of 16 signatures for training was to make the results comparable to [3], [5], and [10]. This was then repeated 10 times for each set. By varying the offset, the error trade-off can be plotted (See Figure 5), allowing the offset that minimises both the FAR and FRR to be identified and used for the final classification accuracy.

The results in Table 1 show that AFT produces competitive results to the graph matching method, with the manual method achieving an accuracy approximately 2% greater than the automatic method. AFT, in comparison to GSC, has a significant accuracy increase of approximately 14%.

Table 1: CEDAR Results

Method	1-FAR	1-FRR	Accuracy
GSC [5]	80.5	77.55	78.5
Zernike [10]	83.7	83.4	83.6
Graph Matching [3]	91.8	92.3	92.1
AFT (Auto)	**89.04**	**91.84**	**90.44**
AFT (Manual)	**92.58**	**92.25**	**92.42**

4.2 GPDS Results

In GPDS, training was carried out in the same fashion as CEDAR, except only 12 randomly selected signatures were used, so that the results would be comparable to [4]. The remaining 12 genuine signatures and 30 forgeries were then used for testing. This was once again repeated 10 times for each set. The offset that minimises both the FAR and FRR was then found in the same way as CEDAR, and was used for the final classification accuracy.

Table 2 shows that AFT can match the classification accuracy that the more complex method DWT is able to achieve. Once again, there is approximately a 2% accuracy difference between the manual and automatic methods.

Table 2: GPDS Results

Method	1-FAR	1-FRR	Accuracy
DWT (Random) [4]	82.08	80.69	81.39
DWT [4]	88.11	86.74	87.43
AFT (Auto)	**82.76**	**89.21**	**85.99**
AFT (Manual)	**85.79**	**89.52**	**87.66**

4.3 Reduced Training Size

AFT also remains fairly stable when the number of training signatures is varied, showing that it is remarkably resilient. This is shown in Figures 6 and 7. The manual method in both CEDAR and GPDS required the offset to be adjusted with relation to the change in the number of training signatures. This change allows the classification accuracy to remain stable when 8 or more training signatures were used; when less than 8 signature were used, the accuracy dropped off fairly quickly. The automatic method tended to consistently follow the same pattern, except it dropped off at a much quicker rate. The stability of the classification accuracy when the number of training signatures varied was not tested in [3], [4], [5], and [10], therefore, the comparable ability of AFT in this regard cannot be determined.

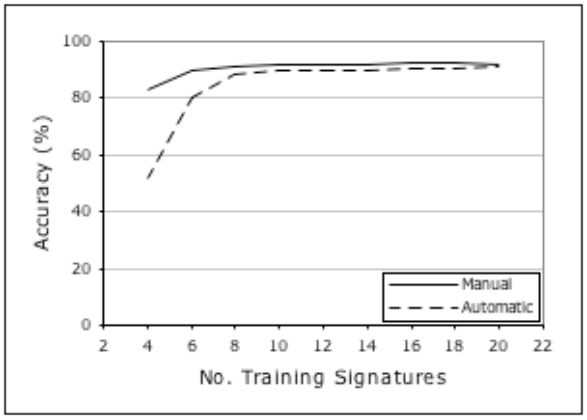


Figure 6: CEDAR stability when the number of training signatures is varied. The standard training amount is 16 signatures.

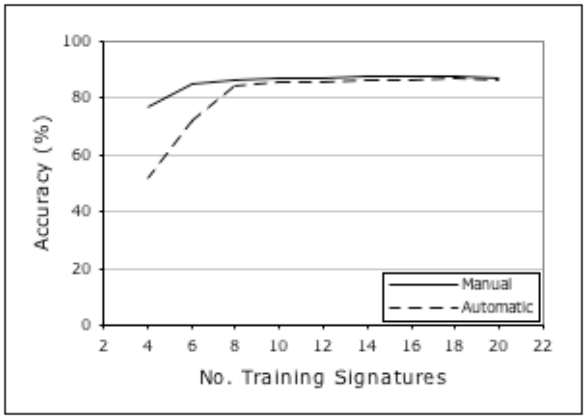


Figure 7: GPDS stability when the number of training signatures is varied. The standard training amount is 12 signatures.

5 Conclusions and Future Work

This paper presented a novel method for off-line signature verification by introducing what has been termed adaptive feature thresholding. AFT is designed to greatly restrict how a binary feature vector is created, improving its representative similarity in relation to the training signatures. Along with AFT, we found that the combination of spatial pyramids and equimass sampling grids helped to improve the extraction and representation of a signature through the use of the gradient direction. Using these methods, AFT achieved a classification accuracy that is competitive to both the graph matching and the DWT methods. As well as this, AFT also remains computationally less complex due to using only one image feature as opposed to the ensemble of image features that other methods tend to use. Experimentation was carried out using

two different approaches, manual and automatic. Manual tended to achieve an accuracy that was 2% greater than automatic. It was also shown that this method remained relatively stable when the number of training signatures was greater than 8.

Possible future work would be to test AFT with other image features, as well as combining it with the graph matching method, as this may further enhance the classification accuracy. The use of AFT in other areas of research is also possible due to its generalist nature, and may prove to be beneficial.

References

[1] A. Jain, A. Ross, and S. Prabhakar, "An Introduction to Biometric Recognition," *Circuits and Systems for Video Technology, IEEE Transactions on*, vol. 14, no. 1, pp. 4–20, 2004.

[2] A. Jain, F. Griess, and S. Connell, "On-line signature verification," *Pattern Recognition*, vol. 35, no. 12, pp. 2963–2972, 2002.

[3] S. Chen and S. Srihari, "A New Off-line Signature Verification Method based on Graph Matching," *Proceedings of the 18th International Conference on Pattern Recognition (ICPR'06)-Volume 02*, pp. 869–872, 2006.

[4] W. Tian, Y. Qiao, and Z. Ma, "A New Scheme for Off-line Signature Verification Using DWT and Fuzzy Net," *Software Engineering, Artificial Intelligence, Networking, and Parallel/Distributed Computing, 2007. SNPD 2007. Eighth ACIS International Conference on*, vol. 3, no. 2, pp. 30–35, 2007.

[5] S. Srihari, A. Xu, and M. Kalera, "Learning strategies and classification methods for off-line signature verification," *Proc. of the 7th Int. Workshop on Frontiers in handwriting recognition (IWHR)*, pp. 161–166, 2004.

[6] B. Zhang and S. Srihari, "Properties of Binary Vector Dissimilarity Measures," *Proc. JCIS Int'l Conf. Computer Vision, Pattern Recognition, and Image Processing*, 2003.

[7] S. Lazebnik, C. Schmid, and J. Ponce, "Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories," *Proc. CVPR*, vol. 2, no. 2169-2178, p. 1, 2006.

[8] J. Favata and G. Srikantan, "A Multiple Feature/Resolution Approach To Handprinted Digit and Character Recognition," *International journal of imaging systems and technology*, vol. 7, no. 4, pp. 304–311, 1996.

[9] G. Srikantan, S. Lam, and S. Srihari, "Gradient-based contour encoding for character recognition," *Pattern Recognition*, vol. 29, no. 7, pp. 1147–1160, 1996.

[10] S. Chen and S. Srihari, "Use of Exterior Contours and Shape Features in Off-line Signature Verification," *Document Analysis and Recognition, 2005. Proceedings. Eighth International Conference on*, pp. 1280–1284, 2005.

[11] B. Zhang, "Off-Line Signature Recognition and Verification by Kernel Principal Component Self-Regression," *Proceedings of the 5th International Conference on Machine Learning and Applications*, pp. 28–33, 2006.

[12] M. Kalera, S. Srihari, and A. Xu, "Offline Signature Verification and Identification Using Distance Statistics," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 18, no. 7, pp. 1339–1360, 2004.

[13] T. Ridler and S. Calvard, "Picture thresholding using an iterative selection method," *IEEE Transactions on Systems, Man and Cybernetics*, vol. 8, no. 8, pp. 630–632, 1978.

[14] M. Ferrer, J. Alonso, and C. Travieso, "Offline geometric parameters for automatic signature verification using fixed-point arithmetic." *IEEE Trans Pattern Anal Mach Intell*, vol. 27, no. 6, pp. 993–997, 2005.