IMPROVED MODELS AND ALGORITHMS IN OFF-LINE, FEATURE-BASED, HANDWRITTEN SIGNATURE VERIFICATION

PhD Thesis

Bence Andras Kovari

Advisor:
Dr. Hassan Charaf Ph.D.
Associate Professor

Budapest University of Technology and Economics
Department of Automation and Applied Informatics

Budapest, 2012

Table of contents

List of tables and figures

Abstract

The verification of handwritten signatures is one of the oldest and the most popular biometric authentication methods in our society. A history which spans several hundred years has ensured that it also has a wide legal acceptance all around the world.

As technology improved, the different ways of comparing and analyzing signatures became more and more sophisticated. Since the early seventies, people have been exploring how computers may aid and – maybe one day – fully take over the task of signature verification. Based on the acquisition process, the field is divided into on-line and off-line parts. In on-line signature verification, the whole process of signing is captured using some kind of an acquisition device, while the off-line approach relies merely on the scanned images of signatures. This thesis addresses some of the many open questions in the off-line field.

As the best current systems use artificial intelligence-based approaches, the insight we get into the reasons behind classification decisions is very limited. To achieve a better understanding of the whole process, our first contribution proposes a simplified probabilistic model for off-line signature verification. In this model, each of the verification steps can be mathematically described and, therefore, individually analyzed and improved. Using this model, it is possible to predict the accuracy of a signature verification system based on just a few a priori known parameters, such as the cardinality and the quality of input samples. Several experiments have been conducted using a statistics-based classifier to confirm the assumptions and the results of our model.

As next, some of the consequences of the previously introduced model are analyzed. It is proven that increasing the number of reference samples or the number of observed features will improve the achievable classification results. It has also been demonstrated that the only free parameter of our equation – which describes the quality of the forged signatures – can be specifically set to minimize the negative effects of a possible misestimation.

The next part of our thesis deals with feature extraction and matching algorithms used during the process of signature verification. A heuristic stroke extraction, a baseline extraction, a loop extraction and a feature-matching algorithm are introduced and described in detail.

In order to demonstrate the practical applications of the results, a complete signature verification framework has been developed, which incorporates all the previously introduced algorithms.

The results provided in this thesis aim to present a deeper analytical insight into the behavior of the verification system than the traditional artificial intelligence-based approaches.

Összefoglaló

A kézeredet azonosítás a legrégebbi és legnépszerűbb biometrikus azonosítási módszerek egyike, mely egyben több évszázados múltjának köszönhetően világszerte széleskörű jogi elfogadottsággal is rendelkezik.

A technológia fejlődésével az aláírások ellenőrzésére is egyre kifinomultabb módszerek születtek, mígnem a 70-es évek kezdetétől a számítógépeket is bevonták a folyamatba. A kezdeti kisegítő funkciókat követően, ma már az ellenőrzés teljes körű automatizálásának a lehetősége áll a kutatások középpontjában. Ezek a kutatások két nagy irányvonalra bonthatóak. Az on-line hitelesítés során az aláírás teljes folyamatát rögzítésre kerül, míg az off-line módszerek kizárólag a már elkészült aláírások szkennelt képeinek az elemzésére korlátozódnak. Dolgozatomban ez utóbbi terület néhány nyitott kérdésére keresem a válaszokat.

Napjaink legsikeresebb aláírás-hitelesítő rendszerei többnyire mesterséges intelligencia (MI) alapú módszerekkel működnek, így a döntések miértjeibe sajnos kevés betekintést kapunk. Hogy mégis jobban megismerhessük a folyamatot, kidolgoztam egy statisztikai modellt a jellemző alapú off-line aláírás-hitelesítéshez. A modell elkülöníti az ellenőrzés fontosabb lépéseit, s lehetővé teszi, hogy azokat önmagukban elemezhessük, fejleszthessük, vagy akár matematikai módszerekkel leírjuk. A modellem ezen felül lehetővé teszi, hogy néhány előre ismert paraméter (mintaszám, jellemzők száma, hamisítvány minősége) alapján előre megbecsüljük egy hitelesítő rendszer várható pontosságát. Modellem helyességét számos méréssel támasztom alá.

Második tézisemben az előző modell matematikai tulajdonságait vizsgáltam. Bebizonyítottam, hogy mind a mintaszám, mind a vizsgált jellemzők számának a növelése kisebb végső hibaarányt fog eredményezni. Ezen felül megfogalmaztam és kísérletekkel alátámasztottam egy sejtést, mely szerint a rendszer egyetlen nem általunk ellenőrzött paraméterének – a hamisítványok minőségének – megfelelő becslésével jelentősen csökkenthető az esetleges tévedésekből eredő hiba mértéke.

Harmadik tézisemben a korábbi eredményekhez szükséges feldolgozási lépéseket tárgyaltam. Egy újfajta heurisztikus vonalkövetési algoritmus mellett eljárásokat adtam az alapvonalak és hurkok kinyerésére, jellemzésére és párosítására is.

A gyakorlati alkalmazhatóság demonstrálásra megvalósítottam egy komplett aláírás-hitelesítési keretrendszert dolgoztam ki, mely – számos egyéb funkción felül – magába foglalja az összes dolgozatomban bemutatott algoritmust.

Eredményeim lehetővé teszik, hogy a hagyományos MI alapú módszereknél mélyebb betekintést nyerhessünk az ellenőrzési folyamat részleteibe.

Preface

Dedication

The content of this thesis is a product of the author's original work except where explicitly stated otherwise.

Nyilatkozat

Alulírott Kővári Bence András kijelentem, hogy ezt a doktori értekezést magam készítettem, és abban csak a megadott forrásokat használtam fel. Minden olyan részt, amelyet szó szerint, vagy azonos tartalomban, de átfogalmazva más forrásból átvettem, egyértelműen, a forrás megadásával megjelöltem.

Budapest, 2012. November 26.

_____

(Kővári Bence András)

Acknowledgements

This thesis could not have been created without the support of many people. First of all I am really grateful to my family, who have been standing by me all my life. Thanks to my friends, who reminded me again and again in different ways about the importance of my studies. I am indebted to my advisor, Hassan Charaf, for facilitating the human and the financial conditions of the work, and no less indebted to Istvan Albert for similar reasons.

I am indebted to Laszlo Lengyel who helped me to find the way in the maze of scientific publications and devoted a significant amount of time to the proofreading of this work.

I am grateful to all of my students who participated in my research in the past seven years. Their devotion and enthusiasm helped me to overcome the encountered dead ends and throw-backs, which unfortunately are the natural companions of basic research.

I would also like to express my gratitude to Mihaly Bokor and Tamas Lukovszki whose advice inspired many improvements in my proofs.

I am indebted to Tamas Agardi, the president of the Hungarian Institute of Graphology, who provided me with useful advice and background material for my research.

I owe a special thanks to Melinda Tünde Dóra, for correcting my spelling and grammar mistakes with infinite patience.

And last, but not least, I would like to thank to my coauthors.

1. Introduction

The verification of handwritten signatures is one of the oldest biometric identification methods. As it gained a high legal acceptance, both the methods of forgers and verifiers became more elaborate. At the beginning of the 20th century, anyone who was in some way professionally connected to handwriting (teachers, notaries) could be treated as handwriting experts [1]. Today, however, being a forensic document examiner is an independent profession, which requires special training and a rich technological toolkit. There are even some guides [2] to modularize and organize the human workflow of the verification process. Despite these facts, the task of signature verification is still challenging, even for a human expert.

When confronted with professionally forged signatures, the average error rate of a forensic expert lies between 0.5% and 7% [3] while non-experts' results may be much worse [4] (error rates between 10% and 26% were measured). These numbers show that even the opinions of forensic document examiners are prone to human errors and that there are still many open questions in the field. Questions such as: Why is it that the experts were unable to achieve better results? Are the previous numbers due to the limits of the verifiers, or the limits of the discriminative power of handwritten signatures? Now, however, rapid advancement of computer sciences has made it possible to analyze some of these questions more thoroughly.

The aim of computer-based signature verification is to automatically decide whether a given signature (questioned signature) belongs to a given person. The decision must only be based on some samples (original signatures) from the signer. Depending on the format of the samples, the field can be divided into two main categories, the on-line (or dynamic) and the off-line (or static) approach.

Figure 1.1 On-line verification requires special acquisition devices

In on-line signature verification, the whole process of signing is captured using some kind of an acquisition device (a camera, a digitizing tablet, a stylus-operated PDA etc.), then analyzed and used to make a decision. The captured information usually includes pen position, pen pressure, pen azimuth and pen inclination as a function of time. This set of data gives automatized verification two important advantages. First, because data is available as a function of time, it is easier to identify corresponding parts, second, this kind of acquisition records information which is not directly available to the forger (like pen velocity, or pressure). Even when producing an almost perfect visual copy of a

signature, these invisible factors will usually significantly differ from the values measured during the original signing process. This is why state of the arte on-line signature verification systems can deliver error rates below 1% [5].

On the other hand, the aim of off-line signature verification is to decide, whether a signature originates from a given signer merely based on the image of the questioned signature and a few images of the original signatures of the signer. This means the input is limited to two-dimensional images while important pieces of information like velocity, inclination or pressure are mostly lost. Unlike on-line signature verification, which requires a special acquisition hardware and setup, off-line verification can be performed separately from the normal signing process and is thereby less intrusive and more user-friendly. Moreover, the off-line scenario can have a much wider range of practical applications as it can be seamlessly fit into many existing workflows. For example, the majority of financial institutions already digitize their contracts, mail or transfer orders, therefore adding automatic signature verification to their workflows would require "just" a software upgrade without directly affecting any of the clients and most of the employees.

While on-line signature verification has a definite advantage over human experts because of the captured non-visual information, off-line verification has a disadvantage, because it works with the same input, but does not have the extensive background knowledge of humans. This is the main reason why even the best off-line signature verifiers on the most studied databases cannot break the 9% error barrier [6] [7]. This limits their practical applicability significantly.

Several different approaches exist to overcome these limitations, but most of the signature verification systems have one thing in common. They behave mostly like black boxes (mainly due to the software's AI-based approach) providing only little meaningful information about the reasons for a decision [8]. This makes their improvement a hard task.

In this work, we introduce an approach more similar to the federal document examiners (FDE) [2] who compare specific features of a signature individually to make their final decision. This approach also allows us to create a probabilistic model for our verification system. Using this probabilistic model we are not only able to create a classifier for our verification system, but also to make a good estimation about the accuracy of the system. For example, we are able to directly calculate how a change in the number of original samples, or the quality of forgeries will influence the final classification results without actually performing the tests.

1.1 Thesis contributions

In the next sections we propose a formal approach for off-line signature verification. By breaking down the process into smaller, loosely coupled tasks, and by ensuring a controlled information flow, it allows us to separately model and benchmark our components, and to identify and measure the different sources of errors. In contrast to usual monolith approaches [8] it not only allows us to better understand the weaknesses of a given verification approach but also

provides means to predict the performance of a system based on parameters of the database.

The objective of my research was to establish a signature model as well as new evaluation methodologies, suitable for differentiating between original and forged signatures. Taking these aspects into account, my goals – addressed by this thesis – were the following:

- Investigate the common characteristics in existing off-line signature verification approaches and identify their strong and weak points
- Examine and – when necessary – improve existing feature extraction algorithms
- Identify the dominant features of signatures to be used to discriminate between original and forged ones.
- Establish and investigate classification techniques which may be used to differentiate between original and forged signatures
- Apply the methods to common signature databases and demonstrate the feasibility of practical application

1.2 Thesis structure

The rest of this thesis is organized as follows.

- Chapter 2 is devoted to illustrate the motivations of the results in this thesis. This chapter serves as a general introduction to the topic of off-line signature verification. It also outlines the most important approaches in the field. A modular and streamlined signature verification architecture is introduced and adapted to feature based verification. In the closing part of the chapter, we examine some of the major signature corpuses which were also used to test our algorithms.
- In Chapters 3-4, after reviewing the most relevant related work, we present the major contributions of this thesis. A statistical model is introduced, which is able to describe several important aspects of a signature database and predict their effect on the accuracy of the classification phase. Both the basic assumptions and the effects of the model are validated through a number of experiments.
- Chapter 5 summarizes the improvements we made to existing feature-extraction and -matching algorithms, which contributed to the results introduced in the previous chapters.
- To illustrate the practical relevance of this research, the application of the results is shown in Chapter 6. In the past 7 years, a rich infrastructure has been built around our signature verification core. This chapter briefly introduces the most important ones.
- The last chapter (Chapter 7) is devoted to the summary and the outline of future work.
- In the Appendix, some detailed proofs are given, along with sample signatures from the major signature databases.

2. Motivation and background

This chapter introduces the basic concepts of off-line signature verification and gives an overview of the current state of the art.

2.1 Terms and concepts

As automatized signature verification is still in development, some terms in the field may have ambiguous interpretations. To avoid miscomprehension, throughout this thesis we are going to use the following interpretations of basic terms.

Definition 2.1. (basic terms):

Signature: A person's name written as a form of identification in authorizing a document.

Signer: The creator of a signature.

Original signature (genuine signature): a signature originating from a given signer.

Forged signature (forgery): a signature which imitates the signature of a given signer but does not originate from him/her. Forgeries can be divided into several subclasses

- **random forgery:** a signature, with negligible similarities to the reference signatures. In common testing scenarios a random forgery is just a randomly chosen signature of another signer,
- **simple forgery:** a forgery, which was created with the knowledge of the name of the original signer, but without access to the reference signatures,
- **synthetic forgery:** an artificially generated signature, which was created by distorting some features of the reference signatures,
- **skilled forgery:** a forgery created after a thorough study of the reference signatures,
- **reference signature:** a signature of a signer known to the verification system. This kind of signature is used to train a verification system.

Sample signature (sample, questioned signature): a signature whose signer is not known to the verification system. This kind of signature is used to test a verification system.

Original sample: an original signature used as a sample.

Forged sample: a forged signature used as a sample.

Accepted sample: a sample, which was classified as an original signature by the verification system.

Rejected sample: a sample, which was classified as a forgery by the verification system.

Inconclusive sample: a sample upon which the system was unable to make a decision about the origin of the signature.

2.2 Performance metrics

As most biometric authentication systems, the performance of signature verifiers is usually measured in terms of error rates [9] [10]. Different error rates can be derived from statistical hypothesis testing, where the null hypothesis is:

H_0: "The sample signature originates from the same signer as the reference signatures"

and the possible decisions are a definitive "accept" or "reject". We should note that in practical applications an "inconclusive" answer may be also allowed, but in our case this would severely limit the comparability of different verification systems.

Definition 2.2. (Type I error):

A type I error, also known as error of the first kind occurs when the null hypothesis (H_0) is true, but is rejected. In signature verification, the rate of type I error is often called False Rejection Rate (FRR) and is denoted by α.

$$\alpha = \frac{number\ of\ rejected\ original\ samples}{total\ number\ of\ original\ samples}$$

Definition 2.3. (Type II error):

A type II error, also known as an error of the second kind, occurs when the null hypothesis (H_0) is false, but it is erroneously accepted as true. In signature verification, the rate of type II error is often called False Acceptance Rate (FAR) and is denoted by β.

$$\beta = \frac{number\ of\ accepted\ forged\ samples}{total\ number\ of\ forged\ samples}$$

Definition 2.4. (Equal error rate):

Each verification system can be tuned to a level at which both accept and reject errors are equal. This rate is called equal error rate (EER).

EER provides a quick way to compare the accuracy of verification systems. In general, a system with lower EER is considered to be more accurate.

Definition 2.5. (Average error rate):

Average error rate (AER) is the average of FAR and FRR for a given experimental configuration.

In several cases, it is not feasible or possible to calculate or measure the exact EER (e.g. some papers only provide FAR and FRR measurements). For most systems considered in this thesis, the difference between the reported FAR and FRR values is small (mostly below 5%), therefore AER may be used as an admissible approximation of EER.

2.3 The human factor

Signature verification is a behavioral biometric authentication method. This means that unlike fingerprints or iris patterns, which cannot be influenced by the subject, signatures are the results of conscious human behavior and can thereby be influenced by a wide range of factors [11].

On the positive side, this allows the seamless combination of biometric and knowledge-based authentication. While we currently limit our studies to written names, these algorithms could be easily extended to process any word, or combination of words written in the own handwriting of the person.

On the negative side, the conscious nature of the signing process can result in a wide range of variations in the signature of the same person. Fatigue, mood, glasses, pen, writing position and even the alignment of the form can affect the details of the signature. The basic assumption of signature verification is that there are some unconscious features, which stay mostly stable despite the previously mentioned factors. Moreover, some of these features (like the tremor of strokes) are extremely hard to forge by freehand methods. The main challenge is to find algorithms that can efficiently and universally locate, extract and compare such features.

2.4 Artificial forgeries

There is a wide variety of methods to forge signatures with artificial aids. Modern photocopiers or special plotters can create forgeries that are perfectly indistinguishable from the original signature to the naked eye. These methods always leave behind traces easily recognizable at a higher magnification rate, or by the analysis of color histograms. Our thesis will therefore be limited to freehand forgeries by humans.

2.5 State of the art

At the First International Signature Verification Competition [12], 16 different **on-line** signature verification systems were evaluated under similar conditions, among them those of the leading research groups in the field. The best system delivered an average EER of 2.8% while the second best achieved 4.4%. Some studies suggest, that even better error rates can be achieved such as 0.35% in [13] or 0.4% in [14], however, these systems were evaluated on different datasets.

The **off-line** verification of signatures is even harder to tackle. While the on-line problem can mostly be reduced to matching and comparing one-dimensional functions (pen velocity, vertical, horizontal positions and all other measurements can be expressed as functions of time), the input for the off-line

decision is only a single two-dimensional function, describing the intensity (or the color components) of a given pixel at a given vertical and horizontal position for an image. On-line capturing devices are also able to record additional information like pen pressure, azimuth, and altitude, which have been shown [15] to improve the final accuracy of a system.

Lacking a common signature corpus and a well-defined evaluation methodology, the results of different studies are difficult to compare; therefore, reported error rates should be only taken as approximate.

To highlight some common problems of comparison, we should consider the following papers. In [16] the authors achieved an FRR of 3% using a neural network classifier, but they only used 160 signatures from 10 signers. In addition, they only performed a recognition task (their system had to assign each of the 60 test signatures to one of the previously known signers). [17] reports an EER of 1% on random forgeries, however, this value drops to 19,8%, when using skilled forgeries. To allow for a mostly uniform comparison of different systems and the filtering of results irrelevant for our thesis, we shall formulate the following preconditions:

- A verification system should not take advantage of the fact that it has knowledge of all possible signers, as this is not the case in real world scenarios.

- We do not have any forgeries for a given signer during the training. Therefore the training phase should not take advantage of human-created forged signatures (synthetic forgeries are allowed)

- We assume that a system that is able to detect skilled forgeries will perform even better on simple or random forgeries. Therefore, we limit our analysis to skilled forgeries.

Based on the surveys, which together cover more than 500 papers from the last 30 years [5] [18] [19] [20] [21] and on [6] [7] covering some of the latest developments, it is safe to conclude that the best equal error rate currently achievable by off-line signature verifiers for skilled forgeries is approximately 9%. Some of the best results were achieved by [22] (FRR: 11.32% FAR: 6.48%), [6] (EER: 9.02% on GPDS-100 corpus and 8.8% on MCYT corpus[1]) and [23] (EER: 10.65%).

To allow a better understanding of these results, they should be compared to results achieved by human experts. A 3 year long study of 28 FDEs, examining 11643 signatures (3387 originals and 8256 skilled forgeries) has shown, that human experts are able to confidently identify original signatures with a false rejection rate below 7% [3]. Unfortunately, this study also allowed "inconclusive" answers, therefore the evaluation of FAR was not possible. In [24] a study was performed on 432 genuine and 333 forged signatures from 51

[1] Siganture databases are discussed in detail in section 2.10.

writers, concluding that (non-expert) humans performance is FRR: 9% and FAR: 10.5%. In other studies experts achieved 0.5% FAR and 7% FRR, compared to the 6.5% FAR and 26% FRR of a layperson [25] [4].

As long as humans outperform automatic signature verifiers, there is space for improvement. In the next section, we examine the structure of major existing verification systems and identify the possible means of improvement.

2.6 The verification process

The majority of signature verification methods can be divided into five main phases: acquisition, preprocessing, feature extraction, processing and classification (although these steps are not always separable). In an off-line scenario data acquisition is simply the scanning of a signature. This is followed by preprocessing, whereby the images of signatures are altered (cropped, stretched, resized, normalized, etc.) to create a suitable input for the next phase. The next step is feature extraction, the process of identifying characteristics inherent to the particular person. The processing phase prepares the raw data extracted from the signature for classification. Using these results, the classification phase is able to make a decision whether to accept or reject the tested signature. This coarse separation of processing phases is already an extension to [4] which does not separate feature extraction from processing and classification. In our model, even further extensions will be necessary to allow a better control of the dataflow. In the following subsections, these 5 steps will be explained in detail and matched to the steps of several other signature verifiers.

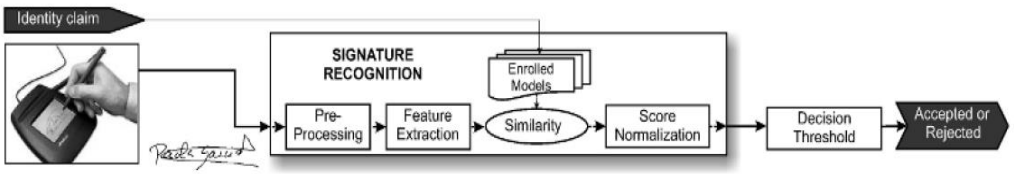


Figure 2.1 A typical off-line verifier with a threshold classifier. Source: [26]

2.6.1 Acquisition

In general, the acquisition step converts a number of paper sheets to a set of digital images, each of them containing one or more signatures. It is essential to note that scanning paper sheets with written signatures is not the only way to acquire digital images. As noted in [27], samples can also be generated from on-line databases or by altering existing signatures [28]. Of course, these later methods cannot be used to validate the whole signature verification system; however they usually contain valuable additional information (like the correct order, direction and position of strokes), and this can be used to benchmark separate parts of the system. In most of the observed systems, the digitalized images of the signatures are assumed to be present, therefore the acquisition phase is usually not part of system diagrams.

2.6.2 Preprocessing

The preprocessing phase is a sequence of image transformations creating the best possible input for feature extraction algorithms. In on-line signature

verifiers, the acquired data is usually already in an optimal form for further processing, therefore this phase is superfluous [29] [30] (see also Figure 2.4). In the off-line case it is usually necessary to eliminate the noise introduced during the acquisition phase [31]. Some preprocessing steps such as noise filtering, rotation normalization and position normalization induce only minimal information loss, while others, like binarization, morphological closing or size normalization can cause the loss of valuable information. Thus the second class of preprocessing steps is only applied where the feature extraction algorithm directly benefits from them. Although preprocessing is used in all of the examined off-line verifiers, only three [26] [32] [33] display it on their system diagram.

2.6.3 Feature extraction

"An image feature is a distinguishing primitive characteristic or attribute of an image. Some features are natural in the sense that such features are defined by the visual appearance of an image, while other, artificial features result from specific manipulations of an image […] Image features are of major importance in the isolation of regions of common property within an image (image segmentation) and subsequent identification or labeling of such regions (image classification)." [34]. Therefore feature extraction is the location and characterization of features, and generally it should not be confused with later processing phases. Contrary to preprocessing, which is defined as a sequence of transformation steps altering the original images, feature extraction is a set of (usually) independent functions returning a characteristic feature set for their input image. Several systems (like those on Figure 2.2) take advantage of multiple features to improve the quality of the input provided for distance calculations and classifiers.

Features can either be particular to the whole signature (global features) or to a part of the signature (topological or local features) [35].

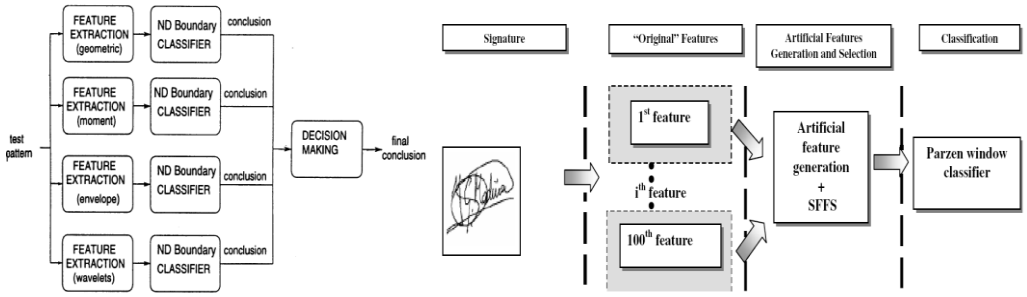


Figure 2.2 Multiple features are used to allow a refined decision. Sources: [36] [37]

2.6.4 Processing

The processing phase differs from the previous ones in that it can work with multiple images. First, a matching is defined between the features of the different images, then a distance (or similarity) measure is calculated based on the corresponding features and finally this measure may be normalized to make it a suitable input for a classifier. All of the three steps can be identified in Figure 2.1, while the other systems do not mention all of them. This can be explained by a different interpretation of phases, for example the feature

extraction boxes usually also represent the processing phase (Figure 2.2) and score normalization is sometimes considered to be part of the classification phase.

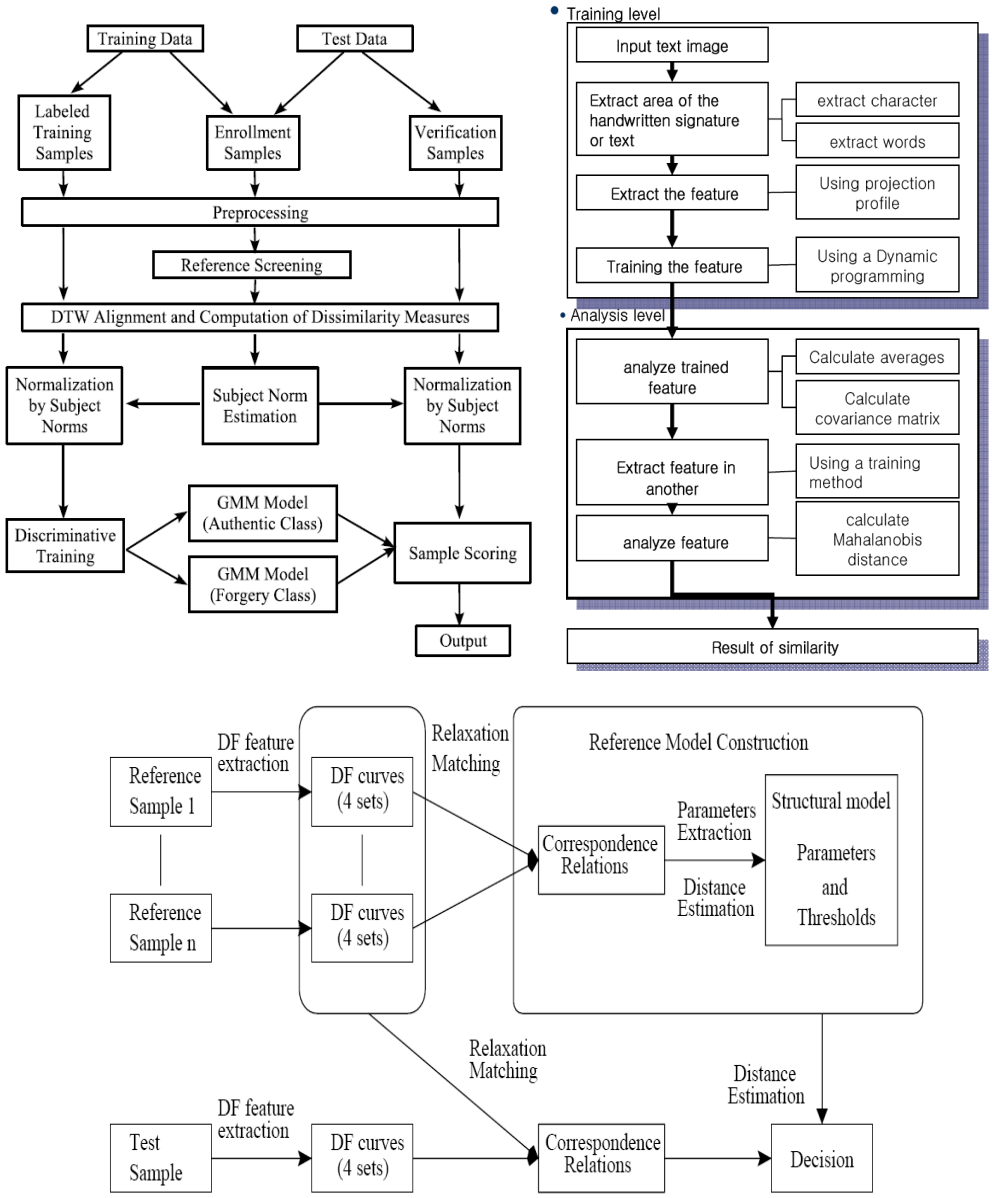


Figure 2.3 Separation of training and testing paths. Sources: [38] [32] [33]

2.6.5 Classification

In the classification phase, a single classifier is trained with reference signatures. Based on the training, the classifier can make decisions about the acceptance or rejection of a sample signature.

The complexity of classifiers can vary from checking a single threshold by using a hidden Markov model (HMM) to involving a neural network or Support Vector Machine (SVM) in the decision. These methods can also be combined in a composite system to allow the decision to be made with a deeper understanding of the context.

From the aspect of applicability it is essential to see whether the classification methodology only requires the presence of original signatures (a one-class problem) or both originals and forgeries (a two-class problem). The latter usually generates better results [39] [40], especially for small databases, but in "real world" scenarios, it can only be used if synthetic forgeries for each signer are provided.

There seems to be a major trend towards using general classifiers, like neural networks or SVM instead of concentrating on semantically interpretable distance measures. One major drawback of this is the increase in the number of two-class-problems.

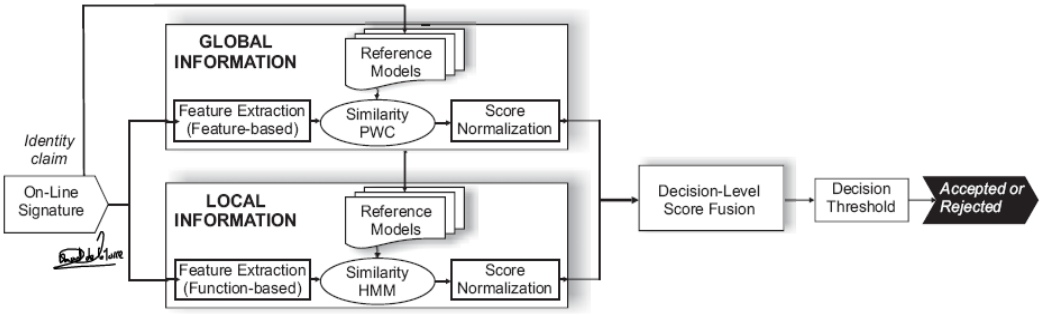


Figure 2.4 Global statistics can help interpret local information. Source: [41]

2.7 Off-line signature verification systems

This section summarizes some of the best off-line signature verifiers introduced in the last two decades. To allow a better understanding of the results, we also briefly describe the algorithmic approach as well as the signature databases used by these systems. A detailed description of the databases can be found in section 2.10.

Table 2.1 Comparison table of off-line signature verifiers

	Year	EER (rand)	Processing	Classification	Classes	Signers	Original signatures	Skilled forgery (simple forgery)
[42]	1995	(9.14)	Fuzzy ARTMAP		one	5	200	
[43]	1996	14.3 (3)	Structuring elements	Linear classifier	one	55	550	300 (100)
[44]	1997	(0.84)	Shape factor, shape matrices	Distance function	one	20	800	
[45]	1999	24.5	Extracted displacement fn.	Threshold	one	20	200	200
[46]	1999	(14)	HMM	HMM	one	60	1440	
[47]	2000	(0.9)	Horizontal, vertical segmentation	HMM, majority vote	one	100	4000	

	Year	EER (rand)	Processing	Classification	Classes	Signers	Original signatures	Skilled forgery (simple forgery)
[48]	2000	9**	Edges, directions, model	Thresholds	one	10	50	200 (100)
[49]	2002	5.9	Columns as vectors	HMM	one	100	4000	1200
[50]	2002	18.1	Projection, DPM	Threshold	one	55 CEDAR	1320	1320
[51]	2003	9.1	Modified Pattern Matching	Threshold	one	10	1000	1000
[40]	2004	9.3	Geometry, strokes, topology	SVM	two	55 CEDAR	1320	1320
[40]	2004	21	Geometry, strokes, topology	Distance methods	one	55 CEDAR	1320	1320
[52]	2004	9.6	Direct image	SVM (RBF)	one	40	160	
[53]	2005	13.3*	Signature envelope shape	HMM	two	160 GPDS	3840	4800
[39]	2005	10	Density, moment, structure	multiple feed forward neural networks	two	30	300	150
[55]	2006	8.9	Modified Direction Feature	Neural Network (RBP, RBF)	two	39 GPDS	936	1170
[56]	2010	15.1	Gradient features	SVM (Gaussian kernel)	one	160 GPDS	3840	4800
[57]	2010	10.9	Gradient features	SVM (Gaussian kernel)	one	400 GPDS	9600	12000
[6]	2011	9.4	LBP+GLCM features	LS-SVM	one	75 MCYT	1125	1125
[6]	2011	9	LBP+GLCM features	LS-SVM	one	100 GPDS	2400	3000
[7]	2012	20.6		HMM+SVM	one	160 GPDS	3840	4800

* estimation based on FAR and FRR in the original paper

Abbreviations:

- RBP: resilent backpropagation
- RBF: radial basis function
- LBP: local binary pattern
- GLCM: grey level co-occurrence matrix
- MDF: modified direction feature
- SVM: support vector machine
- LS-SVM: least square support vector machine
- GPDS, CEDAR, MCYT: public signature databases, see section 2.10

2.8 Local features in signature verification

Table 2.1 shows that the major signature verifiers tend to work with global features like shape factors as well as different projections of the signature. Although this approach eliminates the need for complex matching algorithms, it also reduces the control over how individual parts of the signature are compared. Although using global features and advanced AI-based classification methods can yield error rates between 9% and 20%, it is hard to further improve these methods, as the source of error is difficult to pinpoint. To break the 9% EER barrier, it is essential to identify and understand the different sources of error in the algorithms. Although such research has been conducted for on-line signatures [15], [58], the off-line scenario is essentially unanalyzed.

By using local (topological [35]) features, classification errors can almost always be traced directly back to specific errors in the processing chain that can be corrected, thereby improving the whole system. The best solutions in the two other major fields of biometrics – fingerprint and face recognition – are all based on matching some predefined features in the samples.

Based on the observations of the previous section, a generalized model for off-line signature verifiers is proposed [59].

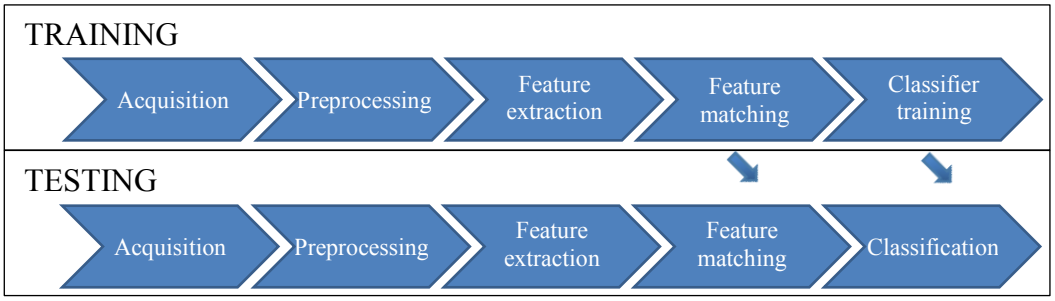


Figure 2.5 Feature-based off-line signature verification process

The model integrates the advantages of the previously introduced systems and their representations. Similarly to [38] and [33], it isolates the path of the training set (original signatures) from the path of test signatures while traversing the same modules of the system. The classifier is trained during the training phase only using the original signatures from the reference database, and classification decisions about test signatures will be made based on the training during the testing phase.

The model allows a straight data flow without feedback. This makes the loose coupling and individual testing of the components possible and allows the monitoring of error propagation.

As noted previously, individual phases can consist of multiple sequentially or parallel coupled subcomponents.

2.9 The feature matching problem

To allow the comparison of two signatures, the values extracted from one signature must be matched to the values of the other signature. This is a prerequisite for almost any classification approach. When working with global features (like width or height of a signature) this mapping can be defined easily. On the other hand, local features tend to be much more unstable. Some signatures of a signer may have 3 loops, while others may contain 4 loops. Without further preprocessing, this could result in input vectors of different lengths for the classifier. In Figure 2.5 the processing phase is represented by "feature matching" to emphasize the importance of this step. We are going to elaborate on this subject in section 5.1.3.

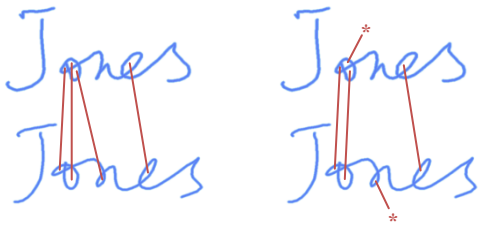


Figure 2.6 Incorrect (left) and correct (right) matching of loops

2.10 Test databases, preprocessing, experimental setup

Verification algorithms are developed and evaluated with, and tested against signature databases. A signature database (or signature corpus) is a set of digitalized signatures from a range of subjects. The database is commonly divided into a training set and a validation or test set, thereby eliminating the direct influence of the learning process on the verification process. A database can be characterized by a number of factors:

- Number and type of signers: these factors ensure the diversity of the database. To allow for a representative testing of algorithms, the number of signers should be as high as possible and not limited to a single social group (like students). Only a handful of public databases contain more than 100 signers. In [46] and [39], some of the experiments were repeated with an increasing number of signers (10-20-25-30 signers) and it always resulted in a worse EER value.
- Number of original signatures: this number should be high enough to ensure a set for training and a disjoint set for testing the signature verifier.
- Quality of forgeries: forgeries may be simple or skilled. Skilled ones are preferred, as their distinction from the original is a harder task.
- Origin of forgeries: Skilled forgeries may originate from one or from more forgers. The latter is preferred, as it increases the diversity of the test set.
- Number of forgeries: the number of forgeries should be in the same magnitude as the number of original samples in the test set.

- Resolution: an increased resolution allows better processing, while it also increases the required computational resource and storage space. Usual resolutions are 300dpi and 600dpi.
- Color depth: most of the public databases contain binary (black and white) images. Greyscale or color images increase storage requirements, but contain important information about the dynamics of the signing process.

In the following sections we introduce our test databases and some of the major public databases used or referenced in this thesis.

2.10.1 BME-AUT1

The BME-AUT1 [60] database is a collection of 1400 signatures. It contains 20 originals and 8 skilled forgeries for each of 50 signers. The 32-bit color images were scanned at 600dpi. The forgeries are of unusually good quality as they were all produced by handwriting experts. We used these samples during the first years of our research. The database is freely available for research purposes at https://www.aut.bme.hu/Signature.

2.10.2 BME-AUT2

The BME-AUT2 [60] database contains 50 original signatures from each 50 signers. This database contains no forgeries. The high quality (600dpi, 32-bit color) images were used in the statistical analysis of our signature model. The database is freely available for research purposes at https://www.aut.bme.hu/Signature.

2.10.3 BME-AUT9

The BME-AUT9 is a smaller database mainly used in the fine-tuning of our feature extraction algorithms. It contains 16 original and 16 forged signatures for each of the 9 signers. The images are available at 600 dpi resolution with 32-bit color depth. As the signers are teachers from the university, this database is not available for public use.

2.10.4 SVC2004

The SVC2004 is a public signature corpus, created for the First Signature Verification Competition [12]. This is an on-line signature database; therefore, it contains stroke information without the images themselves. The stroke information was used to synthesize signatures similar to the original ones. Stroke points were connected with straight lines, fading out on the line borders. Bicubic interpolation and anti-aliasing were used to make the final image smoother. Eight hundred Latin-type signatures from 20 signers (20 originals and 20 forgeries from each) were used to test our feature extraction and classification algorithms. Appendix C.1 contains a range of image samples from this database.

2.10.5 GPDS

The database created by Grupo de Procesado Digital de Señales at the University of Las Palmas in Gran Canaria, Spain is the largest public corpus for off-line signature verification [61]. Signatures were collected from 960 different signers. 24 original and 30 forged signatures ensure enough samples for both

training and testing different verification systems. As the database was created incrementally, several publications may refer to GPDS100, or GPDS300 meaning thereby the first 100 or 300 signers of the database. Although the range of signers is very wide, the 300dpi binary images are of very low quality.

Also, the majority of the signatures contain wide loops covering a larger part of the signature. Such features are (according to our experience) not representative for the global population. Such signatures favor global feature-based approaches, while local feature-based approaches could get confused by the large variety of line crossings.

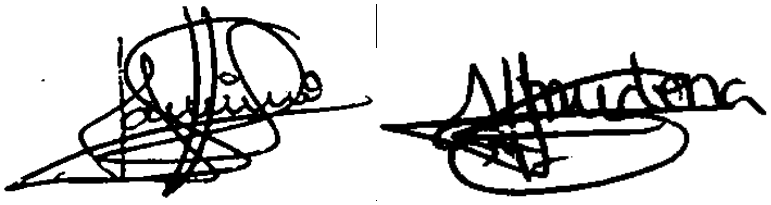


Figure 2.7 Samples from the GPDS database

2.10.6 MCYT

The MCYT is a multimodal biometric database containing fingerprint and signature data from 330 contributors collected on 4 different Spanish sites [62]. The MCYT-75 off-line signature corpus is a publicly available subset of the database containing images for 15 original and 15 forged signatures of 75 signers. The 2250 images were scanned at a resolution of 600 dpi.

2.10.7 CEDAR

The database [40] of the Center of Excellence of Document Analysis and Recognition at the University at Buffalo, the State University of New York is a collection of 2640 signatures containing 24 original and 24 forged samples from 55 signers. The 600dpi greyscale images are publicly available.

3. The statistical model of signature verification

We have already discussed in section 2.6.5 that there is an overwhelming trend of using AI-based classifiers. The reason behind this is simple: we do not know exactly what we are looking for. The basic assumption behind most of these approaches is that the feature values of original signatures will cluster around specific means and these clusters will be separable from the feature values of forgeries. Although these assumptions are mainly confirmed by the experimental results, a deeper investigation of the reasons behind it is rarely conducted.

In this section, we review the related work and establish the basic terminology, next we present and refine a novel classification approach, purely based on the statistical characterization of features. All theoretical results are presented in conjunction with their experimental validations.

3.1 Related work

As an example for statistical methods, in [63] the Kolmogorov-Smirnov Test is used for obtaining a probability of similarity between the distribution of the reference signatures and the sample signatures in distance space. The achieved error rate was 16.4% on the CEDAR database, but no further efforts were done to refine the model or to examine the distributions of feature values.

A more thorough study of the statistical approach was presented in [64]. The basic assumption of the paper is that when measuring any physiological characteristic of a person[2], the values will be distributed according to the normal distribution $\mathcal{N}(\mu_i, \sigma_i^2)$ and that the entire population is also similarly distributed but with parameters μ_p and σ_p. We can use this to calculate the statistical probability of a given sample belonging to a given person. A key element influencing the accuracy of the system is the proportion of σ_i and σ_p, which they described with $\sigma = \sigma_i / \sqrt{\sigma_i^2 + \sigma_p^2}$.

Our approach is very similar, but we made three major improvements. First, we focus on verification instead of identification. Therefore, while we do not take advantage of the knowledge of the population, we assume knowledge of the distribution of forgeries $\mathcal{N}(\mu_i', \sigma_i'^2)$. Second, we will use the quotient $q = \sigma_i / \sigma_i'$ to describe the relation of standard deviations, which is in our opinion a more intuitive descriptor for the quality of forgeries. Third, and most importantly, we will show that a low sample size (the paper mentions sample sizes between 3 and 5) significantly influences the accuracy of classification and should therefore be incorporated into calculations. In addition, we will perform several tests of the assumption of normally distributed variables.

[2] The paper was not directly focused on signature verification, but considered all kinds of biometric authentication and identification.

It is also widely accepted that the accuracy of a biometric system can be improved by increasing the number and diversity of features [65]. Several approaches (like [23]) have significantly improved this way, but the effect of the number of features on the final results is always established empirically. We are going to show that as long as the features remain mostly independent, their effect on the final results can be modeled and estimated without actually performing the verification.

3.2 The terminology of features

After acquisition and pre-processing, several features are located and their properties are extracted. Feature properties are quantitative descriptors of different aspects of the signature (e.g., height of the first loop, pitch of the signature, etc.). We refer to a class of similar features as "feature types". For example, "loop" is a feature type, as it represents a generic characteristic element of signatures. Instances of a feature type will be called "features" (e.g., in Figure 2.6, there are four loops in a signature). A feature is described by several quantities (a loop can have a height, a width, an area, etc.). These quantities will be called feature properties.

A special group of feature types is characteristic for the whole signature (width, pitch, etc.). These feature types within the group are called global feature types. All of the other feature types, which may have multiple instances within a single signature (e.g., loops or line crossings), are called local feature types. Common signature verification approaches may take advantage of both global and local types [36].

Definition 3.1. (feature-related terms):

Feature: a feature is a part of the signature described by a set of properties. We distinguish two types of features:

- **global feature**: a feature which covers the whole signature
- **local feature**: a feature which only covers a smaller part of the signature

Feature property: a number describing a given aspect of a feature

Feature type: describes a group of features that have the same properties

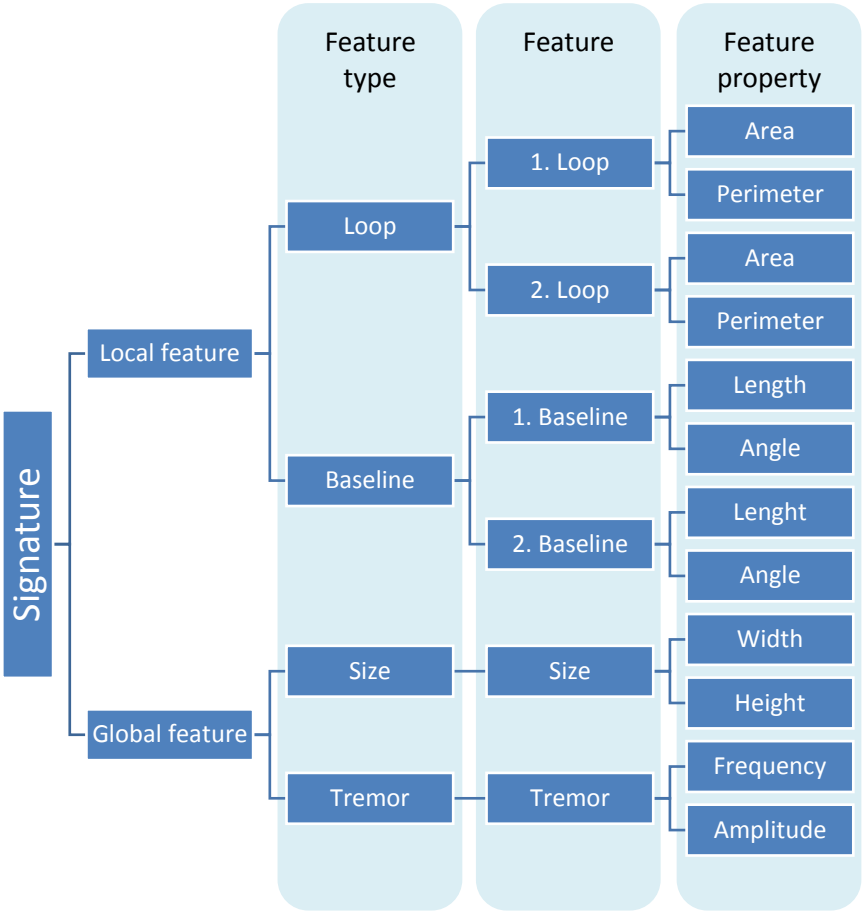


Figure 3.1 Feature hierarchy of signatures (example)

Although there is a huge repertoire of feature types in the literature, our research focuses only on the ones that are easily interpretable by humans. This process most closely matches the FDEs approach and fosters a better intuitive understanding of the final results. In our current experiments, two feature types are included: baselines and loops. Both types were chosen and defined using the knowledge gained in several consultations with handwriting experts [8].

3.3 Local feature-based classification

Our current approach is based on the statistical characterization of feature properties. First, we introduce a method for a single feature property, analyze and confirm our assumptions, and then we generalize the method to multiple feature properties.

3.4 The distribution of feature properties in original signatures

Assumption 3.1

A feature property can be approximated as a normally distributed random variable X, with mean m and variance s^2

$$X \sim \mathcal{N}(m, s) \tag{3.1}$$

Remark: It is natural to assume that the signer is aiming to reproduce his own signature as accurately as possible; therefore, a given feature property will most likely vary around a given mean. Additionally, the exact result is influenced by a large number of different independent factors. These conditions in natural sciences are usually enough to warrant the use of a normal distribution to (at least approximately) characterize a variable. This assumption was also used in [64]. In the next sections, we demonstrate with several experiments that this approximation is precise enough for the purposes of signature verification.

3.5 Experiments

The main purpose of this experiment is to support the assumption of normally distributed feature properties. The experiments were performed on 20 original signatures from each of the 20 signers from the SVC2004 database. During the feature extraction and matching steps, 35 baselines and 37 loops were identified from the original signatures of signers. Baselines consisted of 6 properties and loops consisted of 14 properties.

Two different normality tests were selected to check Assumption 3.1. The Lilliefors adaptation of the Kolmogorov-Smirnov test [66] and the Shapiro-Wilk test [67] are both known to perform well on low sample sizes; therefore, all of the data series were tested with both of these tests. Figure 3.2-Figure 3.5 provide a simplified view of the results. The results are grouped by feature type (e.g., Figure 3.2 and Figure 3.3 show the results for the baselines). Each column represents a baseline in a signature, and each row represents a feature property (e.g., X position) of the baseline. If a given cell is dark, the given property of the given feature did not pass the normality test (the null hypothesis was rejected with a 0.05 significance level).

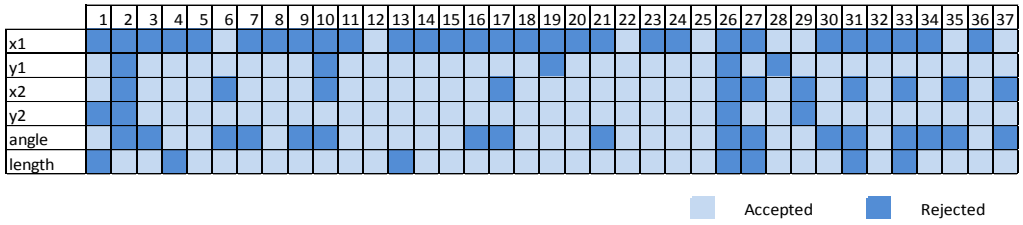


Figure 3.2. The Shapiro-Wilk test for baselines

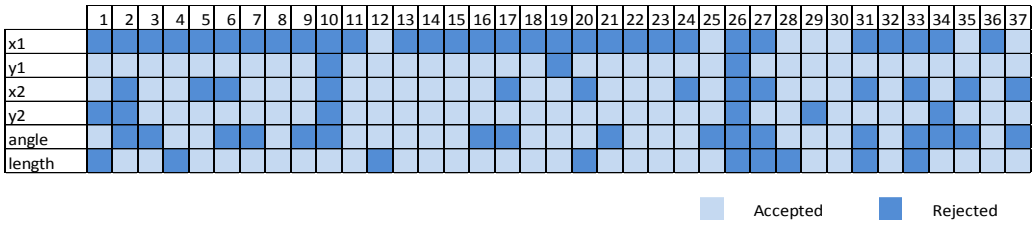


Figure 3.3. The Lilliefors test for baselines

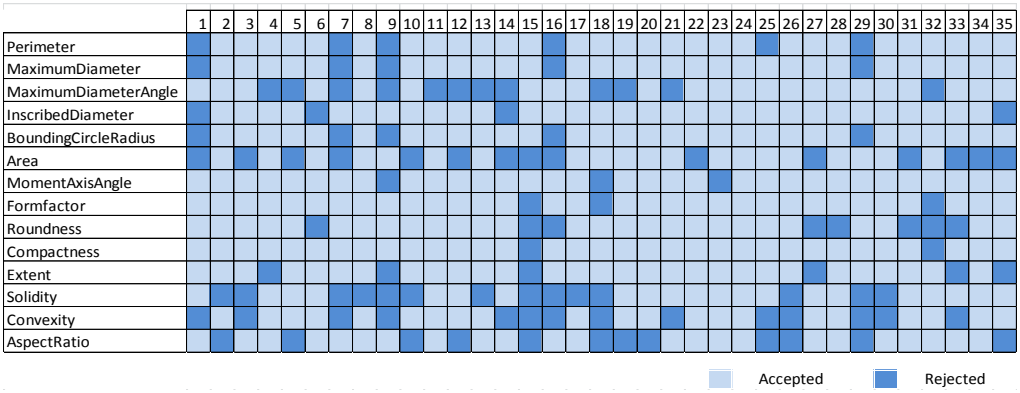


Figure 3.4. The Shapiro-Wilk test for loops

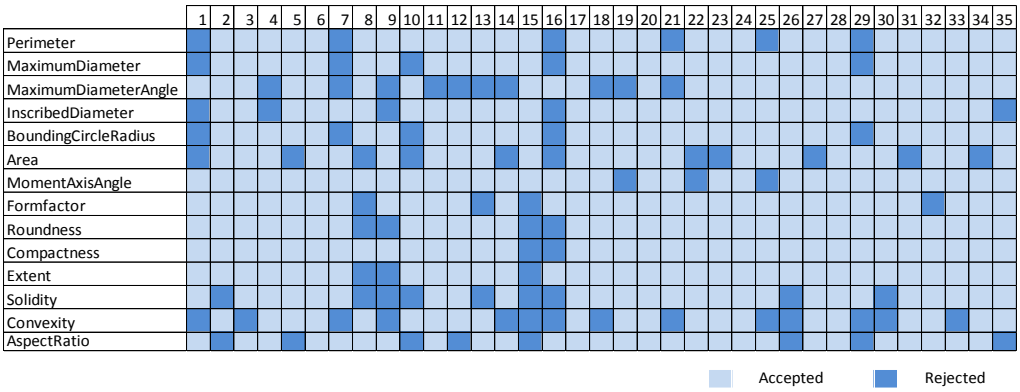


Figure 3.5. The Lilliefors test for loops

As a side effect of our trimming algorithm, the horizontal position (x1) of the first baseline was almost always chosen to be a fixed value. Therefore, it almost always failed the tests (Figure 3.2 and Figure 3.3). Some invalid matches could also be identified by the increased number of errors in a given column (columns 2, 10, and 26 on Figure 3.2 and Figure 3.3 and columns 15 and 16 in Figure 3.4 and Figure 3.5). These matching errors were also confirmed by a visual analysis of the signatures.

Altogether, if the first parameter of the baselines was omitted, the normality tests would have only failed in 20% of the cases for the Shapiro-Wilk test and in 22% of the cases in the Lilliefors test. Yet a significant portion of errors is explained by errors introduced previously in the processing phases. Visual inspection showed that 90% of the errors were introduced because of an erroneous matching of features or errors in the feature extraction algorithms. Only 10% of all of the failed normality tests can directly be explained by an irregular distribution of feature attributes. Although these results should not be considered a direct proof of normality, we consider these results encouraging enough to use Assumption 3.1 to approximate the characteristics of feature properties.

3.6 The distribution of feature properties in forged signatures

Assumption 3.2

A forged feature property can be approximated as a normally distributed random variable Y.

$$Y \sim \mathcal{N}(m_f, s_f) \tag{3.2}$$

In addition, because the forger is aiming to forge the original signature, the means of these distributions overlap.

$$M(X) = M(Y) \rightarrow m_f = m \tag{3.3}$$

Therefore, the only difference between original and forged feature values is in their deviations. We define the quality of a forgery (q) as their quotient.

$$q = \frac{s_f}{s} \tag{3.4}$$

For the rest of this thesis, we are going to use the lower index o to annotate functions or variables belonging to original signatures and the lower index f for forged signatures.

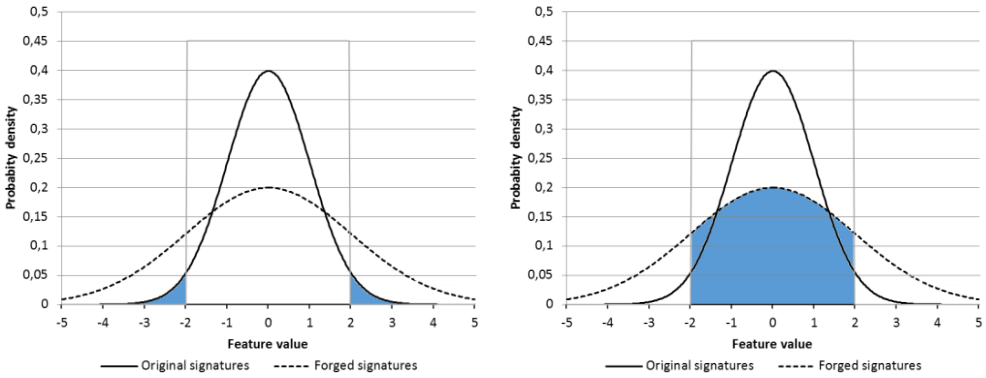
Remark: The first statement of Assumption 3.2 is a corollary of Assumption 3.1. As forgers aim to reproduce a given feature, the resulting properties will vary around a given mean. Although a single forger may make the same systematic error, when considering several possible forgers, these deviations should cancel themselves out resulting in the same expected value as in the feature properties of the original signatures (Eq. 3.3).

3.7 Experiments

We have also evaluated the assumptions on the BME-AUT2 database. The average deviation of expected values of forged feature properties was below 3% compared to the expected values of the originals. The average quotient of standard deviations (q) was 2.04.

3.8 Distinguishing original and forged feature properties

Let w be the threshold used to distinguish between original and forged feature properties using the following rule: each feature value that falls in the interval $[m - ws, m + ws]$ is accepted as an original, and each feature value which does not fall into the interval is taken as a forgery. This definition allows the unambiguous classification of a feature value as forged or original and introduces type I and type II errors.



False Rejection Rate (α) False Acceptance Rate (β)

Figure 3.6 The illustration of error rates

Error rates can be expressed based on the previous equations and w.

$$FRR = \alpha = P(m - ws > X) + P(m + ws < X) = 2 - 2\Phi(w) \quad (3.5)$$

$$FAR = \beta = P(m - ws < Y < m + ws) = 2\Phi\left(\frac{w}{q}\right) - 1 \quad (3.6)$$

Provided that the a priori probabilities of receiving an original sample and receiving a forged sample are equal, the average error of such a system would be:

$$AER = \frac{\alpha+\beta}{2} = \frac{1}{2} + \Phi\left(\frac{w}{q}\right) - \Phi(w) \quad (3.7)$$

Proposition 3.1

$$AER < \frac{1}{2}, \text{ when } q>1 \quad (3.8)$$

Proof:

Because Φ is monotonically increasing, and $\frac{w}{q} < w$, when $q > 1$

$$\Phi\left(\frac{w}{q}\right) < \Phi(w) \Rightarrow \Phi\left(\frac{w}{q}\right) - \Phi(w) < 0 \quad (3.9)$$

$$AER = \frac{1}{2} + \Phi\left(\frac{w}{q}\right) - \Phi(w) < \frac{1}{2} \quad (3.10)$$

We have shown that as long as the "quality" of forgeries is worse than that of original signatures, any threshold w will result in a better classification result than a simple coin toss. In

Proposition 3.2 we are going to show that this error rate can be minimized by the optimal choice of w.

Proposition 3.2

The sum of the probabilities of false rejection and false acceptance is minimized when

$$w = \pm\sqrt{2}\sqrt{\ln q}\sqrt{\frac{q^2}{q^2-1}} \tag{3.11}$$

Proof: The error is minimized if

$$\left(\frac{1}{2} + \Phi\left(\frac{w}{q}\right) - \Phi(w)\right)' = 0 \tag{3.12}$$

This is the case if

$$f_o(x) = f_f(x) \tag{3.13}$$

$$\frac{1}{\sqrt{2\pi s_0^2}} e^{-\frac{1}{2}\left(\frac{m-x}{s_o}\right)^2} = \frac{1}{\sqrt{2\pi s_f^2}} e^{-\frac{1}{2}\left(\frac{m-x}{s_f}\right)^2} \tag{3.14}$$

Using: $x = m + ws$ we get:

$$\frac{1}{s\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{m-(m+ws)}{s}\right)^2} = \frac{1}{qs\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{m-(m+ws)}{qs}\right)^2} \tag{3.15}$$

From which:

$$w = \pm\sqrt{2}\sqrt{\ln q}\sqrt{\frac{q^2}{q^2-1}} \tag{3.16}$$

The detailed proof can be found in Appendix A.1.

In spite of the slightly different notation, these results can also be derived from the one obtained in [64] for the "single feature" scenario.

The above equations reflect only a theoretical scenario where both the mean and variance are known. In practice, this is not the case, as only the *sample* mean and the *sample* variance are available. Because of the usually small sample size (typically $n \leq 10$) this has a direct effect on the optimal choice of w. In Proposition 3.3 we are going to calculate w by taking the sample size into consideration.

Proposition 3.3

Approximating the distributions with Student's t-distributions, the error is minimized if

$$w = \pm\sqrt{\left(\frac{q^2 - q^{2+\frac{2}{n}}}{q^{\frac{2}{n}} - q^2}\right)\left(\frac{n^2-1}{n}\right)} \tag{3.17}$$

Proof: Let $\hat{m}_n$ denote the sample mean and $\hat{s}_n^2$ denote the sample variance for n samples, where $n > 1$.

$$\hat{m}_n = \hat{m}_n(\mathrm{X}) = \frac{1}{n}\sum_{i=1}^{n} \mathrm{X_i} \tag{3.18}$$

$$\hat{s}_n^2 = \frac{1}{n-1}\sum_{i=1}^{n}(\mathrm{X}_i - \hat{m}_n)^2 \tag{3.19}$$

In this scenario, both the distributions of original and forged samples should be described with Student's t-distribution.

$$t(x) = \frac{1}{\hat{s}_n\sqrt{1+\frac{1}{n}}}\, t_{n-1}\left(\frac{x-\hat{m}_n}{\hat{s}_n\sqrt{1+\frac{1}{n}}}\right) \tag{3.20}$$

Where t_{n-1} stands for the probability density function of the t-distribution with $n-1$ degrees of freedom.

Let $t_o(x)$ and $t_f(x)$ denote the probability distribution function and $T_o(x)$ and $T_f(x)$ denote the cumulative distribution function of the original and forged samples, respectively.

In this case:

$$\alpha = P(m - ws > X) + P(m + ws < X) = 2\mathrm{T}_o(\hat{\mathrm{m}}_\mathrm{n} - w\hat{s}_n) \tag{3.21}$$

$$\beta = P(m - ws < X < m + w\sigma_o) = 1 - 2\mathrm{T}_\mathrm{f}\left(\hat{\mathrm{m}}_\mathrm{n} - w\frac{\hat{s}_n}{q}\right) \tag{3.22}$$

$$AER = \frac{\alpha+\beta}{2} = \frac{1}{2} + T_o(\hat{m}_n - w\hat{s}_n) - T_f\left(\hat{m}_n - w\frac{\hat{s}_n}{q}\right) \tag{3.23}$$

The error is minimized if

$$\left(\frac{1}{2} + T_o(\hat{m}_n - w\hat{s}_n) - T_f(\hat{m}_n - w\frac{\hat{s}_n}{q})\right)' = 0 \tag{3.24}$$

Expanding the equation we get:

$$\frac{1}{\hat{s}_n\sqrt{1+\frac{1}{n}}}\, t_{n-1}\left(\frac{w\hat{s}_n}{\hat{s}_n\sqrt{1+\frac{1}{n}}}\right) = \frac{1}{q\hat{s}_n\sqrt{1+\frac{1}{n}}}\, t_{n-1}\left(\frac{w\hat{s}_n}{q\hat{s}_n\sqrt{1+\frac{1}{n}}}\right) \tag{3.25}$$

From which:

$$w = \pm\sqrt{\left(\frac{q^2 - q^{2+\frac{2}{n}}}{q^{\frac{2}{n}} - q^2}\right)\left(\frac{n^2-1}{n}\right)} \tag{3.26}$$

The detailed proof can be found in Appendix A.2.

Equation 3.26 shows that the value of w is not independent of n. This is a refinement of the estimation from [64], where the number of samples was ignored. The effect of sample size (n) on the optimal threshold choice (w) is illustrated in Figure 3.7. As sample size increases, the decision threshold converges to the threshold defined in [64]; however, at low sample sizes there is a notable difference in the results. This difference gets even more significant as the quality of forgeries decreases (as q increases).

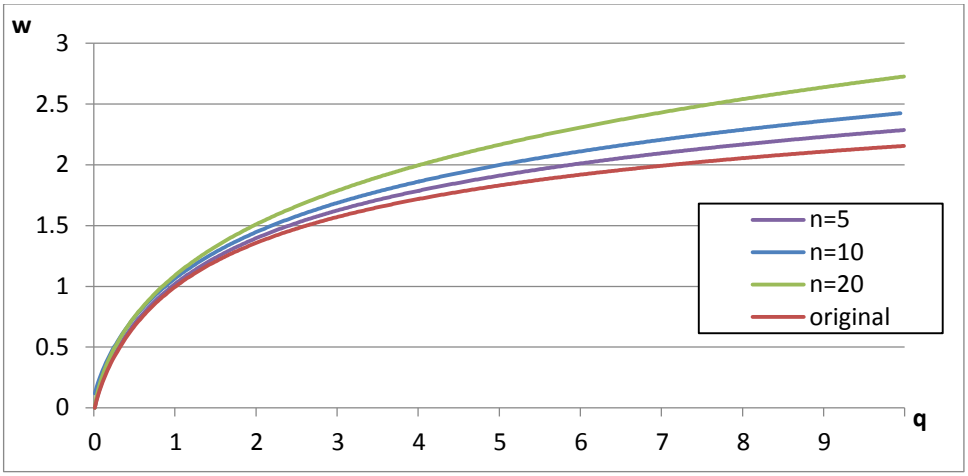


Figure 3.7 Optimal choices of decision threshold (w)
based on the quality of forgeries (q) and the number of samples (n)

Based on Eq. 3.26, Eq. 3.21 and 3.22 both α and β can be calculated. See Appendix A.3 for details.

3.9 Building a classifier

Equation 3.23 describes the accuracy of a decision that is based on a single property of a feature. However, in real world scenarios, we are able to utilize several different feature properties. In this section, we first analyze the accuracy of a classifier built on the above principles using a single feature property, and then extend the model for multiple feature properties.

3.9.1 Single feature property

Let the sample space $\Theta = \{\theta_o, \theta_f\}$ denote the genuineness of the questioned signature, where θ_f denotes the event when the signature is forged and θ_o denotes the event when a signature is original. We have no a priori knowledge about the signature; therefore, the a priori probabilities of the events are taken as equal:

$$P(\theta_o) = P(\theta_f) = 0.5 \tag{3.27}$$

Let X be an observation of a given feature property, where $X = 1$ means that the observed feature property fell in the interval $[m - ws, m + ws]$ and was therefore accepted. Similarly, $X = 0$ means rejection. The probabilities of these observations in the case of a genuine signature are:

$$P(X = 1|\theta_o) = 1 - \alpha \tag{3.28}$$

$$P(X = 0|\theta_o) = \alpha \tag{3.29}$$

while in the case of a forgery:

$$P(X = 1|\theta_f) = \beta \tag{3.30}$$

$$P(X = 0|\theta_f) = 1 - \beta \tag{3.31}$$

In the case of a single feature property and a single observation where $X = 0$ (rejection), the probability that the signature is really a forgery can be calculated as follows:

$$P(\theta_f|X = 0) = \frac{P(\theta_f)P(X=0|\theta_f)}{P(\theta_f)P(X=0|\theta_f)+P(\theta_o)P(X=0|\theta_o)} = \frac{1-\beta}{1-\beta+\alpha} \tag{3.32}$$

The error probability (FRR) can be calculated as:

$$P(\theta_o|X = 0) = \frac{\alpha}{1-\beta+\alpha} \tag{3.33}$$

3.9.2 Multiple feature properties

The accuracy of the final decision can be further improved by performing several tests where each test is performed on a different feature property. If $I \subseteq \{1 \dots k\}$ tests identify the sample as a forgery, and $\{1 \dots k\} \setminus I$ tests identify the sample as an original (where $I \in 2^k$), and assuming the independence of features, the probability of the signature being a forgery can be calculated as:

$$P(\theta_f|I) = \frac{\prod_{i\in I}(1-\beta_i)\prod_{j\notin I}\beta_j}{\prod_{i\in I}(1-\beta_i)\prod_{j\notin I}\beta_j+\prod_{i\in I}\alpha_i\prod_{j\notin I}(1-\alpha_j)} \tag{3.34}$$

Special case: If for all tests α_i-s are equal and β_i-s are equal, i.e., $\forall\alpha_i = \alpha$ and $\forall\beta_j = \beta$ and $l = |I|$, we obtain the following equations:

$$P(\theta_f|I) = \frac{(1-\beta)^l\beta^{k-l}}{(1-\beta)^l\beta^{k-l}+\alpha^l(1-\alpha)^{k-l}} \tag{3.35}$$

Similarly, the probability that the signature is original is:

$$P(\theta_o|I) = \frac{\alpha^l(1-\alpha)^{k-l}}{(1-\beta)^l\beta^{k-l}+\alpha^l(1-\alpha)^{k-l}} \tag{3.36}$$

For simplicity's sake, we use these simplified formulas for our classifications.

The above two equations allow us to make a decision that minimizes the average error. If $P(\theta_f|I) > P(\theta_o|I)$, the signature is classified as forged, and in the opposite case, it is classified as original. Note that α and β only depend on n and q. The figure below then shows how the value of l affects the certainty of the decision for a typical classification scenario ($n = 10$, $q = 2.0$, $k = 20$).

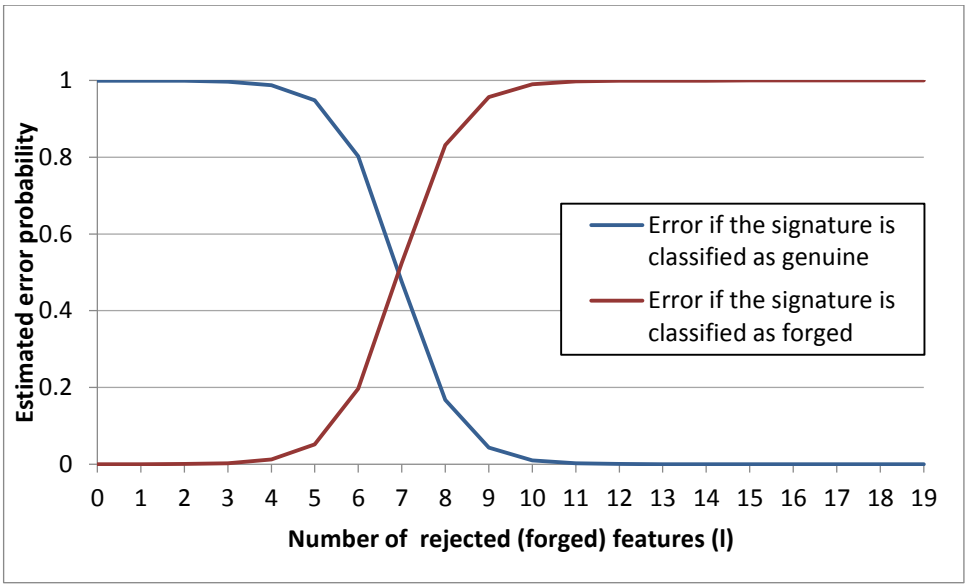


Figure 3.8 Probabilities of improper classification
based on the number of rejected feature properties (l)

Proposition 3.4: If each of the feature properties is either rejected or accepted, and α and β are known, the number of rejected features required to reject a signature while minimizing the average error rate is:

$$l = k\frac{\ln(1-\alpha)-\ln(\beta)}{\ln(1-\alpha)-\ln(\beta)+\ln(1-\beta)-\ln(\alpha)} \tag{3.37}$$

Proof: We can also find the point where

$$P(\theta_f|I) = P(\theta_o|I) \tag{3.38}$$

$$\frac{(1-\beta)^l\beta^{k-l}}{(1-\beta)^l\beta^{k-l}+\alpha^l(1-\alpha)^{k-l}} = \frac{\alpha^l(1-\alpha)^{k-l}}{(1-\beta)^l\beta^{k-l}+\alpha^l(1-\alpha)^{k-l}} \tag{3.39}$$

$$l = k\log_{\frac{(1-\alpha)(1-\beta)}{\alpha\beta}}\frac{1-\alpha}{\beta} = k\frac{\ln(1-\alpha)-\ln(\beta)}{\ln(1-\alpha)-\ln(\beta)+\ln(1-\beta)-\ln(\alpha)} \tag{3.40}$$

The detailed proof can be found in Appendix A.4

In this context, $\lfloor l \rfloor$ describes the maximum number of rejected features that can be allowed in an original signature. It should be noted that the certainty of the decision increases as l approaches k or 0. Figure 3.8 also shows that there is a greater uncertainty between 4 and 10 forged features. In a practical application this could be the interval of "inconclusive" decisions.

Using these results we can answer questions such as how many feature parameters must be classified as forged to make a decision about the origin of the signature with 99% certainty ($n = 10, q = 2.1, f = 20$).

Proposition 3.5: The lowest achievable average error rate of a signature verification system can be predicted based on the number of reference signatures used to train the system (n), the number of independent feature properties (k) and the quality of the forged features (q) as follows:

$$AER(q,n,k) = \frac{\sum_{j=0}^{\lfloor l \rfloor}\binom{k}{j}\beta^{k-j}(1-\beta)^{j} + \sum_{i=\lfloor l \rfloor+1}^{k}\binom{k}{i}\alpha^{i}(1-\alpha)^{k-i}}{2} \tag{3.41}$$

Proof: The total rates of false acceptance and rejection can be calculated from equations 3.35 and 3.36 by summing the error probabilities for all possible values of l:

$$FRR(q,n,k) = \sum_{j=\lfloor l \rfloor+1}^{k}\binom{k}{j}\alpha^{j}(1-\alpha)^{k-j} \tag{3.42}$$

$$FAR(q,n,k) = \sum_{i=0}^{\lfloor l \rfloor}\binom{k}{i}\beta^{k-i}(1-\beta)^{i} \tag{3.43}$$

$$AER(q,n,k) = \frac{FRR(q,n,k)+FAR(q,n,k)}{2} \tag{3.44}$$

The diagram below shows the above three error measures as a function of k, for $n = 5$, $q = 2.0$.

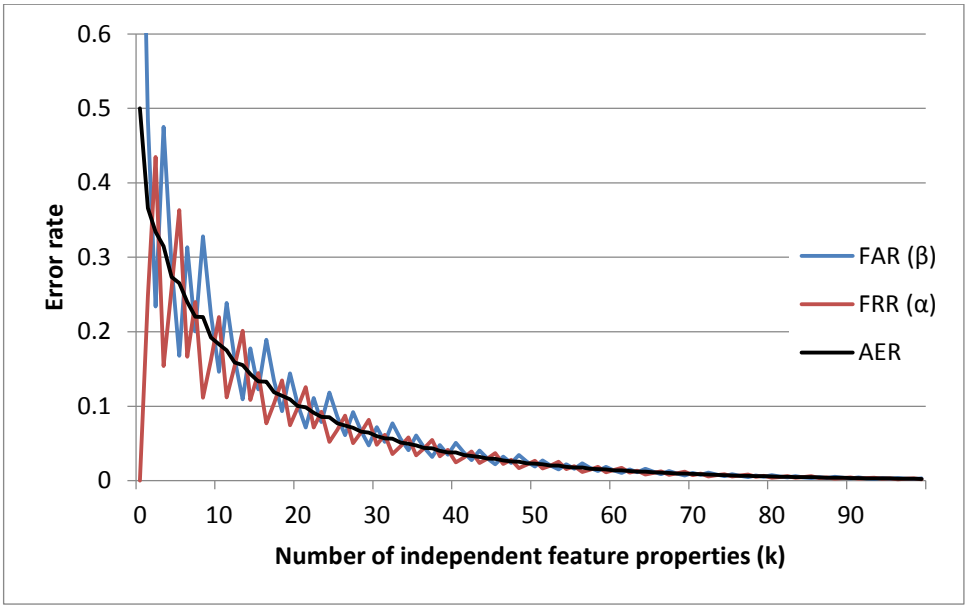


Figure 3.9 Predicted error rates

based on the number of examined independent feature properties (k)

Figure 3.9 also shows that (considering a fixed amount of original samples) at least 10 independent feature properties are required to achieve an error rate below 20%, 20 feature properties for 10% and further improvements require exponentially more properties, which may be why most verification systems fail to achieve error rates below 10%.

3.9.3 Experimental results

A classification module was created to test the precision of the original forgery decision, made only on the basis of our test database and using the previously introduced algorithms. Note that although several errors have been noted in the previous section, we did not alter our algorithms for this experiment.

The main purpose of the experiment is to show that our mathematical estimations can successfully predict the precision of a signature verifier. Therefore, several measurements were conducted with different original sample counts (n) and feature counts (k). During our experiments, the value for q was set to 1.6 for the SVC database and to 1.4 for the GPDS corpus, as the lower resolution and the higher complexity of signatures made the loop matching a harder task in the latter case.

Although the independence of features is a precondition of our theory, 100% independence can most likely never be ensured. To best match these preconditions, we have chosen (prior to the experiment) two global features, two attributes of loops and two attributes of baselines to approximate these conditions:

- aspect ratio (width/height) of the signature,
- moment axis of the signature,
- angle of the first baseline,
- length of the first baseline,
- angle of the second baseline,
- length of the second baseline,
- perimeter of the first loop (in pixels),
- angle of the moment axis of the first loop,
- perimeter of the second loop (in pixels), and
- angle of the moment axis of the second loop.

The number of baselines and loops taken into consideration was limited to two in each case. This limitation allowed us to compare the test results to our theoretical estimations, which assume a fixed number of feature properties.

The experimental setup takes n randomly chosen original signatures and k features from each signer. Afterwards, the classifier is tested with 10 more original and 10 forged signatures. The average error rate (AER) is then calculated for the whole database by averaging the error rates for each signer. The whole process is repeated 20 times (for different samples), and the results are averaged. To account for the differences of features, we evaluated the system for all possible feature combinations (all possible feature pairs, triplets, etc.) and again averaged the results. The final error rates are summarized in Table 3.1 and Table 3.2.

To allow comparisons, each table also shows the predicted error rates (b). It can be observed that the obtained results almost always fall behind the predicted

ones. This is normal and is a result of the errors in the matching and feature extraction algorithms. Additionally, as the number of features increases, the precision of our verifier falls further and further behind the expected results. This finding is also normal and is a result of the interdependency between features. However, even with these two anomalies, there is a clear correlation between expected and achieved results.

Table 3.1 Classification results on the SVC2004 database

a) Achived error rate (%)

reference signatures	number of features									
	1	2	3	4	5	6	7	8	9	10
2	44.6	43.0	41.8	40.9	40.1	39.4	38.8	38.3	38.0	36.8
3	43.2	41.0	39.4	38.0	37.0	36.2	35.5	34.9	34.4	33.2
4	42.7	39.9	37.9	36.4	35.1	34.0	33.2	32.4	31.7	30.9
5	42.2	39.2	36.9	35.2	33.8	32.6	31.7	31.0	30.2	29.4
6	41.9	38.6	36.2	34.4	33.0	31.7	30.7	29.8	29.2	27.8
7	41.9	38.3	35.8	33.8	32.3	31.1	30.1	29.1	28.4	27.9
8	41.5	37.8	35.1	33.1	31.5	30.2	29.2	28.3	27.5	27.7
9	41.3	37.6	34.9	32.9	31.2	29.9	28.8	27.9	27.0	26.6
10	41.3	37.6	34.9	32.8	31.3	29.9	28.9	27.9	27.0	26.8

b) Predicted error rate (%)

reference signatures	number of features									
	1	2	3	4	5	6	7	8	9	10
2	42.5	42.5	38.9	38.9	36.2	36.2	34.1	34.0	32.2	32.2
3	41.1	39.7	37.0	35.1	34.3	32.1	31.9	29.8	29.1	28.0
4	40.4	38.4	36.4	33.7	33.4	30.8	29.6	28.8	27.0	26.8
5	40.1	37.6	36.2	32.9	32.2	30.2	28.5	28.5	26.2	25.3
6	39.8	37.2	36.0	32.4	31.5	29.9	27.9	27.5	25.7	24.5
7	39.7	36.8	35.9	32.1	31.0	29.7	27.5	26.9	25.5	24.0
8	39.6	36.6	35.9	31.9	30.6	29.6	27.2	26.4	25.3	23.6
9	39.5	36.4	35.8	31.8	30.3	29.5	27.0	26.1	25.2	23.4
10	39.4	36.2	35.8	31.7	30.1	29.5	26.8	25.8	25.1	23.2

c) Differences between a) and b)

reference signatures	number of features									
	1	2	3	4	5	6	7	8	9	10
2	2.1	0.5	2.9	2.0	3.9	3.2	4.7	4.3	5.8	4.6
3	2.1	1.3	2.4	2.9	2.7	4.1	3.6	5.1	5.3	5.2
4	2.3	1.5	1.5	2.7	1.7	3.2	3.6	3.6	4.7	4.1
5	2.1	1.6	0.7	2.3	1.6	2.4	3.2	2.5	4.0	4.1
6	2.1	1.4	0.2	2.0	1.5	1.8	2.8	2.3	3.5	3.3
7	2.2	1.5	-0.1	1.7	1.3	1.4	2.6	2.2	2.9	3.9
8	1.9	1.2	-0.8	1.2	0.9	0.6	2.0	1.9	2.2	4.1
9	1.8	1.2	-0.9	1.1	0.9	0.4	1.8	1.8	1.8	3.2
10	1.9	1.4	-0.9	1.1	1.2	0.4	2.1	2.1	1.9	3.6

Table 3.2 Classification results on the GPDS300 database

a) Achived error rate (%)

reference signatures	number of features									
	1	2	3	4	5	6	7	8	9	10
2	44.8	43.8	43.1	42.5	41.9	41.3	40.8	40.5	39.9	39.0
3	43.6	42.0	41.3	40.4	39.6	38.9	38.4	37.9	37.3	37.4
4	43.2	41.2	40.2	39.2	38.3	37.6	36.9	36.5	36.0	36.3
5	43.0	40.7	39.4	38.4	37.4	36.7	36.2	35.6	35.3	34.4
6	42.7	40.4	39.0	37.9	36.8	36.0	35.5	35.0	34.4	33.8
7	42.7	40.0	38.7	37.4	36.4	35.5	34.9	34.5	33.9	33.4
8	42.5	40.0	38.4	37.1	36.0	35.1	34.5	34.0	33.6	33.1
9	42.3	39.7	38.1	36.8	35.7	34.7	34.1	33.6	33.2	33.1
10	42.1	39.5	37.9	36.5	35.4	34.5	33.7	33.2	32.9	32.2

b) Predicted error rate (%)

reference signatures	number of features									
	1	2	3	4	5	6	7	8	9	10
2	44.6	44.6	42.0	42.0	40.0	40.0	38.4	38.4	37.0	37.0
3	43.6	42.6	40.6	39.2	38.6	36.9	36.8	35.2	34.6	33.8
4	43.1	41.6	40.2	38.1	37.9	35.9	35.0	34.4	33.0	32.8
5	42.8	41.1	40.0	37.5	37.0	35.5	34.2	34.2	32.4	31.7
6	42.7	40.7	39.9	37.2	36.5	35.3	33.7	33.4	32.0	31.0
7	42.6	40.4	39.8	37.0	36.1	35.1	33.4	32.9	31.8	30.6
8	42.5	40.3	39.8	36.8	35.8	35.1	33.1	32.6	31.7	30.3
9	42.4	40.1	39.7	36.7	35.6	35.0	33.0	32.3	31.6	30.1
10	42.4	40.0	39.7	36.6	35.4	35.0	32.9	32.1	31.5	30.0

c) Differences between a) and b)

reference signatures	number of fatures									
	1	2	3	4	5	6	7	8	9	10
2	0.1	-0.9	1.2	0.5	1.9	1.3	2.4	2.1	2.9	2.1
3	0.1	-0.6	0.6	1.2	1.0	2.0	1.7	2.7	2.7	3.6
4	0.1	-0.4	0.0	1.1	0.3	1.7	1.9	2.1	3.0	3.5
5	0.1	-0.4	-0.5	0.9	0.4	1.2	2.0	1.4	2.9	2.7
6	0.0	-0.3	-0.8	0.7	0.4	0.7	1.8	1.6	2.4	2.8
7	0.1	-0.4	-1.1	0.5	0.3	0.4	1.6	1.6	2.1	2.8
8	0.0	-0.3	-1.3	0.3	0.2	0.1	1.3	1.4	1.9	2.7
9	-0.1	-0.4	-1.6	0.1	0.1	-0.2	1.1	1.3	1.6	2.9
10	-0.2	-0.5	-1.8	-0.1	0.0	-0.5	0.9	1.1	1.3	2.2

3.10 Chapter summary

An algorithmic background for feature-based off-line signature verification has been contributed in this chapter. It was shown that the distribution of several feature values can be reasonably well approximated with normal distribution. The intuitive explanation was confirmed by our experimental results. The statement has been also extended to forged feature values, which follow the same distribution but with (usually) greater variance.

Using the previously formulated assumptions, it became possible to describe the total error rate of a threshold classifier working on a single feature (Proposition 3.1). Moreover, in

Proposition 3.2 we have shown that it is possible to select a threshold, which also minimizes the total error rate. As signature verification usually has to deal with a low number of reference samples, the sample count has been also incorporated in our calculations by using Student's t-distribution to approximate the distribution of feature values (Proposition 3.3).

Next, we extended our model to take the number of features into consideration. Provided that each of the features has a binary vote about the origin of a signature, we have shown the minimal number of features that can be classified as forged before the whole signature should be classified as a forgery (Proposition 3.4) and calculated the resulting error rate of a system (

Proposition 3.5).

To test the validity of our model, a complex experiment was introduced in Section 3.9.3. The experiment has shown that our analytical estimations were in good accordance with the measured results, although – because of the interdependency of feature values – these results tend to diverge when working with 10 or more features.

The main benefit of the introduced model is that – unlike the artificial intelligence-based approaches – it can be analytically described, providing a deep insight into the decision mechanisms. To demonstrate this, we are going to examine 3 major properties of our model in the next sections.

4. The implications of the statistical model

Based on the results of the previous chapter, we are now able to predict the accuracy of a signature verification system based on only 3 parameters: the number of original signatures used to train the system (n), the number of independent features (k) and the quality of the forged features (q). Moreover, the first two can be directly controlled by the experimental setup. Equation 3.44 allows us to model the effects of any of the parameters on each other. This yields several practical applications, some of which are shown in the next subsections.

4.1 The effect of feature count (k) on AER.

In the next proposition, we are going to prove that increasing the number of features will, almost always decrease and never increase the average error rate.

4.1.1 Proposition

Proposition 4.1

Given the constraints

$$q > 1, k \geq 1, n > 2, n, k \in \mathbb{Z}, q \in \mathbb{R} \tag{4.1}$$

and provided that the number of reference samples (n_0) and the forgery quality (q_0) is given, the function $AER(q_0, n_0, k)$ is decreasing.

Proof:

We have to prove that

$$AER(q_0, n_0, k) \geq AER(q_0, n_0, k+1) \tag{4.2}$$

As the parameters q_0 and n_0 are fixed, they will be omitted during this proof. As α and β are only dependent on n and q they can be regarded as fixed numbers.

We should note some of the main properties of the arguments, which will be used in the following proofs

$$0 < \alpha < 1 \tag{4.3}$$

$$0 < \beta < 1 \tag{4.4}$$

$$\alpha < \beta \tag{4.5}$$

$$\frac{\alpha+\beta}{2} < 0.5 \tag{4.6}$$

$$1 < \frac{1-\beta}{\alpha} \tag{4.7}$$

$$1 < \frac{1-\alpha}{\beta} \tag{4.8}$$

Equations 4.3, 4.4 and 4.5 are trivial, equation 4.6 is a direct consequence of 3.10 and equations 4.7 and 4.8 are transformations of equation 4.6

Lemma 4.1

If

$$l = k \log_{\frac{(1-\alpha)(1-\beta)}{\alpha\beta}} \frac{1-\alpha}{\beta} \tag{4.9}$$

then

$$\left\lfloor k \log_{\frac{(1-\alpha)(1-\beta)}{\alpha\beta}} \frac{1-\alpha}{\beta} \right\rfloor = \lfloor l \rfloor \tag{4.10}$$

and

$$\left\lfloor (k+1) \log_{\frac{(1-\alpha)(1-\beta)}{\alpha\beta}} \frac{1-\alpha}{\beta} \right\rfloor = \begin{cases} \lfloor l \rfloor \\ \lfloor l \rfloor + 1 \end{cases} \tag{4.11}$$

Proof:

Equation 4.10 is direct consequence of equation 4.9. To prove 4.11, we have to prove, that

$$\log_{\frac{(1-\alpha)(1-\beta)}{\alpha\beta}} \frac{1-\alpha}{\beta} < 1 \tag{4.12}$$

This is always true as both the argument (equation 4.8) and the base (equations 4.7 and 4.8) are greater than 1 and the base is greater than the argument, and all of them are positive.

Because of Lemma 4.1, the proof must be divided into two separate cases.

4.1.2 Case 1

$$\left\lfloor (k+1) \log_{\frac{(1-\alpha)(1-\beta)}{\alpha\beta}} \frac{1-\alpha}{\beta} \right\rfloor = \lfloor l \rfloor \tag{4.13}$$

In this case

$$AER(k) = \frac{FAR(k)+FRR(k)}{2} \tag{4.14}$$

$$AER(k+1) = \frac{FAR(k+1)+FRR(k+1)}{2} \tag{4.15}$$

We are going to prove that

$$AER(k) > AER(k+1) \tag{4.16}$$

$$\frac{\frac{\sum_{j=0}^{\lfloor l \rfloor}\binom{k}{j}\beta^{k-j}(1-\beta)^j+\sum_{i=\lfloor l \rfloor+1}^{k}\binom{k}{i}\alpha^i(1-\alpha)^{k-i}}{2}}{\frac{\sum_{j=0}^{\lfloor l \rfloor}\binom{k+1}{j}\beta^{k+1-j}(1-\beta)^j+\sum_{i=\lfloor l \rfloor+1}^{k+1}\binom{k+1}{i}\alpha^i(1-\alpha)^{k+1-i}}{2}} > \tag{4.17}$$

After expanding and performing the summations (see Appendix A.5 for details)

$$\frac{1-\beta}{\alpha}\left(\frac{1-\beta}{\alpha}\right)^{\lfloor l \rfloor}\left(\frac{1-\alpha}{\beta}\right)^{\lfloor l \rfloor} > \frac{(1-\alpha)^k}{\beta^k} \tag{4.18}$$

From equation 3.39

$$\left(\frac{1-\beta}{\alpha}\right)^{\lfloor l \rfloor}\left(\frac{1-\beta}{\alpha}\right)^{\{l\}}\left(\frac{1-\alpha}{\beta}\right)^{\lfloor l \rfloor}\left(\frac{1-\alpha}{\beta}\right)^{\{l\}} = \frac{(1-\alpha)^k}{\beta^k} \tag{4.19}$$

Combining the previous equations, we get:

$$\frac{1-\beta}{\alpha}\left(\frac{1-\beta}{\alpha}\right)^{\lfloor l \rfloor}\left(\frac{1-\alpha}{\beta}\right)^{\lfloor l \rfloor} > \left(\frac{1-\beta}{\alpha}\right)^{\lfloor l \rfloor}\left(\frac{1-\beta}{\alpha}\right)^{\{l\}}\left(\frac{1-\alpha}{\beta}\right)^{\lfloor l \rfloor}\left(\frac{1-\alpha}{\beta}\right)^{\{l\}} \tag{4.20}$$

$$\frac{1-\beta}{\alpha} > \left(\frac{1-\beta}{\alpha}\right)^{\{l\}}\left(\frac{1-\alpha}{\beta}\right)^{\{l\}} \tag{4.21}$$

We also know from equation 4.13 that

$$\{l\} + \log_{\frac{(1-\alpha)(1-\beta)}{\alpha\beta}}\frac{1-\alpha}{\beta} < 1 \tag{4.22}$$

therefore

$$\left(\frac{1-\beta}{\alpha}\right)^{\{l\}+\log_{\frac{(1-\alpha)(1-\beta)}{\alpha\beta}}\frac{1-\alpha}{\beta}}\left(\frac{1-\alpha}{\beta}\right)^{\{l\}+\log_{\frac{(1-\alpha)(1-\beta)}{\alpha\beta}}\frac{1-\alpha}{\beta}} < \left(\frac{1-\beta}{\alpha}\right)^{1}\left(\frac{1-\alpha}{\beta}\right)^{1} \tag{4.23}$$

from which

$$\frac{1-\beta}{\alpha} > \left(\frac{1-\beta}{\alpha}\right)^{\{l\}}\left(\frac{1-\alpha}{\beta}\right)^{\{l\}} \tag{4.24}$$

The constraint ensured by equation 4.24 supports equation 4.21, which was derived from our original statement with equivalent transformations. Therefore, the original statement is true.

4.1.3 Case 2

$$\left\lfloor (k+1)\log_{\frac{(1-\alpha)(1-\beta)}{\alpha\beta}}\frac{1-\alpha}{\beta}\right\rfloor = \lfloor l \rfloor + 1 \tag{4.25}$$

The proof is mainly analogous to case 1. Only the main differences are highlighted below, while the details are provided in Appendix A.5.

We are going to prove that

$$AER(k) \geq AER(k+1) \tag{4.26}$$

After expanding and performing the summations:

$$\frac{1-\beta}{\alpha}\left(\frac{1-\beta}{\alpha}\right)^{\lfloor l \rfloor}\left(\frac{1-\alpha}{\beta}\right)^{\lfloor l \rfloor} \leq \frac{(1-\alpha)^k}{\beta^k} \tag{4.27}$$

Combining the above with equation 4.19 we get:

$$\frac{1-\beta}{\alpha} \leq \left(\frac{1-\beta}{\alpha}\right)^{\{l\}}\left(\frac{1-\alpha}{\beta}\right)^{\{l\}} \tag{4.28}$$

We also know from equation 4.25 that

$$\{l\} + \log_{\frac{(1-\alpha)(1-\beta)}{\alpha\beta}} \frac{1-\alpha}{\beta} \geq 1 \tag{4.29}$$

therefore

$$\left(\frac{1-\beta}{\alpha}\right)^{\{l\}+\log_{\frac{(1-\alpha)(1-\beta)}{\alpha\beta}}\frac{1-\alpha}{\beta}}\left(\frac{1-\alpha}{\beta}\right)^{\{l\}+\log_{\frac{(1-\alpha)(1-\beta)}{\alpha\beta}}\frac{1-\alpha}{\beta}} \geq \left(\frac{1-\beta}{\alpha}\right)^{1}\left(\frac{1-\alpha}{\beta}\right)^{1} \tag{4.30}$$

from which

$$\frac{1-\beta}{\alpha} \leq \left(\frac{1-\beta}{\alpha}\right)^{\{l\}}\left(\frac{1-\alpha}{\beta}\right)^{\{l\}} \tag{4.31}$$

The constraint ensured by equation 4.31 supports equation 4.28, which was derived from our original statement with equivalent transformations, therefore the original statement is true.

Figure 4.3 provides a visual representation of the results (for $n = 2$). A cross section for $q = 2$ can be also seen in Figure 3.9.

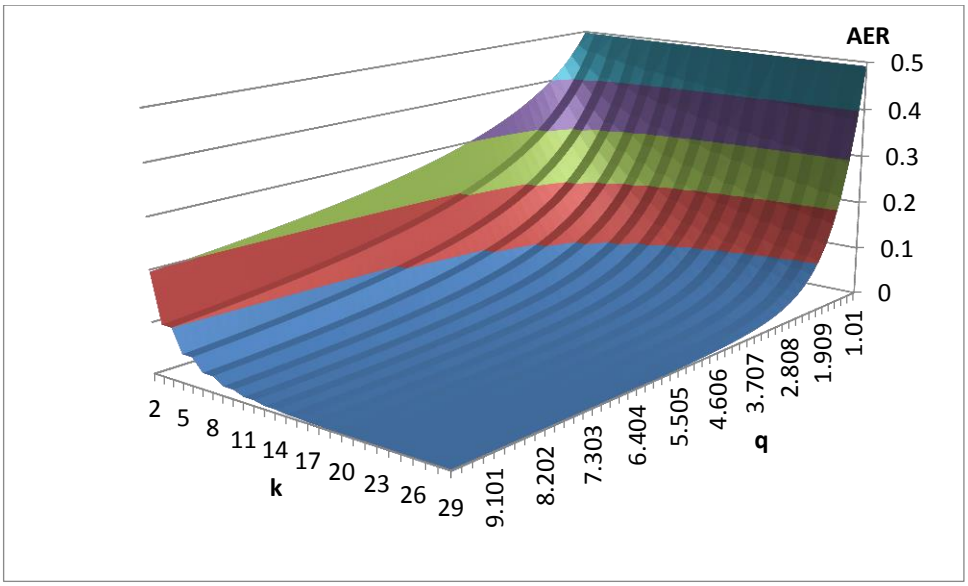


Figure 4.1 The relation of AER to the value of k at a given forgery quality (q)

Remark 1: Although we allowed for equality in Equation 4.2, this equality arises only in the extremely unlikely case when the left side of Equation 4.29 equals 1. Thus it is safe to say that increasing the feature count will lead to better classification results.

4.2 The effect of sample count (n) on AER

In the next proposition we are going to show that increasing the number of reference signatures will always decrease the average error rate.

Proposition 4.2

Given the constraints

$$q > 1, k = 1, n > 2, n, k \in \mathbb{Z}, q \in \mathbb{R} \tag{4.32}$$

and provided that the number of observed features (k_0) and forgery quality (q_0) is given, the function $AER(q_0, n, k_0)$ is strictly decreasing.

Lemma 4.2

As the number of degrees of freedom grows, t-distribution approaches normal distribution. Therefore, as the number of samples increases, both the distributions of original and forged samples will converge to normal distribution, resulting in a total error rate converging towards the average error of normal distributions, which was described in Equation 3.7:

$$\lim_{n\to\infty} AER(q, n, 1) = \frac{1}{2} + \Phi\left(\frac{w}{q}\right) - \Phi(w) \tag{4.33}$$

Proof:

Using analytical tools [68], we calculated the partial derivative $\frac{\partial AER(q,n,1)}{\partial n}$ (see Appendix 0 for details). Our simulation (Figure 4.2) indicates that this derivative is negative for the relevant input range ($n \in [2;80], q \in (1,10]$) and it suggests that this is also the case for higher values of n. To support the latter assumption, we will show that the function value converges towards zero.

Lemma 4.2 shows that AER converges to $\frac{1}{2} + \Phi\left(\frac{w}{q}\right) - \Phi(w)$, which is not dependent on n, therefore $\frac{\partial AER(q,n,1)}{\partial n}$ will converge towards 0.

$$\lim_{n\to\infty} \frac{\partial AER(q,n,1)}{\partial n} = 0 \tag{4.34}$$

Since the derivative of the function has a limit of zero and based on the fact that the graph has approached it significantly from below, we can assume that it will not obtain positive values outside the plotted range.

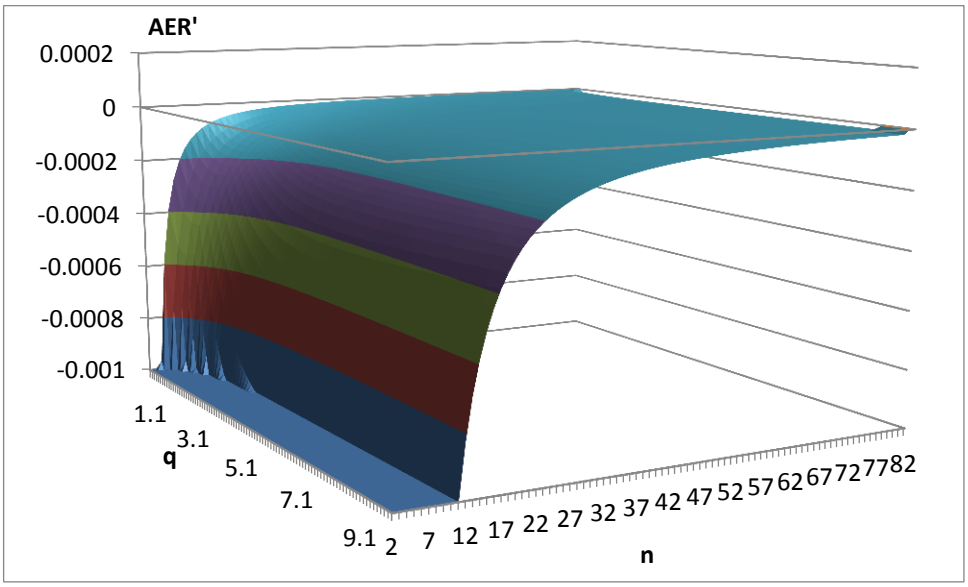


Figure 4.2 The function $\frac{\partial AER(q,n,k)}{\partial n}$

values below -0.001 were removed.

Figure 4.3 illustrates that an increase of n will always result in a lower error rate.

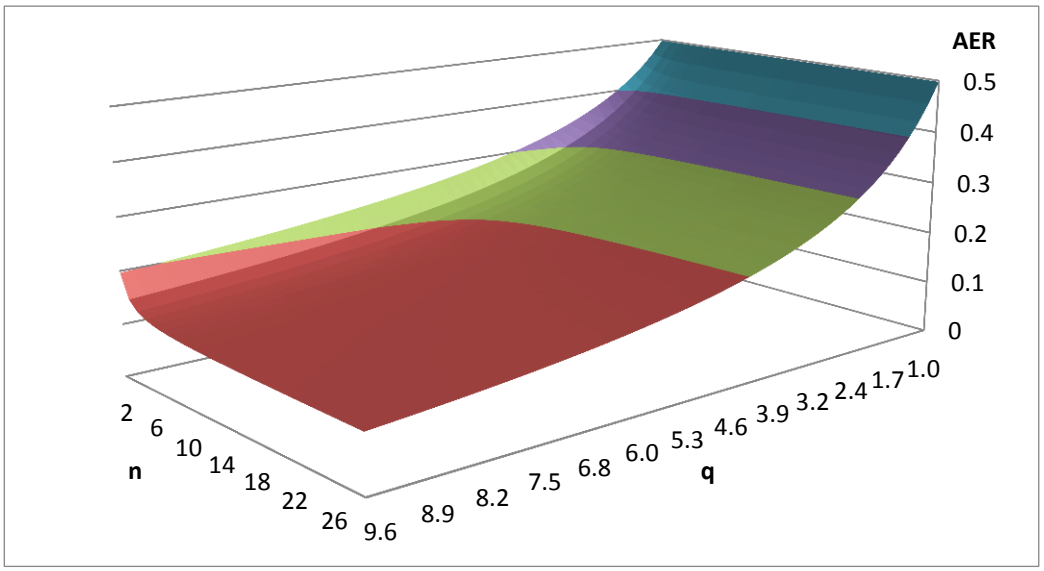


Figure 4.3 Relation of AER to the value of n at given forgery quality (q)

Remark 1: The statement can be generalized for $k \geq 1$ based on Proposition 4.1.

Remark 2: The improvement in error rates decreases rapidly and is almost insignificant when $n > 10$. In conclusion: increasing the sample sizes above 10 has a negligible effect on practical application.

4.3 The effect of threshold (w) on EER

In the previous sections we have seen that the value of q significantly influences the accuracy of the system. Figure 4.4 shows that although this influence is important, there is usually a wide range for the threshold (w) where the differences only play a minor effect on the final classification results. This section is devoted to formulating a conjecture about this property and providing experimental results to verify it.

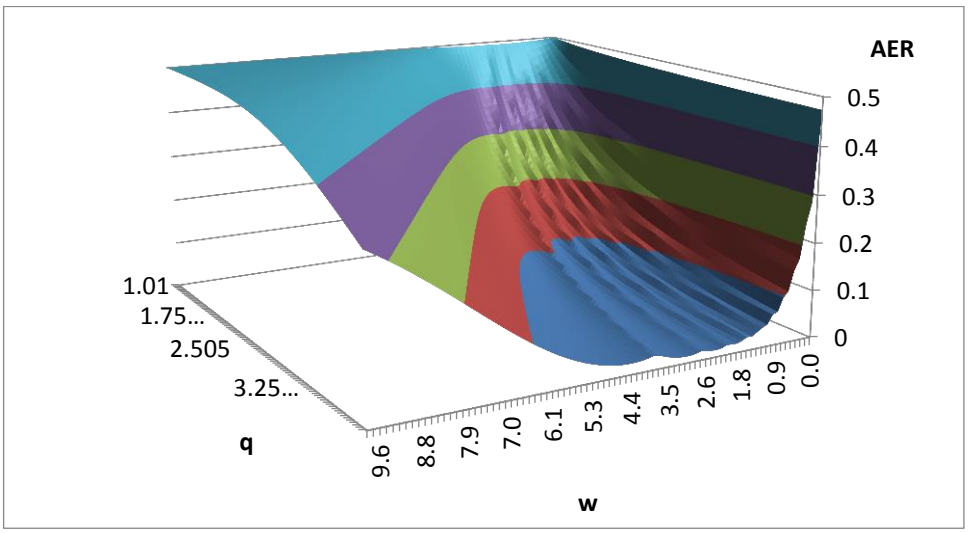


Figure 4.4 The relationship of q, w and AER when $n = 10$ and $k = 10$

Figure 4.5 shows some cross-sections of Figure 4.4 at given forgery qualities (q). At $q = 1.01$, the AER is close to 0.5 because there is almost no way to distinguish between originals and forgeries. By increasing the value of q, the achievable error rate becomes lower. The graph also shows that when $q \approx 4$, the value of w may fall anywhere between 2 and 3.5 without major effects on AER.

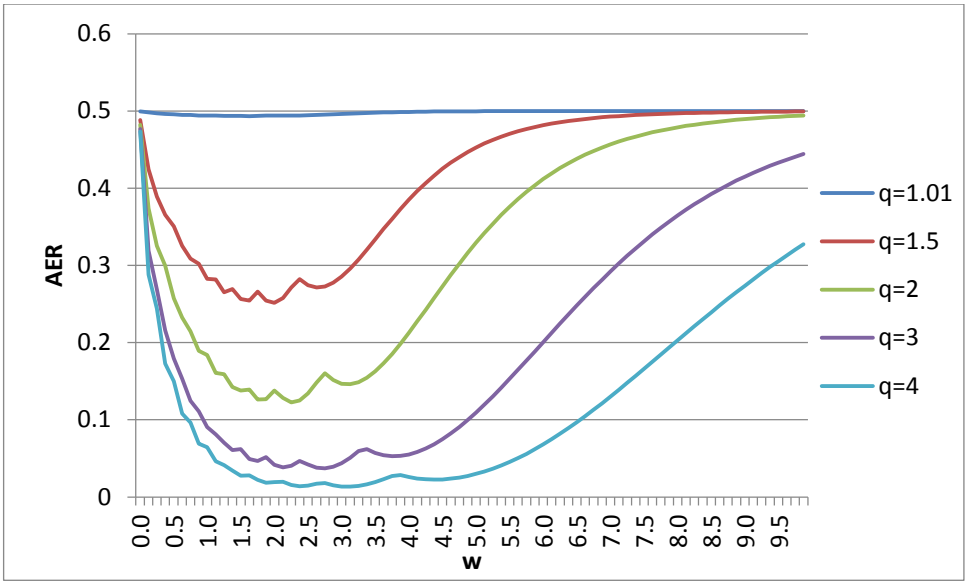


Figure 4.5 Relation of AER to the value of w, at given forgery qualities (q)

Of course, in our application, w is set by our system (equation 3.17) depending on a previously estimated value of q. Figure 4.6 shows the resulting increases of error rates resulting from misestimating q.

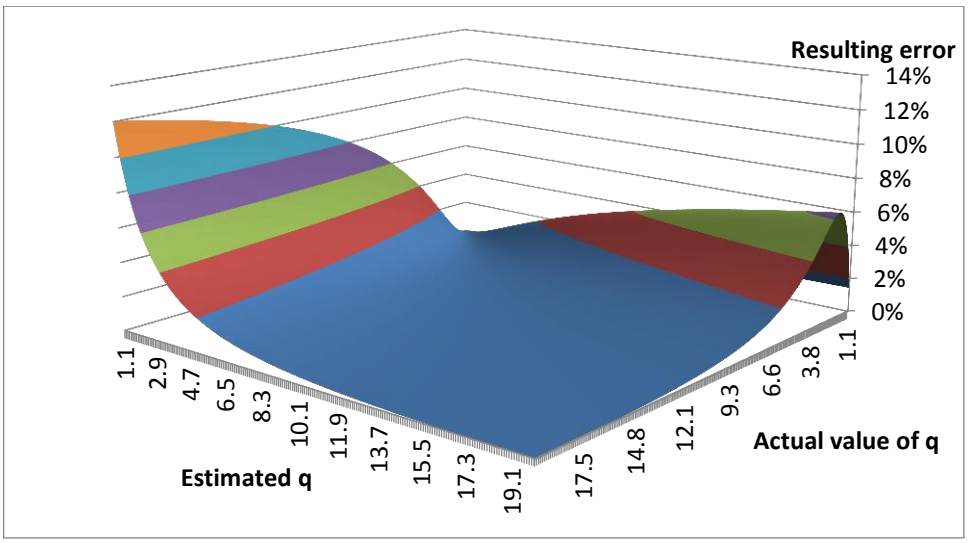


Figure 4.6 The resulting errors

based on the relation of estimated and actual forgery quality (q)

The wide “plane” on Figure 4.6 is the base of our next proposition.

Conjecture 4.1

Supposed that the value of the actual forgery quality (q_{act}) falls in the interval $(1; q_{max})$, there is an ideal value for estimated forgery quality (q_{est}) that minimizes the errors resulting from a suboptimal choice of w.

Experimental results

Figure 4.7 shows that there is a clear choice for q that can minimize the total error, provided that the maximal value of q is known. Figure 4.8 shows these optimal choices, while Figure 4.9 shows the resulting worst-case scenario for an increase in classification errors.

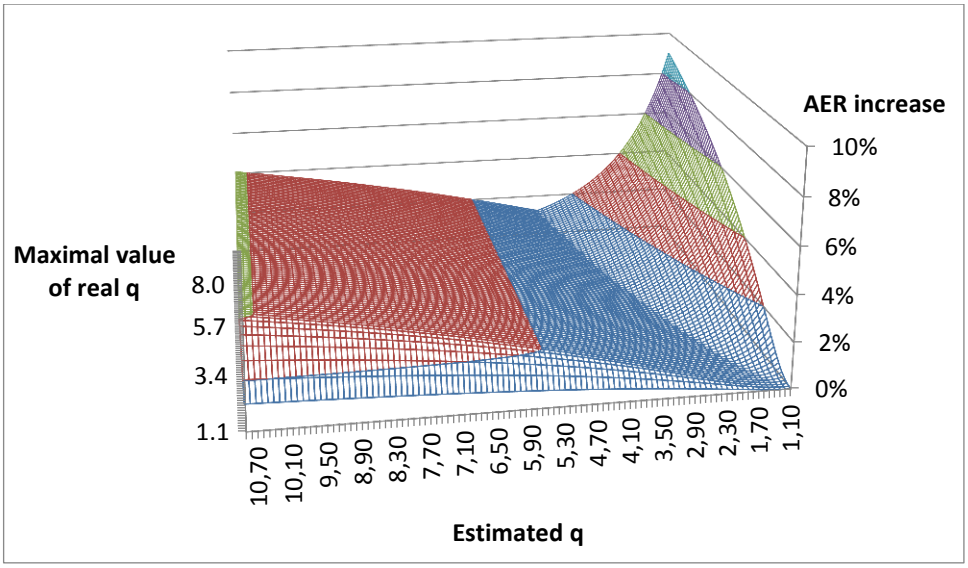


Figure 4.7 The resulting increase in error
depending on the maximal value (q_{max}) and the estimated value (q_{est}) of forgery quality

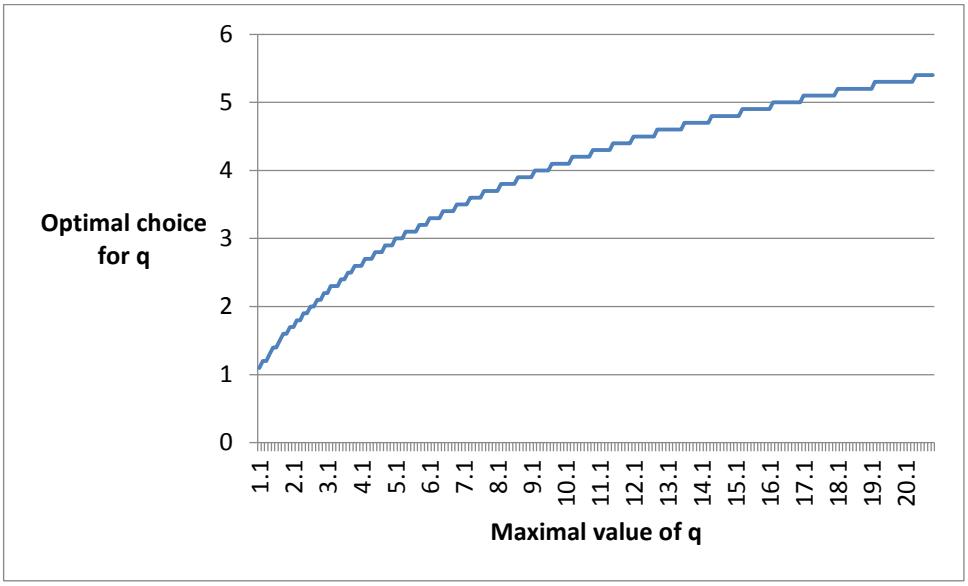


Figure 4.8 The optimal choices for q, depending on the maximal value of actual q

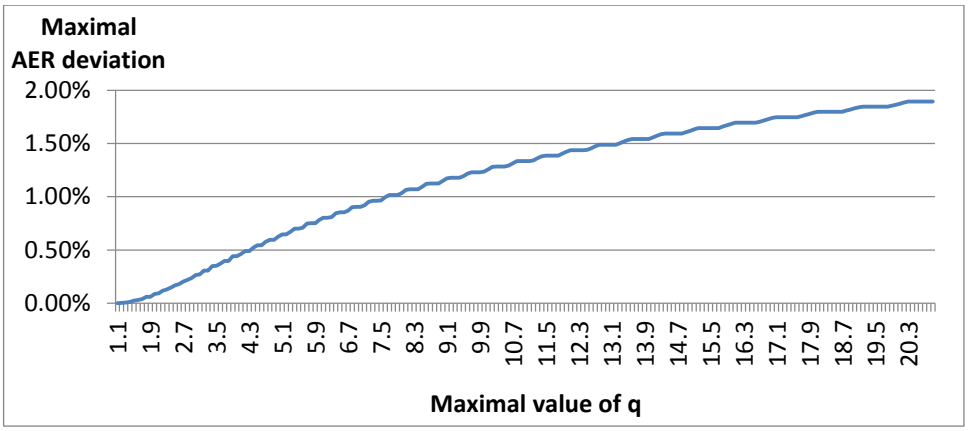


Figure 4.9 The maximal increase in error rate
when choosing the optimal value of q for different maximal values.

4.4 Chapter summary

In this chapter we have introduced three important characteristics of our statistical model.

In Proposition 4.1 it was shown that the average error rate of the system decreases as the number of features increases. This relation is inversely proportional to q. While at $q = 1.01$, this decrease is very slow, at $q = 10$ even 3 feature values are enough to lower the expected error rate to below 10%.

Proposition 4.2 has shown a similar relation between the number of reference signatures and the average error rate. Although the statements of the previous propositions may seem trivial at first glance, even the fact that they could be proven shows that our statistical classifier exposes a unique analytical potential that most other classifiers lack.

Conjecture 4.1 provides a deeper insight into our model by examining possible error increases that arise when the quality of forgeries was misestimated. Our simulations show that this kind of error can be greatly reduced by the optimal choice of parameter q. For example, when the actual value of q is somewhere between 1 and 20 and we estimate it as $q = 5.3$, then the increase of AER will stay below 2% even in the worst possible case.

Last, but not least, all of the previously introduced calculations are highly dependent on the accuracy of previous processing steps such as feature extraction and feature matching. Therefore, the next chapter is devoted to the introduction of a novel stroke extraction algorithm and the main processing algorithms used to provide the input for our classification system.

5. Signature processing algorithms

The previous two chapters have shown how feature values can be used in the classification phase of signature verification. In order to achieve these results, the features first had to be located and assigned to each other in various processing steps. Moreover, the accuracy of these steps highly influences the achievable error rates of the classifier.

This section describes a collection of processing algorithms used in our solution. First, a brief overview of thinning approaches is given and a new thinning algorithm is proposed for the extraction of stroke information from signature images. The remainder of the section describes the matching and feature extraction algorithms used to extract and match loops and baselines in our system.

5.1 Line thinning

The input of off-line signature verifiers consists of scanned images of hand-written signatures that usually cannot be processed in their original form by some modules of the verification system. The purpose of thinning in the preprocessing phase is to clean the image of any kind of noise caused by scanning and to extract a binary, one pixel wide skeleton from the signature. Optimally, this discrete skeleton is a compact representation of the original signature that minimizes the loss of semantic information; therefore, it can be efficiently interpreted by the other modules of the process. Thinned images make several later steps – like the extraction of endpoints in Figure 5.1 – much simpler (and thereby faster) by reducing the amount of information to process.

After giving an overview of the related work and currently applied methods, we will propose a novel thinning algorithm. Experimental validation of the algorithm is presented and evaluated. This algorithm could be used to refine existing signature verification methods and also to create new ones [69].

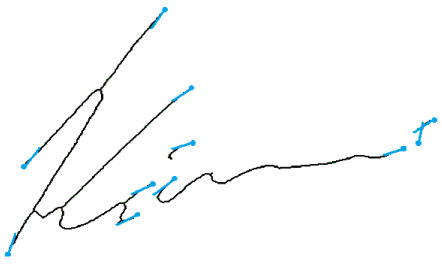


Figure 5.1 Endpoint extraction on a thinned image

5.1.1 The classification of thinning algorithms

Mainstream methodologies have gone through plenty of changes and improvement in the last decades. They can be categorized as follows:

5.1.1.1 Morphology-based approaches

The first attempt at extracting skeletons from binary images was using the operators of mathematical morphology (MM), widely used at the time for texture analysis.

The basic idea in binary morphology [70] is to probe an image with a simple, pre-defined shape (called the "probe" or "kernel"), then to draw conclusions based on how this shape fits or misses the shapes in the image. For thinning, the most important operation of MM is erosion, which basically erodes the boundaries of the region in the foreground. Thus areas of foreground pixels shrink in size and holes within those areas become larger.

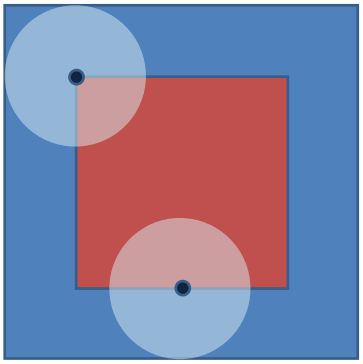


Figure 5.2 The erosion of the blue square by a disk, resulting in the red square.

Erosion in itself does not produce appropriate skeletons, but its logic is clearly present in more complex algorithms, especially that of windowing.

5.1.1.2 Raster scanning based approaches

The execution of raster scan algorithms usually goes as follows [71]:

1. All the pixels of a binary image are examined in a predetermined order[3]
2. A set of conditions is evaluated for each pixel, either marking it for deletion or for retention
3. At the end of the iteration, all the pixels marked for deletion are erased from the image and the process jumps back to step 1.

If no pixel was deleted in the iteration, the thinning process is done and the remaining black pixels ought to form the skeleton of the image.

Raster scan algorithms differ from each other in the set of conditions evaluated to decide whether to delete or retain a particular pixel. Different condition sets preserve different characteristics of the input image.

[3] In some cases, only the borders of connected objects are examined (contour following).

5.1.1.3 Windowing-based approaches

A specialized version of the hit-and-miss morphological operator is used in numerous thinning algorithms. It is usually referred to as windowing or masking.

A thinning mask is an arbitrary-sized – though usually small – grid in which each cell denotes a custom condition. The mask is fitted onto a region of the tested image, and then the conditions of the cells are evaluated for the pixel under it. We say that the mask fits onto the region of the image if every pixel under the mask satisfies the corresponding condition.

This technique is used in raster scan algorithms where the conditions evaluated for each pixel are given in the form of a thinning mask [72].

A	A	A
0	P	0
B	B	B

Figure 5.3 Example of a commonly used thinning mask

In Figure 5.3, an example of thinning masks can be seen where the meaning of the mask cells are the follows:

- Pixels marked with 0 have to belong to the background (white)
- The pixel marked "P" is the center point, the point where the mask is fitted onto an image
- Each group of pixels marked with A or B must have at least one foreground (black) pixel

The mask is used to preserve connected components of an image, because if it (or any of its 90° rotation) fits onto a pixel, it means that erasing it would break apart a connected component into two distinct parts.

5.1.1.4 Complex methods for line thinning

There are numerous other algorithms that do not fit into the above categories. Most are based on a unique idea with which the skeleton of an image can be extracted.

One example is the Sparse Pixel Vectorization method in which a ray is casted and traced inside the foreground of the image components and then the medial points of the rays are connected [73]. Another example is the line following algorithm proposed by Peng in [74], which attempts not to iteratively erode contour pixels, but to extract the skeleton by following the two sides of the stroke in the input image (this method is a natural approach, as the human eye identifies lines in the same way). Our own research also includes several alternative approaches e.g. the physical simulation of a pen [75].

These algorithms are usually more complex to implement and can be quite vulnerable to various attributes of the input skeleton, thus careful configuration is needed in each environment (depending on the line width and other properties of the input shapes). In return, they can perform reasonably well in certain areas

and usually have low computational cost since they do not have to examine every pixel, nor to iterate recursively.

5.1.1.5 Stroke extraction

In signature verification, the notion of thinning is often strongly coupled with the reconstruction of the drawing order of the thinned lines [76]. This combined approach is called stroke extraction. Several stroke extraction methods have been introduced in the past. Some robust stroke extraction solutions have been developed for the purpose of recognizing handwritten text [77] [78] and there seems to be an extensively wide study of extracting strokes from Chinese characters [79] [80] [81] [82]. However, these methods tend to work at a high level of abstraction (they focus on recognizing letters and words) and are thereby not suitable for detecting the fine features used in signature verification.

Another class of methods is based on simple line tracing. Either because the resolution of the signature is already low [46], or mostly because they apply some line thinning algorithms [83] [51] [39]. In both cases, the loss of semantically important information [11] is high. A three-stage stroke extraction involving a heuristic stroke following method has been proposed in [84]. However, this only targeted characters and graphemes.

5.1.2 The Scanning Circle Algorithm

In the following section, a robust algorithm is introduced with the purpose of identifying how exactly the signer wrote his signature. The main goal was to create an algorithm that performs well on noisy, unprocessed images.

In general, this method traces a signature using the image of it, extracts control points, determines their order, and finally assigns them to strokes. This produces a graph representation of the signature, which can be used for further processing. As several steps in the algorithm require the scanning of pixels around a circle, we are going to refer to it as the Scanning Circle Algorithm (SCA).

First, the core of the algorithm is introduced, then this is refined to cope with some specific challenges.

5.1.2.1 Basic stroke extraction algorithm

Definition 5.1

- **Signature point**: a pixel in the image that belongs to the signature
- **Stroke point**: the control points of the polylines representing the strokes
- $\boldsymbol{p}$: average pen width (in pixels)
- $\boldsymbol{r}$: the radius of the scanning circle (in pixels) initially $r = 2p$
- $\boldsymbol{d}$: threshold (in pixels), where $d < r$ initially $d = r - 1$
- **Weighted middle point**: given a series of points $P_1, P_2 \dots P_n$ with corresponding intensity values of $i_1, i_2 \dots i_n$, the weighted middle point P_m is the first point in the series where

$$\sum_{k=1}^{m} i_k > \frac{1}{2}\sum_{k=1}^{n} i_k \tag{5.1}$$

- **Free point**: a pixel which has a minimum distance of d from a previously detected stroke point
- **Connected points**: pixels which can be connected with a straight path consisting only of signature points

Using the above definitions, the algorithm is defined as follows:

Step 1 – Locate a signature point

1. Going from the bottom to the top and from the left to the right, locate the first free signature point (P). This point is going to be the starting point of our next stroke: $S := P$.
2. If there is no free point, then exit.

Step 2 – Find connected stroke points (see Figure 5.4)

3. Examine all points of a circle with the center S and the radius r. More than $p/2$ adjacent signature points on the circle define arcs. Select the weighted middle points ($A_1, A_2 \dots A_n$) of these arcs as possible stroke points.
4. Deselect A_i where A_i is not free, or not connected with S. (see Figure 5.4)

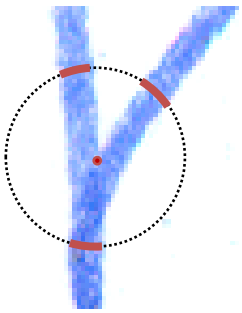


Figure 5.4 Step 2 of the stroke extraction algorithm

The circle in the middle represents S, the dotted line is the scanning circle around S with radius r, the located arcs are shown in red around the circle.

Step 3 – Finish the stroke (if needed) and repeat

5. If count(selected A) is

   0: Finish the current stroke and go to Step 1

   1: Take the arc point as the next stroke point ($S := A$) and go to Step 2

   > 1: Finish the stroke and begin two (or more) new strokes with starting point S, and take A_1, A_2 ... as the second point. Now, follow Step 2 for each of them.

Figure 5.5 shows this logic in a form of a flow chart. A sample run of the algorithm is illustrated in Figure 5.6.

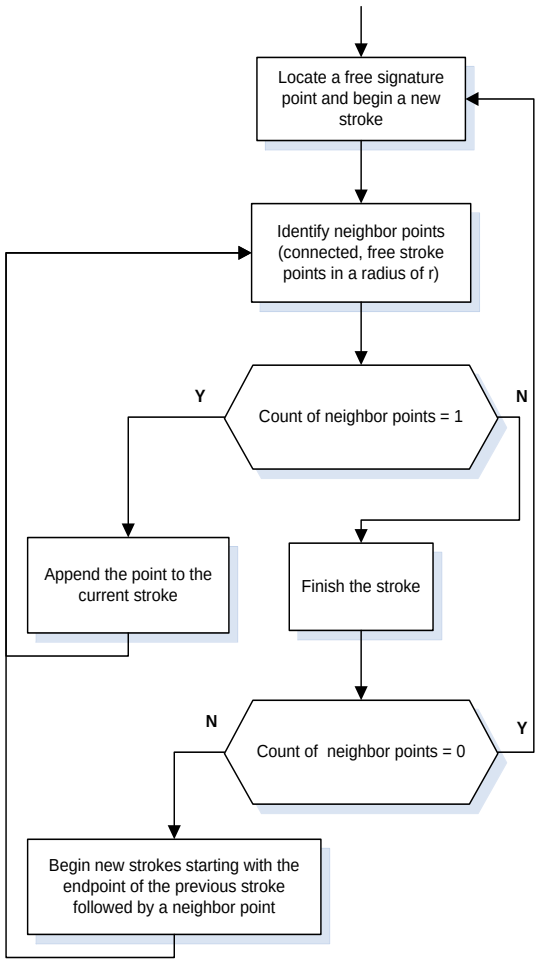


Figure 5.5: Flowchart of the stroke extraction algorithm

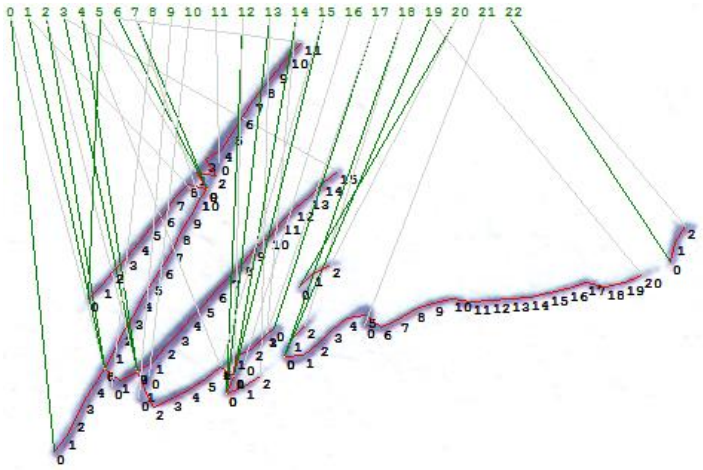


Figure 5.6: A sample run of the basic stroke extraction algorithm

Stroke starts and ends are numbered and marked by the green and gray lines.

22 strokes were located in this signature.

5.1.2.2 Parameterization and extensions

In the field of pattern recognition, the usage of fixed thresholds and parameters is usually avoided [85]. Therefore, the parameters (p, r and d) used in the algorithm are all defined as functions of the image contents (p) or of each other (d, r), thereby allowing the algorithm to be applied to different pen types and different resolutions without manual adjustments.

There are still several elements that can strongly affect the performance of the algorithm. The definition of the "signature point" deliberately does not state how to determine whether a pixel is part of the signature or not. The simplest way of doing this would be to perform noise filtering and a binarization of the original image, perhaps adding morphological closing, as well. This may be adequate for handwriting recognition. However, in the field of signature verification, every single pixel value can be of great importance, therefore a more sophisticated implementation was chosen. Instead of binarization, the pixels of the image are divided into three classes (ink, paper and undefined), using two intensity thresholds.

Every pixel which has an intensity below the lower threshold is assigned to the "ink" class. These pixels probably belong to the signatures. Every pixel with intensity values above the higher threshold is assigned to the "paper" class. All other pixels (with intensity values between the lower threshold and the upper threshold) are assigned to a new class called "undefined".

In Step 1 a pixel is considered to be a "signature point" if in a radius of $p/2$ pixels around it less than 20% of the pixels belong to the paper class. This limit was found to yield the best results. However, this could vary depending on noise ratio and ink type.

To improve performance in step 2, only pixels on the circle around S are considered, and even a single pixel belonging to the "paper" class can break a

row of adjacent pixels. In both cases at least 50% of the pixels must belong to the "ink" class.

The definition of connected points also needs some adjustment. Connected points are pixels that can be connected with a straight path consisting only of ink and undefined points where the number of connected undefined points cannot exceed $p/4$, and the total number of undefined points must be less than the total number of ink points.

5.1.2.3 Stroke sequence estimation

When reaching junction points, several new strokes are defined. This is a necessary side effect of the algorithm which is easy to eliminate in a post processing step. The aim of the stroke sequence estimation is to combine these strokes in a way that will make the resulting strokes correspond to the original handwritten strokes. In Figure 5.7, this would mean that a_1 and a_2 (and similarly b_1 and b_2) have to be combined into a single stroke.

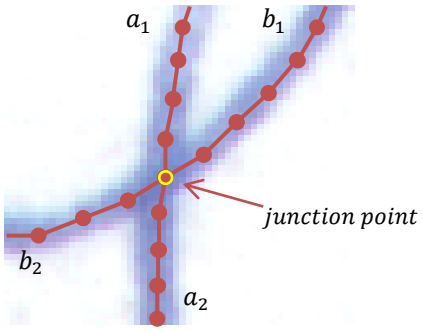


Figure 5.7 The stroke extraction algorithm detects 4 different strokes (a1, a2, b1, b2) in the image, with one common point.

After the stroke extraction algorithm, connected strokes will have common endpoints – because of step 3 –, therefore these common endpoints can be used to locate junctions. A junction point is the meeting point of at least 3, but possibly more strokes.

There are several existing solutions for stroke sequence estimation in the literature. In [86], a mainly local approach is taken by examining typical patterns of thinned images. [87] and [88] take a more generic approach by encapsulating local and global heuristics in the algorithm. As [88] states, to make optimal decisions, almost every combination of strokes must be examined which yields to a combinatorial explosion. To avoid this, our approach will be limited to local decisions (only one junction is considered for examination at a time). This also allows us make the best possible local choice.

Definition 5.2 (Stroke sequence estimation algorithm)

Let n be the count of strokes starting from a common junction point. It is assumed that at least $\lfloor n/2 \rfloor$ stroke pairs have to be combined, therefore all possible stroke-pair combinations are examined. A distance function which

calculates the differences of tangents at the junction points is applied to each stroke pair to calculate the goodness of the match. The best combination is where the sum of distances is the lowest.

A sample run of the improved algorithm is demonstrated in Figure 5.8.

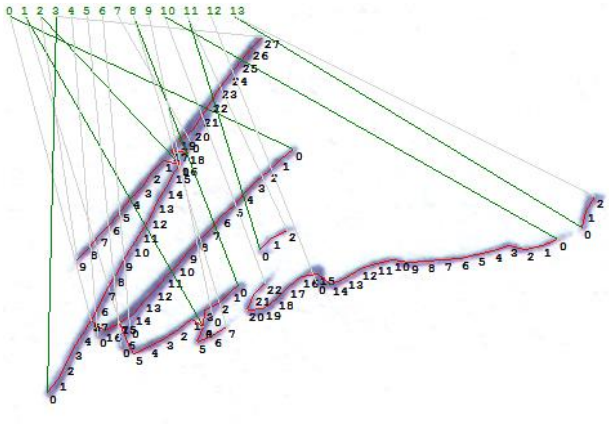
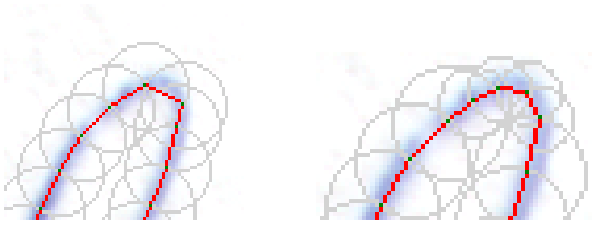


Figure 5.8 A sample run of the improved stroke extraction algorithm

The number of strokes was reduced from 22 to 13.

5.1.2.4 Adaptive circle radius

Although the original algorithm performs well when following straight lines, curves and even when detecting junctions, the high value of the scanning radius (r) makes for it impossible to correctly detect sharp curves in the original signature's strokes. This can result in the detection of phantom strokes (Figure 5.8).



a) b)

Figure 5.9 Stroke extraction

with constant (a) and with adaptive (b) scanning radius.

To tackle this problem, Step 2 is modified to allow the changing of r. When the angle between two stroke segments falls below a given angle or the number of points in an arc goes significantly beyond the pen width (p), a new scanning circle is tested with a reduced radius. The radius is decreased several times if necessary, but it cannot get under $p/2$. In the next iteration, Step 2 will start with the reduced radius, but contrary to the previous case, it will try to increase the radius if it can whilst maintaining a maximum value of $2p$. Figure 5.9 illustrates the difference between the two solutions.

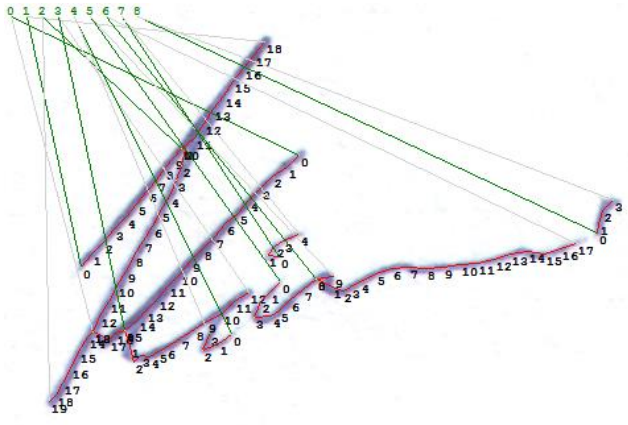


Figure 5.10 Improved stroke extraction

5.1.2.5 Experimental results

Because of the use of several heuristic methods in the algorithm, a continuous monitoring of accuracy is essential. Although we were able to validate single cases with human interaction, a statistical validation is hard to obtain because of the lack of on-line information. To compensate for this, the SVC2004 database was used. This corpus was constructed from an on-line signature database, therefore it contains the original stroke information to be compared to the extracted values.

A comparison can be given by comparing the actual number of strokes to the stroke count detected by SCA. Figure 5.13 compares these average stroke counts for each of the signers in the SVC2004 database. It can be seen that our algorithm always overestimates the count of strokes, which can be tracked back to the fact that nearby strokes do not always have a common point. Therefore the connection of nearby strokes is not always possible (Figure 5.11).

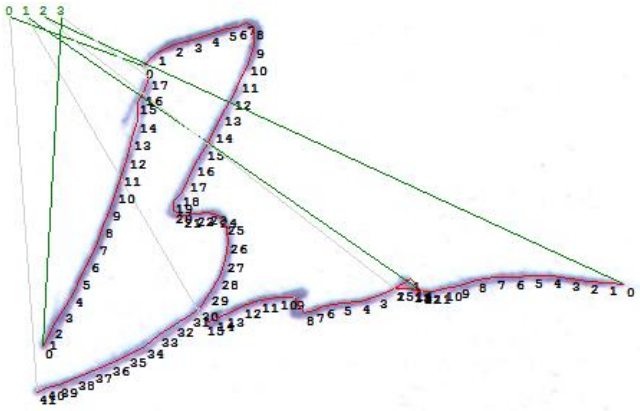


Figure 5.11: 4 strokes are detected instead of the original 3.

However, these results are still promising, because these “extra” strokes are detected in the same way in all signatures. Figure 5.13 shows the standard deviation of the detected stroke counts for each of the signers. The average value of 1.3 is relatively low given that there are signatures with more than 14 detected strokes. This indicates that the output is (mainly) consistent and could thereby be used in further comparison methods.

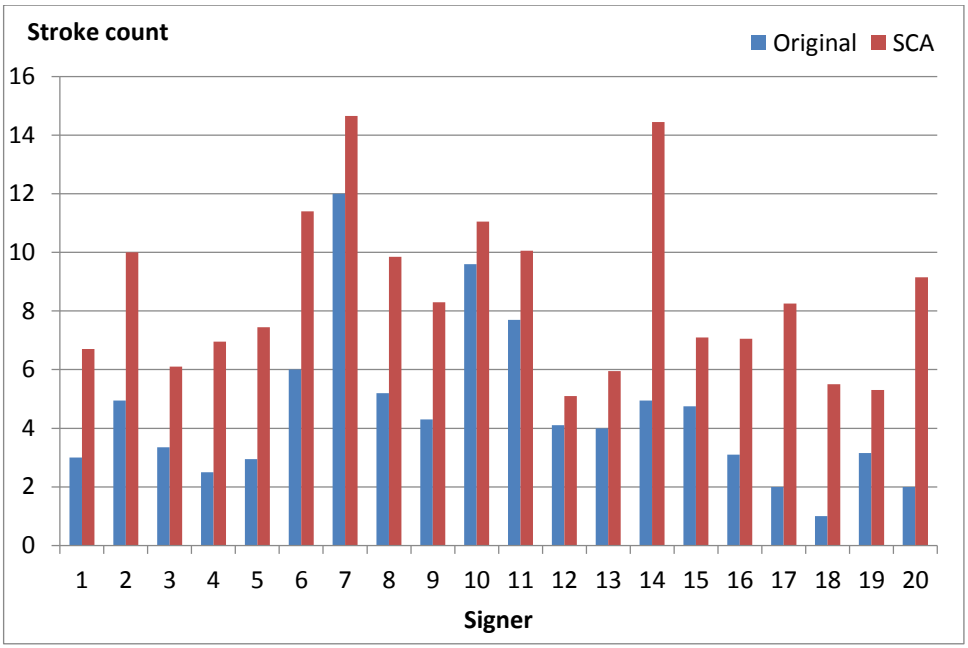


Figure 5.12 Actual and detected average stroke counts

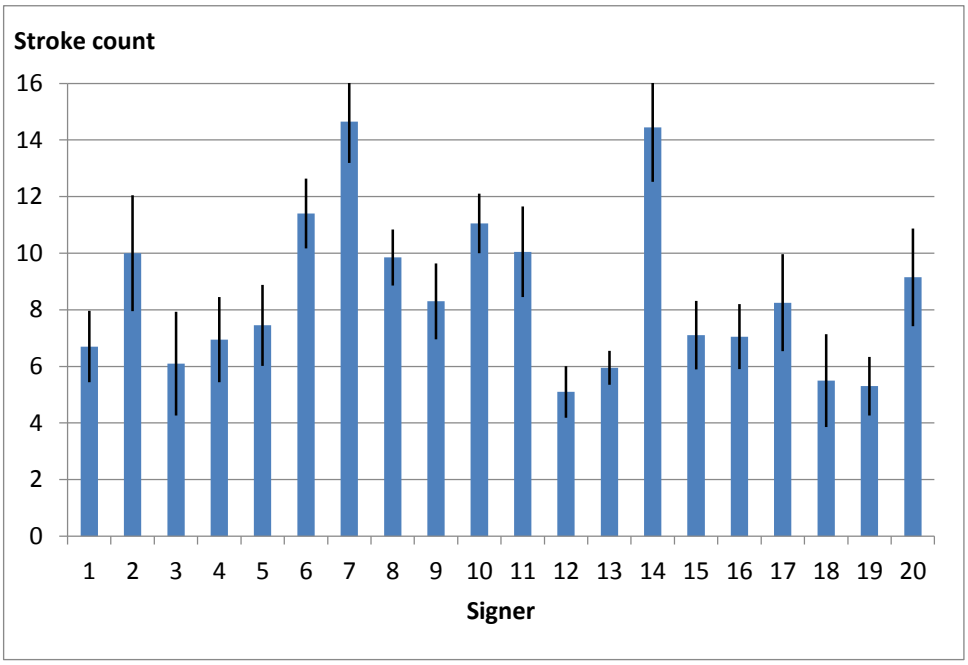


Figure 5.13 Average and standard deviance of stroke counts

Figure 5.14 shows that the changes of the previous three sections really improved the accuracy of the stroke extraction algorithm by reducing the standard deviation of the detected stroke counts.

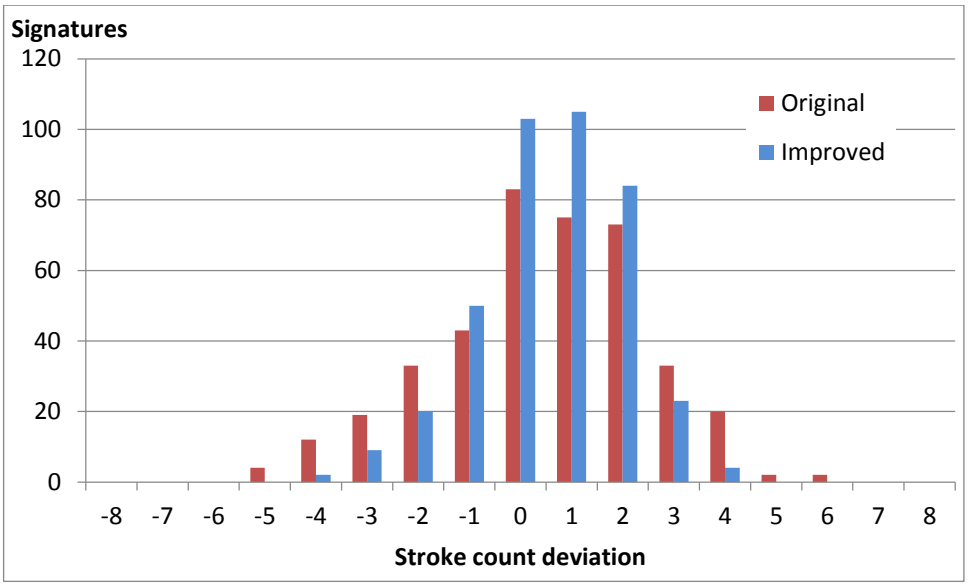


Figure 5.14 Histogram of stroke count deviations in the original and improved SCA implementation

5.1.3 Comparison with other algorithms

There is no widely adopted strategy, nor a common corpus for comparing stroke extraction algorithms; therefore we are going to limit our experiments to the evaluation of the thinning capabilities of SCA.

The process uses reference images of the SVC2004 corpus. The thinned result of each algorithm is compared to the ideal reference skeleton, which is obtained from the on-line signature information (Figure 5.15). The starting point of the comparing technique is the evaluation algorithm introduced by Pratt in [34].

Figure 5.15 Reference skeleton and the reconstructed signature

5.1.3.1 Pratt Evaluation

Let N_R and N_A be the number of foreground (black) points in the reference skeleton and the actual thinned result, respectively. The distance between the i^{th} pixel of the result skeleton and the pixel of the reference skeleton nearest to it is notated by d_i. The Pratt evaluation score is defined by

$$PES = \frac{1}{N_M} \sum_{i=1}^{N_A} \frac{1}{1+a_P d_i^2} \tag{5.2}$$

where $N_M = \max\{N_R, N_A\}$ and a_P is a scaling constant, adjusting the level of penalization for distances.

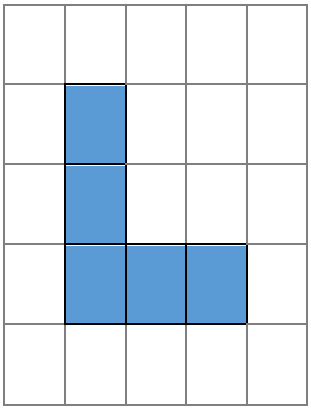


Figure 5.16 Example of a Pratt evaluation rating

In the above example, the evaluation score is calculated as follows:

$$N_R = 5, N_A = 4 \tag{5.3}$$

$$d_1 = 1, d_2 = 1, d_3 = 0, d_4 = 0 \tag{5.4}$$

$$PES = \frac{1}{5}\left(\frac{1}{1+a_P \cdot 1^2} + \frac{1}{1+a_P \cdot 1^2} + 1 + 1\right) = 0.76 \tag{5.5}$$

if a_P is chosen to be 1/9 (which is a typically used value [34]).

If the two compared skeletons are identical to each other, then the result is 1.0 while lower values mean a lower level of similarity.

5.1.4 The experimental setup

The following fourteen algorithms have been implemented [89] and tested with the above evaluation method: raster scan by Rutovitz, 1966 [72], raster scan by Yokoi, 1973 [71], window matching by Beun, 1973 [71], contour scan by Arcelli, 1978 [71], window matching by Pavlidis, 1981 [71], raster scan by Holt, Stewart, Clint and Perrott, 1987 [72], window matching by Chin, Wan, Stover and Iverson, 1987 [72], raster scan by Zhang and Wang, 1988 [72], raster scan by Eckhardt and Maderlechner, 1993 [72], window matching by Wu and Tsai, 1992 [72], raster scan by Guo and Hall, 1992 [72], window matching by

Jang and Chin, 1992 [71], raster scan by Hilditch, [71], and the Sparse Pixel Vectorization method by Dov Dori and Wenyin Liu, 1999 [73].

The first signature from each of the signers of the SVC2004 database was used to evaluate the different algorithms (Figure 5.17). The average score of the algorithms is summarized in Figure 5.18 and

Table 5.1. It can be seen that the best algorithm has a Pratt-score of 0.881, while SCA is only 2.3% behind with 0.858.

Figure 5.17 Some examples of the thinning results

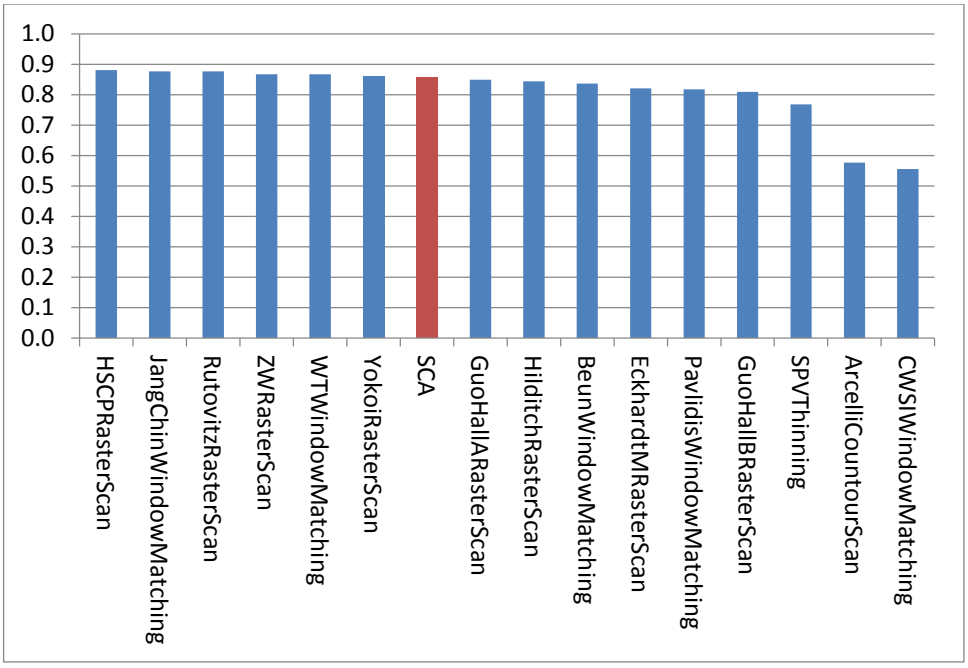


Figure 5.18 Comparison of average Pratt-scores

Reference skeleton Imitated signature

Figure 5.19 Raster scan by Guo and Hall

In order for the thinned images to be useful in the reconstruction of strokes, they must preserve connectivity. Several of the examined methods do not satisfy this criterion, which is not desirable; therefore we have examined this property of each of the introduced algorithms. Finally, as a third aspect of the analysis we have added the average ratios of pixels in the extracted and in the reference signatures.

Experimental results show that the accuracy of SCA is comparable to (although slightly behind) the best thinning algorithms examined. This makes the algorithm a suitable base for further use in a signature verification process, which is the subject of our ongoing research.

Table 5.1 The average ratings, tested with 20 signatures

Algorithm	Pratt rating	Keeps connectivity	Extracted/reference pixels
HSCPRasterScan	0.881	yes	1.050
JangChinWindowMatching	0.877	yes	0.936
RutovitzRasterScan	0.876	yes	1.114
ZWRasterScan	0.867	yes	1.049
WTWindowMatching	0.867	yes	0.946
YokoiRasterScan	0.860	yes	0.933
SCA	0.858	yes	0.942
GuoHallARasterScan	0.849	no	1.063
HilditchRasterScan	0.843	yes	0.929
BeunWindowMatching	0.836	yes	0.930
EckhardtMRasterScan	0.821	no	1.349
PavlidisWindowMatching	0.818	yes	1.413
GuoHallBRasterScan	0.809	no	0.945
SPVThinning	0.768	no	0.872
ArcelliCountourScan	0.576	yes	0.960
CWSIWindowMatching	0.556	no	0.653

5.2 Feature matching

In this section, the algorithms used to match the features of the signatures will be discussed.

5.2.1 Related work

Assumed that a set of original signature images is available as reference we have to decide whether a new image has the same features or not. This image correspondence process is called image registration in the literature. A large number of different applications use image registration, from medical imaging to remotely sensed data processing. There are two major categories of image registration [90], [91], area-based and feature-based methods.

Area-based methods are correlation-like techniques. These algorithms use Fourier properties and other means of structural analysis. There are some limitations, e.g. these methods often work with a rectangular window and if the images are deformed by more complex transformation, the window is not able to find some parts of the scene. Since there can be large variability between the signatures of a signer, area-based methods are less applicable in the case of off-line signature analysis.

Feature-based methods assume that two sets of features have been detected. These find the correspondence between reference and examined images using their spatial relations or various feature descriptors: lines, curves, points, line intersections, boundaries etc.

One innovative way of doing this is the one Valyon et al. use in [92] as preprocessing, a hybrid approach creating a point-to-point matching for further decision-making.

Figure 5.20 Loop matching

To allow a comparison of features, a mapping should be defined between the features of each signature pair. This happens in two phases. During the training phase, a mapping should be established among the features of the reference (original) signatures. During testing, the features of a sample signature should be mapped to the previously matched features. This is a multidimensional assignment problem [93] which is NP hard; therefore we propose special heuristics for it, to take some special properties of the signatures into consideration.

5.2.2 Matching reference signatures

The graph in which the multidimensional assignment problem arises can be defined as follows.

Definition 5.3

- let the nodes of the graph represent the features in the signatures
- let the edges of the graph represent the similarity of features. Therefore, an edge should be defined between each node-pair except the node-pairs within the same signature. The weights of the edges represent how good a matching is (calculated from shape similarity and Euclidean distance)

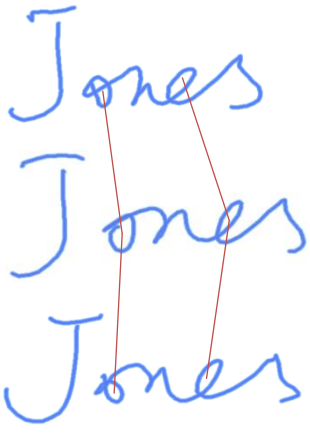


Figure 5.21 Matching of loops

During the matching process, all original samples of the signer are processed together. We focus on identifying matches between features which are available in most of the (currently 80%) signatures. Signatures which do not have a given feature will be ignored when calculating the statistics for it. This is a necessary tradeoff to compensate for the diversity of original signatures and errors in previous processing steps.

The matching process consists of the following steps:

Step 1 – Pairwise matching of signatures (Figure 5.22)

1. Take the nodes representing a signature pair and the edges between them. Calculate the optimal matching between them using the Hungarian algorithm [94] and remove all other edges.
2. Do this for each signature pair.

Step 2 – Enumerating subgraphs (Figure 5.23)

3. Take a node from the graph. This node is connected with at most 1 node from each other signature. Count the edges of this subgraph.
4. Do this for each node in the graph.

Step3 – Removing the best match (Figure 5.24, Figure 5.25, Figure 5.26)

5. Take the subgraph from step 2 with the highest edge count.
6. Let the nodes from this graph define the mapping of a common feature.
7. Remove the nodes and the edges of this subgraph from the original graph.
8. Repeat from step 2 to locate further feature mappings.

We found that the above algorithm produced practical mappings between features in a time-efficient manner. The main steps of the algorithm are illustrated in the figures below.

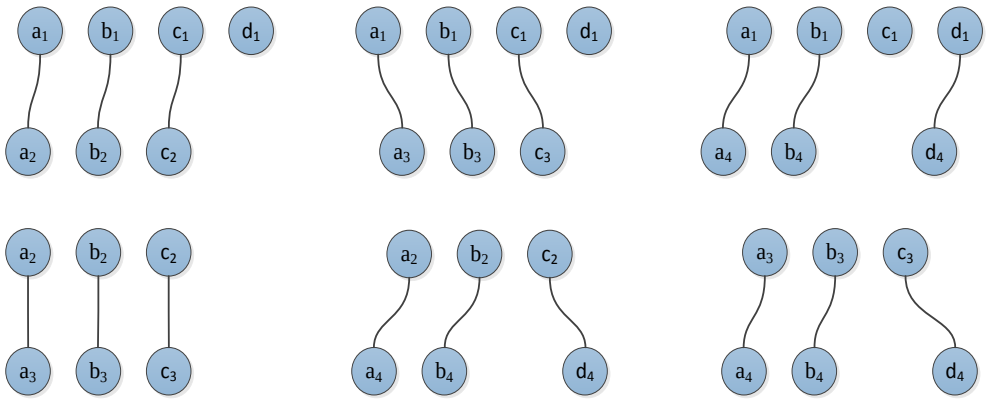


Figure 5.22 The pairwise matching of signatures:
letters represent features, while lower indexes identify the signature

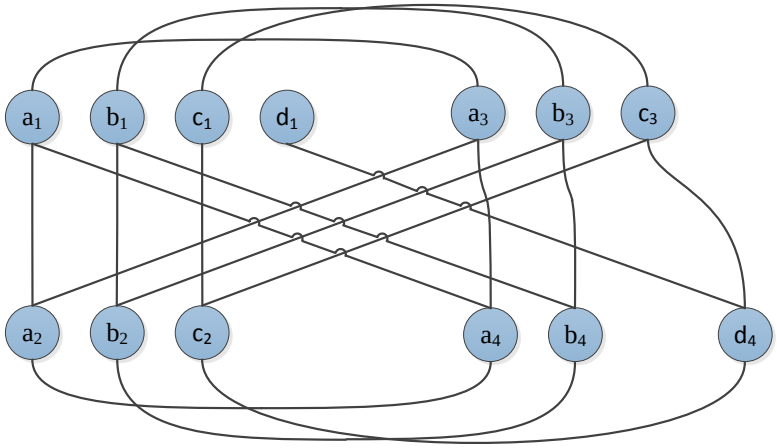


Figure 5.23 The combined graph of feature pairs

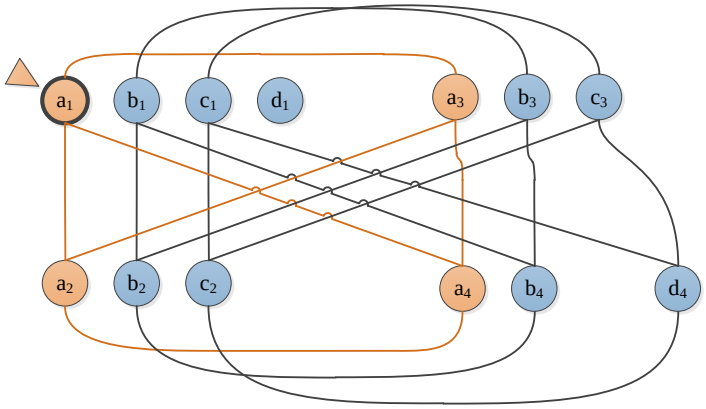


Figure 5.24 Iteration 1 of step 3

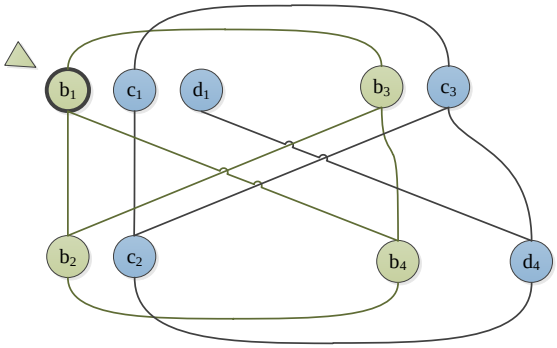


Figure 5.25 Iteration 2 of step 3

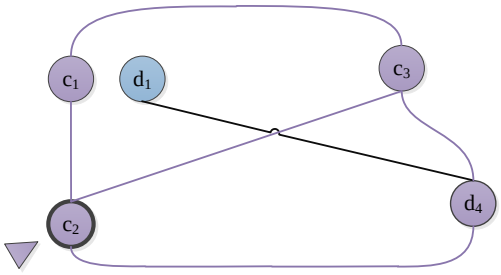


Figure 5.26 Iteration 3 of step 3, note that d_4 is a false match

5.2.3 Matching sample signatures

The next algorithm allows us to map the features of a new signature to the already mapped common features. For this, we also use the previously defined graph representation, extending it with the nodes of the new sample.

Step 1 – Pairwise matching of the sample to the reference signatures

1. Take the nodes from a reference signature and the nodes from the sample signature and the edges between them. Calculate the optimal matching between them using the Hungarian algorithm [94] and remove all other edges.
2. Do this for each reference signature.

Step 2 – Finding the best match

3. Assign each node of the sample signature to the common feature to which it has the most connections.

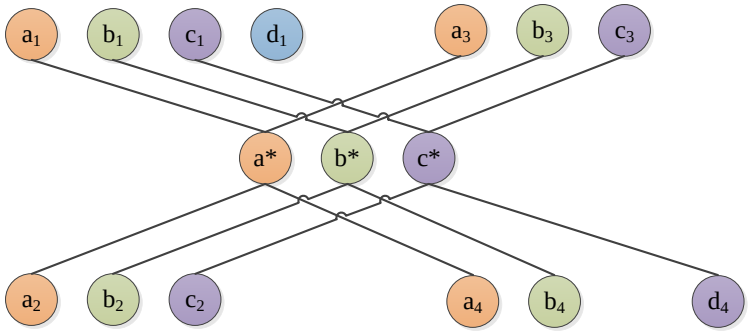


Figure 5.27 Matching a sample signature to four references

5.3 Feature models and extraction algorithms

This section provides a brief description of the properties and extraction of features used in our previous experiments.

5.3.1 Global features

The bounding rectangle is defined as the smallest rectangle needed to enclose the signature. As the bounding rectangle may be severely influenced by a single misplaced stroke or accent, a special variant is also introduced. The trimmed bounding rectangle is the smallest rectangle needed to enclose the binarized version of the signature image by ignoring the first 5% of pixels from each of the four directions (top, bottom, left and right). Based on these rectangles the following global features were used to characterize a signature:

- *Width*: the width of the bounding rectangle
- *Height*: the height of the bounding rectangle
- *Aspect ratio*: width / height
- *Trimmed width*: the width of the trimmed bounding rectangle
- *Trimmed height*: the height of the trimmed bounding rectangle
- *Trimmed aspect ratio*: trimmed width / trimmed height

5.3.2 Baseline model

The ASTM International standard defines a baseline as “the ruled or imaginary line upon which writing or typewriting appears to rest” [95]. Baseline information consists of a set of straight lines. Each line represents an imaginary foundation of a component (usually a word), which can be regarded as an autonomous element of the signature [96]. This definition also allows us to assign baseline information to the gaps between signature elements (according to [11] those gaps are just as characteristic as any other feature of a signature). The following 6 parameters were used to describe a baseline:

- horizontal start position (X),
- vertical start position (Y),
- horizontal end position (X),
- vertical end position (Y),
- length,
- angle.

Figure 5.28 Baselines obtained by our algorithm

Figure 5.29 Phases of the baseline extraction process

The extraction of baselines is performed in a multi-step image-processing algorithm illustrated in Figure 5.29. The main steps of the algorithm are the following:

1. Take original signature
2. Convert to grayscale, using R-Y algorithm
3. Binarize using iterative threshold algorithm
4. Invert image
5. Fill holes in objects
6. Apply dilatation operator
7. Apply dilatation operator
8. Apply erosion operator
9. Apply erosion operator
10. Identify connected components
11. Extract the lowest pixel for each column of each component (lower envelope) and remove overlapping components
12. The baselines are calculated from the pixels of the lower envelopes through linear regression

5.3.3 Loop model

A loop is a "formation that curves and crosses itself" [95]. Loops – in our interpretation – are connected regions in the image that are fully enclosed by the pixels of the signature (Figure 5.30).

This definition implies the following 3 important properties:

- Pixels should be unambiguously classified as either belonging to the background ("paper" pixels) or belonging to the signature ("signature" pixels). This is currently done by testing the color components of a pixel against some fixed thresholds.
- The region must be connected. Although it sounds logical at first glance, this is against the traditional definition of a loop, which can be interrupted by other lines. However, this simplification allows a much faster processing of the image.
- Using fully enclosed regions showed to be an unrealistic target. Because of flaws of the pen and errors of the scanning process, there are often 1-2 pixel wide interruptions in pen strokes, which would break our definition of loops. To eliminate them, a morphological closing [97] is applied (two dilations and two erosions with a 3x3 structuring element) to each image before loop extraction.

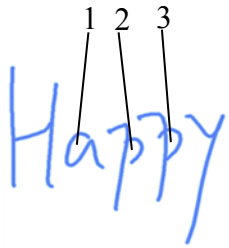


Figure 5.30 Loops within a signature

Shape descriptors [85] are used to describe the different aspects of loops, thereby allowing an easy comparison in later phases [98]. There are several promising formulas described in the literature for calculating shape descriptor values. Instead of choosing one of them, we used as many of them as possible. This will allow us to identify the most significant shape factors in later phases.

The following shape descriptors were used during feature extraction: perimeter, area, formfactor, maximum diameter, maximum diameter angle, roundness, centroid, bounding box, inscribed diameter, extent, modification ratio, compactness, bounding circle, moment axis angle, convexity, solidity, aspect ratio.

Nontrivial descriptors are defined as follows:

$$Modification\ Ratio = \frac{Inscribed\ Diameter}{Maximum\ Diameter} \tag{5.6}$$

$$Formfactor = \frac{4\pi \cdot Area}{Perimeter^2} \tag{5.7}$$

$$Extent = \frac{Area}{Bounding\ Rectangle\ Area} \tag{5.8}$$

$$Compactness\ = \frac{\sqrt{\left(\frac{4}{\pi}\right)Area}}{Maximum\ Diameter} \tag{5.9}$$

$$Convexity = \frac{Convex\ Perimeter}{Perimeter} \tag{5.10}$$

$$Solidity = \frac{Area}{Convex\ Area} \tag{5.11}$$

$$Aspect\ Ratio = \frac{Maximum\ Diameter}{Minimum\ Diameter} \tag{5.12}$$

A detailed introduction of shape descriptors can be found in [85].

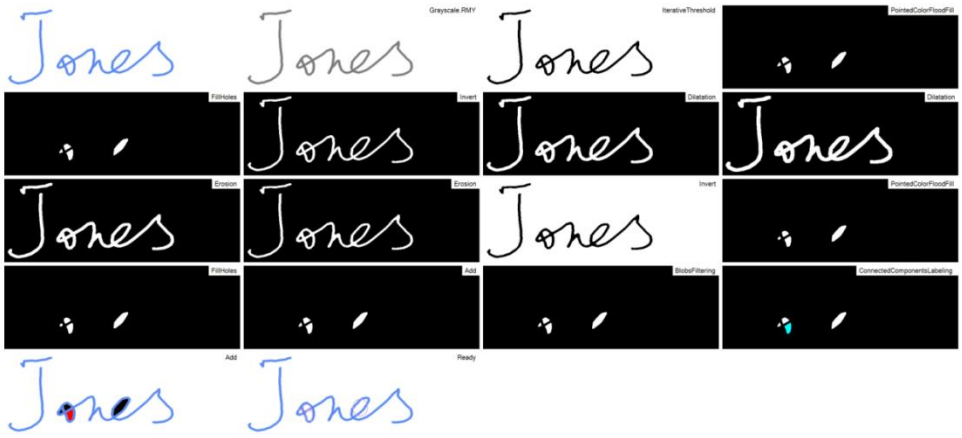


Figure 5.31 Phases of the loop extraction process

The extraction of loops is performed in a multi-step image-processing algorithm illustrated in Figure 5.31. The main steps of the algorithm are the following:

1. Take original signature
2. Convert to grayscale, using R-Y algorithm
3. Binarize using iterative threshold algorithm
4. (1st Copy) Flood fill with black color starting from top first point
5. (1st Copy) Fill holes in objects
6. (2nd Copy) Invert image
7. (2nd Copy) Apply dilatation operator
8. (2nd Copy) Apply dilatation operator
9. (2nd Copy) Apply erosion operator
10. (2nd Copy) Apply erosion operator
11. (2nd Copy) Invert image
12. (2nd Copy) Flood fill with black color starting from top first point
13. (2nd Copy) Fill holes in objects
14. Add 1st Copy and 2nd Copy
15. Remove objects smaller than 5x5 pixels
16. Label connected components with different colors
17. Project components on the original signature (for monitoring)
18. Extract loop information

5.4 Chapter summary

In this chapter, a method was proposed for the extraction of stroke information from scanned images of handwritten signatures. We have overcome the problem of detecting sudden direction changes in the strokes by using an adaptive

scanning circle radius and provided a locally optimal algorithm for estimating the drawing sequence of directly connected strokes. Experimental results show that – in terms of thinning – the accuracy of the Scanning Circle Algorithm is comparable to the best thinning algorithms examined. This makes the algorithm a suitable base for further use in a signature verification process.

We have proposed a heuristic algorithm for the solution of multidimensional assignment problem in the case of signature verification. A simplified version of the algorithm ensures that features of sample signatures can also be assigned to the matched features of reference signatures.

We have described a way of characterizing loop and baseline information and provided algorithms for the extraction of these features from signature images. These algorithms, together with the classifier introduced in Section 3, cover the whole process of signature verification and serve as the basic building blocks of our signature verification system, to be discussed in the next chapter.

6. The application of the results

This thesis is the result of extensive, seven-year-long research. During this period, a rich application system was built around the core of our signature verification system. This section briefly introduces some of the most important applications.

6.1 Signature verification framework

Our signature verification framework is a collection of 37 C# projects, consisting of approximately 148,000 lines of code. About 10% of this codebase belongs to a single class library, which contains the most mature modules and the framework itself in a reusable and portable form.

The verification framework itself is an implementation of the model presented in Section 2.8. It coordinates the training and testing phases and provides benchmarking. Individual modules interact with the framework through the implementation of 4 well-defined interfaces belonging to the different processing phases. A detailed description of these interfaces can be found in Appendix B.

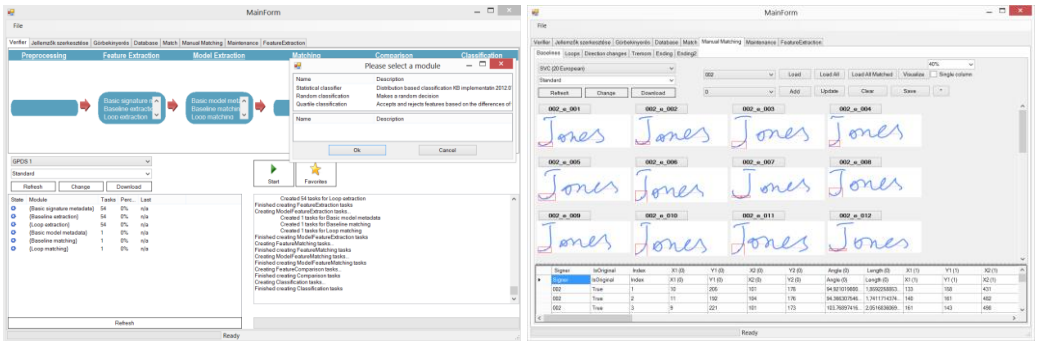
6.1.1 Architecture

One of the main goals of modern software design is to keep cohesion (within a component) high and coupling (between components) low [99]. As we have seen in section 2.6, state-of-the-art signature verifiers mostly lack these characteristics. Although they usually isolate some of the main modules, the core is often a highly coupled composition of different components, often incorporating loops and other complex constructs in the information flow. While this is not necessarily a design flaw, it makes the objective evaluation and the comparison of different systems hard and the simultaneous development of modules almost impossible.

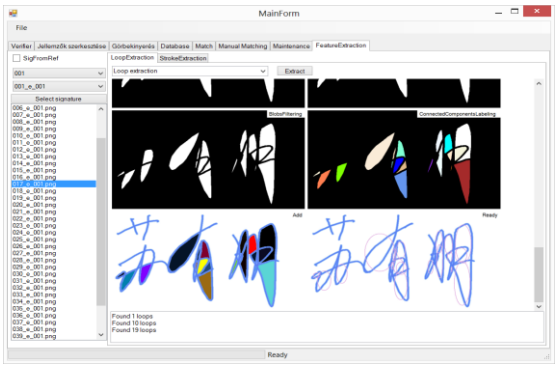
Our model reduces coupling to a minimum. Only signature files and their corresponding metadata are passed between components. The input and output parameters of the interface functions are restricted to the minimum without limiting the functionality of the modules.

The data flow is well defined without any feedback among the processing phases. This structure allows independent development and testing of modules. Although dependencies are possible (e.g. a matching module may depend on a specified feature extraction module) the components implementing the same interfaces are interchangeable. In addition, several components (e.g. feature extraction algorithms) can be used simultaneously to create more sophisticated input for the classification module.

In the next subsections we shall present examples of how this model can be applied to aid the implementation and testing of various signature verification tasks.



a) b)



c)

Figure 6.1 Screenshots from the framework

a) bulk processing b) manual feature matching c) debugging view of the loop extraction algorithm

6.1.2 A modular system

A generic signature verification framework has been established using the previously introduced architecture. Currently, the system supports the addition of:

- custom preprocessing steps,
- custom feature extraction functions,
- custom feature matching functions,
- custom classifiers.

The architecture allowed us to create benchmarks for each of the implemented components and thereby not only measure their effects on the global verification results, but to compare them to each other.

Currently, we have working modules for performing the following tasks:

Preprocessing

- cropping
- removing underlines (3 implementations) [100] [101] [102]
- noise removal
- skeletonization (2 implementations)

- scale transformation
- rotation normalization
- Matlab wrapper

Feature extraction

- extraction of strokes based on the skeleton
- extraction of endpoints based on the skeleton
- extraction of junctions based on the skeleton
- extraction of strokes based on the original image [103]
- extraction of intensity information from strokes
- extraction of line endings [104]
- extraction of slopes [104]
- extraction of baselines (2 implementations) [96]
- extraction of loops (2 implementations) [98]
- extraction of tremor information [104]
- extraction of basic size information
- extraction of lower and upper envelopes [101] [105]
- extraction of signature angle [104]

Matching

- matching of strokes [103]
- matching of baselines [96]
- matching of loops [98]
- generic matching algorithm (2 implementations) [106]

Classification

- threshold based classifier
- ellipsoid classifier [102]
- quartile based classifier
- random classifier
- neural network based classifier (3 implementations) [107] [108]
- statistics based classifier (2 implementations) [102]

6.1.3 Education

The above results could not have been reached without the modular nature of the system. It allowed students to get engaged in the task of signature verification without having to implement a whole verification system on their own. 10 Master's thesis, 7 bachelor's thesis and a Scientific Students' Associations Conference paper are the fruit of their work.

6.2 Analytical tool

The Signature Manager (SigMan) [109] application was created by Bence Kutasi to assist in the visual analysis of signatures and features. The highly modular Windows Presentation Foundation (WPF) based application is able to visualize the previously extracted features beside the signature. All stored descriptive information about the signature and its features can be edited either directly in XML or in a more structured way through a property editor interface.

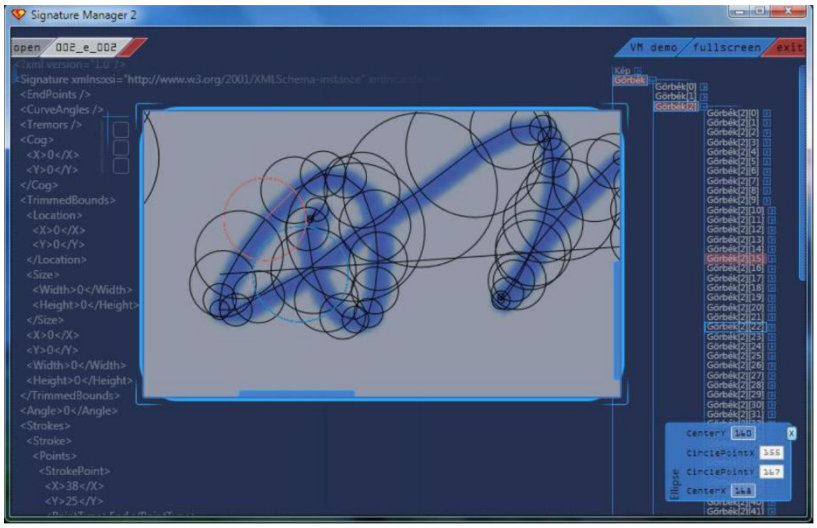


Figure 6.2 Screenshots from the SigMan application

6.3 Silverlight Frontend

To allow other researchers, students and visitors to experiment with our system, Mátyás Pálfalvi created a public interactive frontend [110], available on the homepage of the Department of Automation and Applied Informatics at BME.

The tool provides a simple set of original and forged signatures to demonstrate the capabilities of the system swiftly, but one can also upload (through dragging and dropping) his/her own signatures for testing.

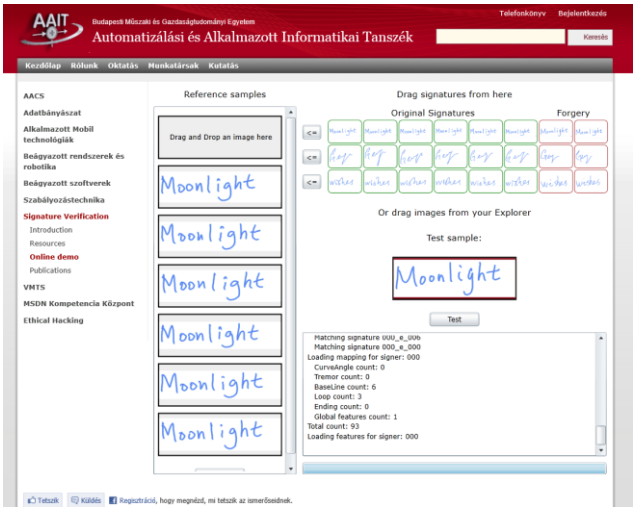


Figure 6.3 Interactive web interface (www.aut.bme.hu/signature)

6.4 WPF Frontend

The user interfaces used during development and testing are overcrowded and complex. The WPF based user interface created by Daniel Nagy [111] demonstrates that the complex inner logic of the system can be presented in a clear and user-friendly way.

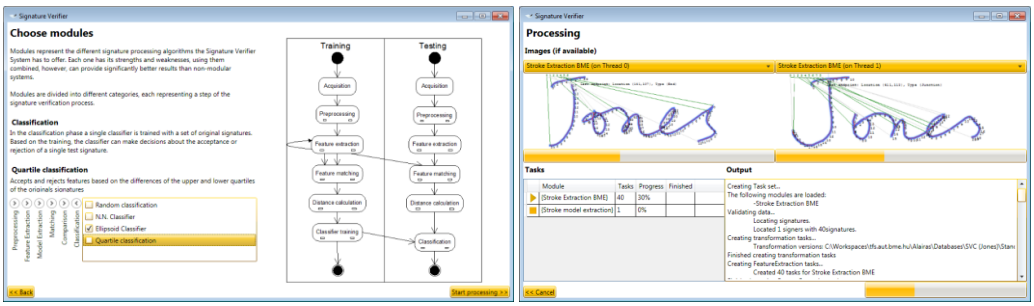


Figure 6.4 Screenshots from the "Signature Verifier" application,
a frontend specifically designed for improved user experience

6.5 Verification reports

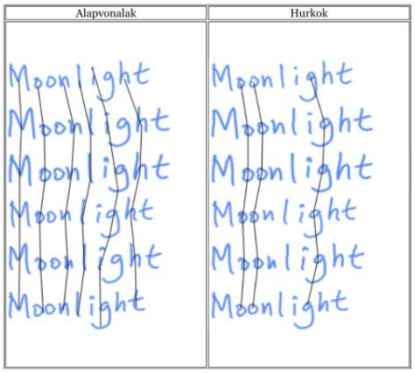
One of the main aims of our research is not only to provide a commercial application for signature verification, but also to deliver meaningful details about the reasons behind a given decision.

The whole model of our system was designed to support this scenario (in contrast to many other similar systems), because:

- The feature set is based on the same features, as used in forensic document examination.
- Each of the features is matched independently, and each of the matched feature groups is evaluated independently.

As a result of these design principles, each signature test conducted by our system also tells us which of the measured features deviated significantly from the usual normal distribution of the signer's features.

As a unique feature among off-line signature verifiers, our system is able to create a report about the verification results of a signature to explain the decision. Instead of abstract results, the report provides a detailed visual feedback about the main processing results as well as statistical parameters for each feature. Using these results, a human expert can easily confirm or question the decision of the system. The current implementation of our reporting component was created by Zoltán Berceli [106] and Mátyás Pálfalvi [110].



	M	S	X	Res
Perimeter	70,83	2,96	66,06	0,46
MaximumDiameter	25,28	1,74	29,41	0,35
MaximumDiameterAngle	-0,94	1,09	1,26	0,4
InscribedDiameter	14,91	2,16	10,83	0,42
BoundingCircleRadius	12,69	0,74	14,71	0,31
Area	341,49	41,26	294,06	0,53
MomentAxisAngle	1,1	0,03	0,92	0,14
Formfactor	0,86	0,14	0,85	0,61
Roundness	0,69	0,11	0,43	0,35
Compactness	0,83	0,06	0,66	0,32
Extent	0,68	0,06	0,68	0,62
Solidity	1	0,06	1,09	0,47
Convexity	0,99	0,04	1,03	0,56
AspectRatio	1,73	0,27	2,72	0,23
ModificationRatio	0,59	0,1	0,37	0,37
X	158,33	6,35	142	0,33
Y	77,67	3,72	101	0,14

Figure 6.5 Parts of the verification report

7. Conclusions

7.1 Summary

In this thesis, we have introduced the problems in feature-based off-line signature verification and proposed a framework that provides a quantitative way to characterize this problem class. By using normality tests, we verified that baseline and loop feature properties can efficiently be approximated with a normal distribution. We have shown that these assumptions can be used to define an acceptance threshold for a feature property that minimizes the average error rate (Lemma 1). We have also shown that when the count of reference signatures is low, the sample size should also be included in our calculations (Lemma 2). Using these results, we were able to predict the average error rate of our verification system (Lemma 3).
By our experiments, we have demonstrated that local features can successfully be used with our system to distinguish original signatures from forgeries with error rates similar to those of our estimations.
Baselines and loops were found to be good features for characterizing signatures written in Latin writing; however, they were insufficient in the case of Chinese writing. Currently, we are studying several other features that could extend the field of applicability and improve the accuracy of our algorithm. Furthermore, we are experimenting with different methods to reduce the possible dependencies among feature properties.

The results discussed in this work are summarized with three theses. I have proven these results with engineering and mathematical methods and have illustrated their practical relevance in engineering applications. Furthermore, I have published these results in several scientific forums.

Thesis I
I have shown that

- the distribution of several feature values can be approximated with normal distribution. The intuitive explanation was confirmed by experimental results. The statement has been also extended for forged feature values, which follow the same distribution but usually with greater variance.

- the average error rate of a classifier with more than 30 samples can be approximated with

$$AER = \frac{\alpha+\beta}{2} = \frac{1}{2} + \Phi\left(\frac{w}{q}\right) - \Phi(w) \tag{7.1}$$

- the previous error rate is always smaller than 0.5 when $q > 1$
- in the previous model, the sum of the probabilities of false rejection and false acceptance is minimized when

$$w = \pm\sqrt{2}\sqrt{\ln q}\sqrt{\frac{q^2}{q^2-1}} \tag{7.2}$$

- approximating the distributions with Student's t-distributions, the error is minimized if

$$w = \pm\sqrt{\left(\frac{q^2 - q^{2+\frac{2}{n}}}{q^{\frac{2}{n}} - q^2}\right)\left(\frac{n^2-1}{n}\right)} \tag{7.3}$$

- If each of the feature properties is either rejected or accepted, and α and β are known, the number of rejected features required to reject a signature while minimizing the average error rate is:

$$l = k\frac{\ln(1-\alpha)-\ln(\beta)}{\ln(1-\alpha)-\ln(\beta)+\ln(1-\beta)-\ln(\alpha)} \tag{7.4}$$

- the lowest achievable average error rate of a signature verification system can be predicted based on the number of reference signatures used to train the system (n), the number of independent feature properties (k) and the quality of the forged features (q) as follows:

$$AER(q,n,k) = \frac{\sum_{j=0}^{\lfloor l \rfloor}\binom{k}{j}\beta^{k-j}(1-\beta)^j + \sum_{i=\lfloor l \rfloor+1}^{k}\binom{k}{i}\alpha^i(1-\alpha)^{k-i}}{2} \tag{7.5}$$

Thesis II

I have proved that

- given the constraints

$$q > 1, k \geq 1, n > 2, n, k \in \mathbb{Z}, q \in \mathbb{R} \tag{7.6}$$

  and provided that the number of reference samples (n_0) and the forgery quality (q_0) is given the function $AER(q_0, n_0, k)$ is decreasing.

- given the constraints

$$q > 1, k = 1, n > 2, n, k \in \mathbb{Z}, q \in \mathbb{R} \tag{7.7}$$

  and provided that the number of observed features (k_0) and the forgery quality (q_0) is given the function $AER(q_0, n, k_0)$ is strictly decreasing.

I have formulated

Conjecture 4.1 and provided simulations to support it.

- Supposing that the value of the actual forgery quality (q_{act}) falls in the interval $(1; q_{max})$, there is an ideal value for estimated forgery quality (q_{est}) that minimizes the error resulting from the suboptimal choice of w.

Thesis III

- I have given a heuristic stroke extraction algorithm and provided experimental evidence to show that it is able to achieve comparable accuracy to the thinning algorithms with the best test results.
- I have proposed a heuristic algorithm for the solution of the multidimensional assignment problem for off-line signature verification.

I have defined a simplified version of the algorithm for matching a sample signature to the reference signatures.

- I have described a way of characterizing baseline information and have provided algorithms for the extraction of these features from signature images.
- I have described a way of characterizing loop information and provided algorithms for the extraction of these features from signature images.

7.2 Future work

Future work can be divided into two main categories. First, there are several smaller questions that we are confident we can answer with a reasonable amount of effort.

- Our current framework takes advantage of relatively few preprocessing steps. We have seen during our research that noise, or different parts of the signed form may be a source of error in further processing. We suspect that other transformations, like size or rotation normalization could also improve the accuracy of the matching algorithm. In the future we would like to investigate these claims more thoroughly.
- As feature extraction algorithms are mainly heuristic algorithms, they make mistakes often. On the other hand, by testing them on large and heterogeneous datasets, a large part of these errors can be systematically eliminated. The improvement of our existing feature extraction algorithms is currently one of our top priorities.

There are also two major questions that will probably require much more investigation.

- One of the major presumptions of our current classification algorithm is the independence of features. As we have seen, as the number of features increases, their interdependency grows, which reduces the accuracy of our model for larger feature counts. To tackle this, either new feature types – mostly independent from the currently analyzed ones – should be introduced, or detected feature properties should be transformed to less but more independent values. We have already done some research in this field, but our current results are not enough to draw a clear conclusion.
- The correct reconstruction of strokes could be a second major step towards an accurate off-line signature verifier. As we have seen, state of the art thinning algorithms cannot cope with overlapping strokes. If this problem could be resolved, the matching problem could undergo a major simplification together with several feature extraction algorithms. There are several promising attempts to do this [46] [75], but they still have a long way to go.

In the literature, the most important (and usually only) parameter used to characterize a verification system is the error rate it can achieve. Although the improvement of this parameter is really the main goal of signature verification,

it should be pointed out that this error rate is actually an aggregation of errors committed in previous processing steps. Hopefully, the achievements of this thesis will move the focus of researchers – at least to an infinitesimal extent – from the final results towards the more thorough analysis of individual processing steps and thereby help us better understand the mathematics of off-line signature verification.

Appendix A Detailed proofs

Detailed proof of

A.1 Proposition 3.2

$$f_o(x) = f_f(x)$$

$$\frac{1}{\sqrt{2\pi s_o^2}} e^{-\frac{1}{2}\left(\frac{m-x}{s_o}\right)^2} = \frac{1}{\sqrt{2\pi s_f^2}} e^{-\frac{1}{2}\left(\frac{m-x}{s_f}\right)^2}$$

$$\frac{1}{s\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{m-(m+ws)}{s}\right)^2} = \frac{1}{qs\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{m-(m+ws)}{qs}\right)^2}$$

$$e^{-\frac{1}{2}w^2} = \frac{1}{q} e^{-\frac{1}{2}\left(\frac{w}{q}\right)^2}$$

$$q = e^{\frac{1}{2}\left(w^2 - \frac{w^2}{q^2}\right)}$$

$$\ln q = \frac{w^2}{2}\left(\frac{q^2-1}{q^2}\right)$$

$$w = \pm\sqrt{2\frac{(q^2)}{q^2-1}\ln q} = \pm\sqrt{2}\sqrt{\ln q}\sqrt{\frac{q^2}{q^2-1}}$$

A.2 Detailed proof of Proposition 3.3

$$t_o(x) = t_f(x)$$

$$\frac{1}{\hat{s}_n\sqrt{1+\frac{1}{n}}}t_{n-1}\left(\frac{w\hat{s}_n}{\hat{s}_n\sqrt{1+\frac{1}{n}}}\right) = \frac{1}{q\hat{s}_n\sqrt{1+\frac{1}{n}}}t_{n-1}\left(\frac{w\hat{s}_n}{q\hat{s}_n\sqrt{1+\frac{1}{n}}}\right)$$

$$t_{n-1}\left(\frac{w}{\sqrt{1+\frac{1}{n}}}\right) = \frac{1}{q}t_{n-1}\left(\frac{w}{q\sqrt{1+\frac{1}{n}}}\right)$$

$$\frac{1}{\sqrt{n\pi}} * \frac{\Gamma\left(\frac{n}{2}\right)}{\Gamma\left(\frac{n-1}{2}\right)} * \left(1+\frac{\frac{w^2}{1+\frac{1}{n}}}{n-1}\right)^{-\frac{1}{2}(n)}$$

$$= \frac{1}{q} * \frac{1}{\sqrt{n\pi}} * \frac{\Gamma\left(\frac{n}{2}\right)}{\Gamma\left(\frac{n-1}{2}\right)} * \left(1+\frac{\frac{w^2}{q^2\left(1+\frac{1}{n}\right)}}{n-1}\right)^{-\frac{1}{2}(n)}$$

$$\left(1+\frac{\frac{w^2}{1+\frac{1}{n}}}{n-1}\right)^{-\frac{n}{2}} = \frac{1}{q}\left(1+\frac{\frac{w^2}{q^2\left(1+\frac{1}{n}\right)}}{n-1}\right)^{-\frac{n}{2}}$$

$$\left(1+\frac{\frac{w^2}{1+\frac{1}{n}}}{n-1}\right)^{n} = q^2\left(1+\frac{\frac{w^2}{q^2\left(1+\frac{1}{n}\right)}}{n-1}\right)^{n}$$

$$1+\frac{nw^2}{n^2-1} = q^{\frac{2}{n}}\left(1+\frac{1}{q^2}\frac{nw^2}{n^2-1}\right)$$

$$1+\frac{nw^2}{n^2-1} = q^{\frac{2}{n}} + q^{\frac{2}{n}}\frac{1}{q^2}\frac{nw^2}{n^2-1}$$

$$1-q^{\frac{2}{n}} = q^{\frac{2}{n}}\frac{1}{q^2}\frac{nw^2}{n^2-1} - \frac{nw^2}{n^2-1}$$

$$w = \pm\sqrt{\frac{1-q^{\frac{2}{n}}}{q^{\frac{2}{n}}\frac{1}{q^2}\frac{n}{n^2-1} - \frac{n}{n^2-1}}} = \pm\sqrt{\left(\frac{q^2-q^{2+\frac{2}{n}}}{q^{\frac{2}{n}}-q^2}\right)\left(\frac{n^2-1}{n}\right)}$$

A.3 The calculation of α and β based on Proposition 3.3

$$t_o(x) = \frac{1}{\hat{s}_n\sqrt{1+\frac{1}{n}}}\frac{1}{\sqrt{\pi(n-1)}}\frac{\Gamma\left(\frac{n}{2}\right)}{\Gamma\left(\frac{n-1}{2}\right)}\left(1+\frac{\left(\frac{x-\hat{m}_n}{\hat{s}_n\sqrt{1+\frac{1}{n}}}\right)^2}{n-1}\right)^{-\frac{n}{2}}$$

$$t_f(x) = \frac{1}{q\hat{s}_n\sqrt{1+\frac{1}{n}}}\frac{1}{\sqrt{\pi(n-1)}}\frac{\Gamma\left(\frac{n}{2}\right)}{\Gamma\left(\frac{n-1}{2}\right)}\left(1+\frac{\left(\frac{x-\hat{m}_n}{q\hat{s}_n\sqrt{1+\frac{1}{n}}}\right)^2}{n-1}\right)^{-\frac{n}{2}}$$

$$\alpha = P(m-ws>X)+P(m+ws<X) = 2\mathrm{T}_o(\hat{\mathrm{m}}_\mathrm{n}-w\hat{s}_n)$$
$$= 2\int_{-\infty}^{\hat{m}_n-w\,\hat{s}_n} t_o(x)dx = \frac{2}{\hat{s}_n\sqrt{1+\frac{1}{n}}}\frac{1}{\sqrt{\pi(n-1)}}\frac{\Gamma\left(\frac{n}{2}\right)}{\Gamma\left(\frac{n-1}{2}\right)}$$
$$\int_{-\infty}^{\hat{m}_n-w\,\hat{s}_n}\left(1+\frac{\left(\frac{x-\hat{m}_n}{\hat{s}_n\sqrt{1+\frac{1}{n}}}\right)^2}{n-1}\right)^{-\frac{n}{2}}dx$$

$$\beta = P(m-ws<X<m+w\sigma_o) = 1-2\mathrm{T}_\mathrm{f}(\hat{\mathrm{m}}_\mathrm{n}-w\hat{s}_n)$$
$$= 1-2\int_{-\infty}^{\hat{m}_n-w\,\hat{s}_n} t_f(x)dx$$
$$= 1-2\frac{1}{q\hat{s}_n\sqrt{1+\frac{1}{n}}}\frac{1}{\sqrt{\pi(n-1)}}\frac{\Gamma\left(\frac{n}{2}\right)}{\Gamma\left(\frac{n-1}{2}\right)}$$

$$\int_{-\infty}^{\hat{m}_n - w\,\hat{s}_n} \left(1 + \frac{\left(\dfrac{x - \hat{m}_n}{q\hat{s}_n\sqrt{1 + \dfrac{1}{n}}} \right)^2}{n - 1} \right)^{-\frac{n}{2}} dx$$

A.4 Detailed proof of Proposition 3.4

$$P(\theta_f|I) = P(\theta_o \,|\, I)$$

$$\frac{(1-\beta)^l \beta^{k-l}}{(1-\beta)^l \beta^{k-l} + \alpha^l (1-\alpha)^{k-l}} = \frac{\alpha^l (1-\alpha)^{k-l}}{(1-\beta)^l \beta^{k-l} + \alpha^l (1-\alpha)^{k-l}}$$

$$(1-\beta)^l \beta^{k-l} = \alpha^l (1-\alpha)^{k-l}$$

$$\frac{(1-\beta)^l}{\alpha^l} = \frac{(1-\alpha)^{k-l}}{\beta^{k-l}}$$

$$\frac{(1-\beta)^l}{\alpha^l} = \frac{\frac{(1-\alpha)^k}{\beta^k}}{\frac{(1-\alpha)^l}{\beta^l}}$$

$$\frac{(1-\beta)^l (1-\alpha)^l}{\alpha^l \beta^l} = \frac{(1-\alpha)^k}{\beta^k}$$

$$l = k \log_{\frac{(1-\alpha)(1-\beta)}{\alpha\beta}} \frac{1-\alpha}{\beta} = k \frac{ln \frac{1-\alpha}{\beta}}{ln \frac{(1-\alpha)(1-\beta)}{\alpha\beta}}$$
$$= k \frac{ln(1-\alpha) - ln(\beta)}{ln(1-\alpha) - ln(\beta) + ln(1-\beta) - ln(\alpha)}$$

A.5 Detailed proof of Proposition 4.1

Details of 4.16

$$AER(k) = \frac{\sum_{j=0}^{\lfloor l \rfloor} \binom{k}{j} \beta^{k-j}(1-\beta)^j + \sum_{i=\lfloor l \rfloor+1}^{k} \binom{k}{i} \alpha^i (1-\alpha)^{k-i}}{2}$$

$$AER(k+1) = \\ = \frac{\sum_{j=0}^{\lfloor l \rfloor} \binom{k+1}{j} \beta^{k+1-j}(1-\beta)^j + \sum_{i=\lfloor l \rfloor+1}^{k+1} \binom{k+1}{i} \alpha^i (1-\alpha)^{k+1-i}}{2}$$

Step 1:

$$AER(k) > AER(k+1)$$

Step 2:

$$\frac{\sum_{j=0}^{\lfloor l \rfloor} \binom{k}{j} \beta^{k-j}(1-\beta)^j + \sum_{i=\lfloor l \rfloor+1}^{k} \binom{k}{i} \alpha^i (1-\alpha)^{k-i}}{2} \\ > \frac{\sum_{j=0}^{\lfloor l \rfloor} \binom{k+1}{j} \beta^{k+1-j}(1-\beta)^j + \sum_{i=\lfloor l \rfloor+1}^{k+1} \binom{k+1}{i} \alpha^i (1-\alpha)^{k+1-i}}{2}$$

Step 3:

$$\sum_{j=0}^{\lfloor l \rfloor} \binom{k}{j} \beta^{k-j}(1-\beta)^j - \sum_{j=0}^{\lfloor l \rfloor} \binom{k+1}{j} \beta^{k+1-j}(1-\beta)^j + \\ > \sum_{i=\lfloor l \rfloor+1}^{k+1} \binom{k+1}{i} \alpha^i (1-\alpha)^{k+1-i} - \sum_{i=\lfloor l \rfloor+1}^{k} \binom{k}{i} \alpha^i (1-\alpha)^{k-i}$$

First, we calculate the sum for β:

$$\sum_{j=0}^{l} \binom{k}{j} \beta^{k-j}(1-\beta)^j - \sum_{j=0}^{l} \binom{k+1}{j} \beta^{k+1-j}(1-\beta)^j =$$

$$\binom{k}{0} \beta^{k-0}(1-\beta)^0 + \binom{k}{1} \beta^{k-1}(1-\beta)^1 + \binom{k}{2} \beta^{k-2}(1-\beta)^2 \ldots \\ + \binom{k}{l} \beta^{k-l}(1-\beta)^l$$

$$- \binom{k+1}{0} \beta^{k+1-0}(1-\beta)^0 - \binom{k+1}{1} \beta^{k+1-1}(1-\beta)^1 \\ - \binom{k+1}{2} \beta^{k+1-2}(1-\beta)^2 - \ldots \binom{k+1}{l} \beta^{k+1-l}(1-\beta)^l$$

Coefficients for β^{k+1}:

$$-\binom{k+1}{0}+\binom{k+1}{1}-\binom{k+1}{2}+\binom{k+1}{3}\ldots-(-1)^l\binom{k+1}{l}=$$

$$-(-1)^l\binom{k}{l}$$

Coefficients for β^k:

$$\binom{k}{0}-\binom{k}{1}+\binom{k}{2}-\binom{k}{3}\ldots+(-1)^l\binom{k}{l}$$

$$+0-\binom{k+1}{1}+\binom{k+1}{2}-\binom{k+1}{3}\ldots+(-1)^l\binom{k}{l}=$$

$$(-1)^l\binom{k}{l}\binom{l+1}{1}$$

Coefficients for β^{k-1}: $-(-1)^l\binom{k}{l}\binom{l+1}{2}$

Coefficients for β^{k-2}: $(-1)^l\binom{k}{l}\binom{l+1}{3}$

Coefficients for β^{k-3}: $-(-1)^l\binom{k}{l}\binom{l+1}{4}$

…

From the above coefficients:

$$\sum_{j=0}^{l}\binom{k}{j}\beta^{k-j}(1-\beta)^j-\sum_{j=0}^{l}\binom{k+1}{j}\beta^{k+1-j}(1-\beta)^j=\binom{k}{l}\beta^{k-l}(1-\beta)^l$$

Analogously:

Coefficients for α^{k+1}: $(-1)^l\binom{k}{l}$

Coefficients for α^k: $-(-1)^l\binom{k}{l}\binom{k-l}{1}$

Coefficients for α^{k-1}: $(-1)^l\binom{k}{l}\binom{k-l}{2}$

Coefficients for α^{k-2}: $-(-1)^l\binom{k}{l}\binom{k-l}{3}$

…

From the above coefficients:

$$\sum_{i=l+1}^{k+1}\binom{k+1}{i}\alpha^i(1-\alpha)^{k+1-i}-\sum_{i=l+1}^{k}\binom{k}{i}\alpha^i(1-\alpha)^{k-i}=\binom{k}{l}\alpha^{l+1}(1-\alpha)^{k-l}$$

Therefore the original equation can be written in this form:

$$\binom{k}{l}\beta^{k-l}(1-\beta)^{l+1} > \binom{k}{l}\alpha^{l+1}(1-\alpha)^{k-l}$$

$$(1-\beta)(\beta^{k-l}(1-\beta)^{l}) > \alpha(\alpha^{l}(1-\alpha)^{k-l})$$

$$\frac{1-\beta}{\alpha} > \frac{(\alpha^{l}(1-\alpha)^{k-l})}{(\beta^{k-l}(1-\beta)^{l})}$$

Details for equation 4.23:

Let

$$u = \log_{\frac{(1-\beta)(1-\alpha)}{\alpha\beta}} \frac{1-\alpha}{\beta}$$

$$\left(\frac{1-\beta}{\alpha}\right)^{\{l\}+u}\left(\frac{1-\alpha}{\beta}\right)^{\{l\}+u} < \left(\frac{1-\beta}{\alpha}\right)^{1}\left(\frac{1-\alpha}{\beta}\right)^{1}$$

$$\left(\frac{1-\beta}{\alpha}\right)^{\{l\}}\left(\frac{1-\alpha}{\beta}\right)^{\{l\}}\left(\frac{1-\beta}{\alpha}\right)^{u}\left(\frac{1-\alpha}{\beta}\right)^{u} < \left(\frac{1-\beta}{\alpha}\right)\left(\frac{1-\alpha}{\beta}\right)$$

$$\left(\frac{1-\beta}{\alpha}\right)^{\{l\}}\left(\frac{1-\alpha}{\beta}\right)^{\{l\}}\left(\frac{(1-\beta)(1-\alpha)}{\alpha\beta}\right)^{u} < \left(\frac{1-\beta}{\alpha}\right)\left(\frac{1-\alpha}{\beta}\right)$$

$$\left(\frac{1-\beta}{\alpha}\right)^{\{l\}}\left(\frac{1-\alpha}{\beta}\right)^{\{l\}}\left(\frac{(1-\beta)(1-\alpha)}{\alpha\beta}\right)^{\log_{\frac{(1-\beta)(1-\alpha)}{\alpha\beta}}\frac{1-\alpha}{\beta}} < \left(\frac{1-\beta}{\alpha}\right)\left(\frac{1-\alpha}{\beta}\right)$$

$$\left(\frac{1-\beta}{\alpha}\right)^{\{l\}}\left(\frac{1-\alpha}{\beta}\right)^{\{l\}}\left(\frac{1-\alpha}{\beta}\right) < \left(\frac{1-\beta}{\alpha}\right)\left(\frac{1-\alpha}{\beta}\right)$$

$$\left(\frac{1-\beta}{\alpha}\right)^{\{l\}}\left(\frac{1-\alpha}{\beta}\right)^{\{l\}} < \left(\frac{1-\beta}{\alpha}\right)$$

Details of 4.27:

$$\sum_{j=0}^{l}\binom{k}{j}\beta^{k-j}(1-\beta)^{j} - \sum_{j=0}^{l+1}\binom{k+1}{j}\beta^{k+1-j}(1-\beta)^{j} = -\binom{k}{l+1}\beta^{k-l}(1-\beta)^{l+1}$$

$$\sum_{i=l+2}^{k+1}\binom{k+1}{i}\alpha^{i}(1-\alpha)^{k+1-i} - \sum_{i=l+1}^{k}\binom{k}{i}\alpha^{i}(1-\alpha)^{k-i} = -\binom{k}{l+1}\alpha^{l+1}(1-\alpha)^{k-l}$$

$$-\binom{k}{l+1}\beta^{k-l}(1-\beta)^{l+1} \geq -\binom{k}{l+1}\alpha^{l+1}(1-\alpha)^{k-l}$$

$$\beta^{k-l}(1-\beta)^{l+1} \leq \alpha^{l+1}(1-\alpha)^{k-l}$$

$$\frac{1-\beta}{\alpha} \leq \frac{(\alpha^{l}(1-\alpha)^{k-l})}{(\beta^{k-l}(1-\beta)^{l})}$$

A.6 Mathematica scripts for Proposition 4.2

The cumulative distribution function for Student's t-distribution, with scale q and location 0.

```
T[x_, n_, q_] =
 1/2 + x/(q*Sqrt[1+1/n])*Gamma[(n)/2]*Hypergeometric2F1[1/2, (n)/2,
 3/2, -(x/(q*Sqrt[1 + 1/n]))^2/(n - 1)]/(Sqrt[Pi*(n - 1)]*Gamma[(n -
1)/2])
```

Decision threshold (Equation 3.26):

```
w[n_, q_] = Sqrt[(q^2 - q^(2 + 2/n))/(q^(2/n) - q^2)*(n^2 - 1)/n]
```

AER for a single feature (Equation 3.23):

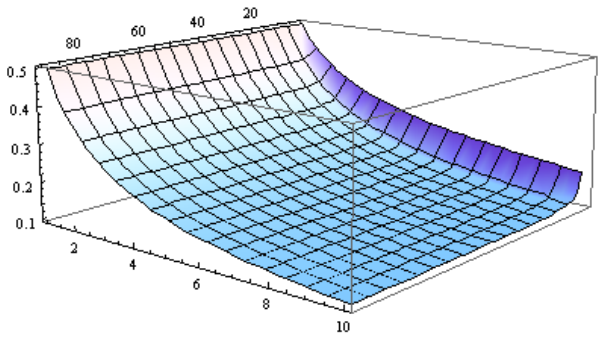
```
F[n_, q_] = (2 T[-w[n, q], n, 1] + 1 - 2 T[-w[n, q], n, q])/2
```

Partial derivative of AER ($\frac{\partial AER(q,n,1)}{\partial n}$):

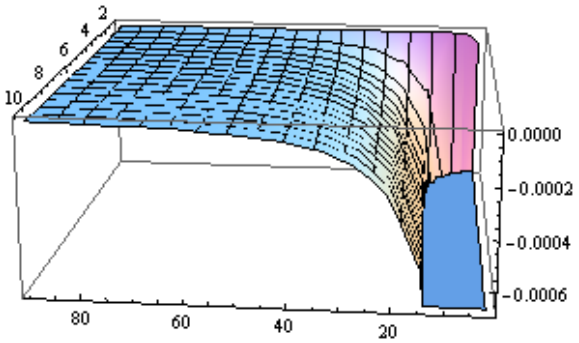
```
f[n_, q_] = D[F[n, q], n]
```

Output

```
Plot3D[F[n, q], {n, 2, 90}, {q, 1.001, 10}]
```




```
Plot3D[f[n, q], {n, 2, 90}, {q, 1.001, 10}]
```



Appendix B Developers' guide to the signature verification framework

This is a compact introduction to the main classes of the signature verification framework. You will need the following three classes to load signature metadata from a predefined database structure and to perform different operations on them:

- **Signer, Signature:** These are data classes that represent all metadata currently available about a signer or a signature. In most of the cases, you should not instantiate these classes directly, but use the methods of IStorage to load them.
- **FileSystemStorage:** Provides access to signatures stored in a predefined folder structure in the file system. This is currently the only available implementation of IStorage.

The framework is designed to allow a loose coupling of several independent processing modules. Based on the type of the module you want to implement, you have to choose between the following interfaces

- **ITransformation**: you should implement this interface if you want to make non-reversible changes on the signature image. The interface exposes a single method that should get the signature to process as a parameter. Every implementation should ensure that signature.SaveImage() is called before leaving the method.

  ```
  void Transform(Signature signature);
  ```

- **IFeatureExtraction**: you should implement this interface if you want to add information to signature metadata. The interface exposes a single method that should get the signature to process as a parameter. Changes are preserved in the signature object, the framework should take care of saving the information if required.

  ```
  void Transform(Signature signature);
  ```

- **IModelFeatureExtraction**: you should implement this interface if you want to add information to signer metadata. The interface exposes two methods. The model parameter contains all signer specific information, while the next parameter contains information about available reference signatures for the signer. The third parameter may contain an array of sample signatures. This is the main interface for implementing matching algorithms.

  ```
  void Process(SignatureModel model, Signature[] signatures);
  void Process(SignatureModel model, Signature[] originalSignatures,
               params Signature[] sampleSignatures);
  ```

- **IClassification**: the primary interface of all classification modules. The classification should be implemented in two phases. First, the Train method is called with the matched feature values of the reference

signatures. Then the Test method can be called multiple times with a feature vector obtained from the sample signature. The headers parameter may contain the names of the feature values allowing the creation of more detailed ClassificationResult structure.

```
FeatureHeader[] Train(double[,] featureValues);

FeatureHeader[] Train(double[,] values, FeatureHeader[] headers);

ClassificationResult Test(double[] featureValues);
```

The following sample demonstrates how the implementation of a simple classifier would look

```
public double Check(string[] originalSignatureImageFiles,
                    string   sampleSignatureImageFile)
{
    // Create data objects from images
    Signature[] originals = originalSignatureImageFiles
                  .Select(img => Signature.LoadFromFile(img)).ToArray();
    Signature sample = Signature.LoadFromFile(sampleSignatureImageFile);
    var all = originals.ToList();
    all.Add(sample);

    // Process all signatures
    foreach (Signature sig in all)
    {
        new BasicMetadataExtraction().Process(sig);
        new BaseLineExtraction().Process(sig);
        new LoopExtraction().Process(sig);
    }

    // Match baselines and loops
    var model = new SignatureModel();
    new BaseLineMatching().Process(model, originals.ToArray(), sample);
    new LoopMatching().Process(model, originals.ToArray(), sample);

    // Load the matched features for classification
    var loader = new FeatureLoader();
    double[,] features;
    FeatureHeader[] headers;
    loader.LoadFeatures(all.ToArray(), model, out features, out headers);

    // Classifier training with the reference signatures
    var classifier = new StatisticalClassifier();
    classifier.Train(
          features.GetPart(0,0,originalSignatureImageFiles.Length,
          features.GetLength(1)), headers);
    // Testing of sample signature
    var classificationResult = classifier.Test(
        features.GetRow(originalSignatureImageFiles.Length).ToArray());

    return classificationResult.MatchProbability;
}
```

Appendix C Signature databases

C.1 Samples from the SVC 2004 database

8. Bibliography

[1] G. Kaszab, A. Soóky, and J. I. Gulyás, *Hamis vagy Valódi - Írásszakértés és jogismeret mindenkinek*. Budapest: Grafodidakt Grafológusképző és Személyiségfejlesztő Központ, 2003.

[2] F. Bryan and R. Doug, "Documentation of Forensic Handwriting Comparison and Identification Method: A Modular Approach," *Journal of Forensic Document Examination*, vol. 12, pp. 1-68, 1999.

[3] B. Found and D. Rogers, "The initial profiling trial of a program to characterise forensic handwriting examiners skill," *Journal of the American Society of Questioned Document Examiners*, vol. 6, no. 2, pp. 483-492, 2003.

[4] K. A. Jain, "Signature Verification," *Michigan State University - Biometrics*, 2010.

[5] D. Impedovo and G. Pirlo, "Automatic Signature Verification: The State of the Art," *IEEE Transactions on Systems, Man, and Cybernetics - Part C: Applications and reviews*, vol. 38, no. 5, pp. 509-635, Sep. 2008.

[6] J. F. Vargas, M. A. Ferrera, C. M. Travieso, and J. B. Alonso, "Off-line signature verification based on grey level information using texture features," *Pattern Recognition*, vol. 44, no. 2, p. 375–385, Feb. 2011.

[7] L. Batista, E. Granger, and R. Sabourin, "Dynamic selection of generative–discriminative ensembles for off-line signature verification," *Pattern Recognition*, vol. 45, no. 4, p. 1326–1340, Apr. 2012.

[8] B. Kovari, B. Toth, and H. Charaf, "Classification Approaches in Off-Line Handwritten Signature Verification," *WSEAS Transactions on Mathematics*, vol. 8, no. 9, pp. 500-509, 2009.

[9] J. Neyman and E. S. Pearson, "The testing of statistical hypotheses in relation to probabilities a priori," *Joint Statistical Papers. Cambridge University Press*, pp. 186-202, 1967.

[10] M. Krause and H. F. Tipton, *Handbook of Information Security Management*. CRC Press LLC, 1993.

[11] R. A. Huber and A. M. Headrick, "Handwriting Identification: Facts and Fundamentals," *CRC Press, LCC*, 1999.

[12] D.-Y. Yeung, et al., "SVC2004: First International Signature Verification Competition," *Lecture Notes in Computer Science, Biometric Authentication, Volume 3072/2004*, pp. 16-22, 2004.

[13] J. Ortega-Garcia, J. Fierrez-Aguilar, J. Martin-Rello, and a. J. Gonzalez-Rodriguez, "Complete signal modeling and score normalization for function-based dynamic signature verification," *Lecture Notes in Computer Science, in Audio- and Video-Based Biometric Person*, vol. 2688, p. 658–667, 2003.

[14] L. Bovino, S. Impedovo, G. Pirlo, and L. Sarcinella, "Multi-expert verification of hand-written signatures," *Proc. 7th Int. Conf. Doc. Anal.*, p. 932–936, Aug. 2003.

[15] D. Muramatsu and T. Matsumoto, "Effectiveness of Pen Pressure,

Azimuth, and Altitude Features for Online Signature Verification," *Advances in Biometrics*, no. 4642, pp. 503-512, Aug. 2007.

[16] A. N. Abu-Rezq and A. S. Tolba, "Cooperative Self-Organizing Maps for Consistency Checking and Signature Verification," *Digital Signal Processing*, vol. 9, pp. 107-119, 1999.

[17] L. P. Cordella, P. Foggia, C. Sansone, F. Tortorella, and M. Vento, "A Cascaded Multiple Expert System for Verification," *Lecture Notes in Computer Science, Multiple Classifier Systems*, vol. 1857, pp. 330-339, 2000.

[18] F. Leerle and R. Palmond, "Automatic Signature Verification - The State of the Art 1989-1993," *Int'l Pattern Recognition and Artificial Intelligence, special issue signature verification*, vol. 8, no. 3, pp. 643-660, 1994.

[19] R. Plamondon and G. Lorette, "Automatic Signature Verification and Writer Identification - The State of the Art," *Pattern Recognition*, vol. 22, no. 2, pp. 107-131, 1989.

[20] R. Plamondon and S. N. Sargur, "On-Line and Off-Line Handwriting Recognition: A Comprehensive Survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 1, pp. 63-80, 2000.

[21] "Pattern Recognition, special issue on automatic signature verification," vol. 8, no. 3, Jun. 1994.

[22] D. Bertolini, L. S. Oliveirab, E. Justinoa, and R. Sabourinc, "Reducing forgeries in writer-independent off-line signature verification through ensemble of classifiers," *Pattern Recognition*, vol. 43, no. 1, pp. 387-396, 2010.

[23] D. Rivard, E. Granger, and R. Sabourin, "Multi-feature extraction and selection in writer-independent off-line signature verification," *International Journal on Document Analysis and Recognition*, pp. 1-21, Nov. 2011.

[24] J. Coetzer, B. M. Herbst, and J. A. d. Preez, "Off-Line Signature Verification: A Comparison between Human and Machine Performance," *Tenth International Workshop on Frontiers in Handwriting Recognition*, 2006.

[25] G. Fielding, K. Gummadidala, M. Kam, and R. Conn, "Signature Authentication by Forensic Document Examiners," *Journal of Forensic Sciences*, vol. 46, no. 4, Jul. 2001.

[26] J. Fierrez and J. Ortega-Garcia, "On-Line Signature Verification," in *Handbook of Biometrics*. Springer US, 2007, pp. 189-209.

[27] B. Kovari, "Time-Efficient Stroke Extraction Method for Handwritten Signatures," in *ACS07, The 7th WSEAS International Conference on Applied Computer Science*, 2007, pp. 157-161.

[28] E. Frias-Martinez, A. Sanchez, and J. Velez, "Support vector machines versus multi-layer perceptrons for efficient off-line signature recognition," *Engineering Applications of Artificial Intelligence*, no. 19, p. 693–704, 2006.

[29] S. Hangai, S. Yamanaka, and T. Hamamoto, "Writer Verification using Altitude and Direction of Pen Movement," in *IEEE, Proceedings of the*

International Conference on Pattern Recognition (ICPR'00), 2000.

[30] M. E. Munich and P. Perona, "Visual Identification by Signature Tracking," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 25, no. 2, Feb. 2003.

[31] L. Feher, "Aláírás rekonstrukció automatizált hitelesítéshez (Signature reconstruction for automated verification)," *Master's thesis*, 2008, Supervisor: Bence Kovari.

[32] S.-H. Kim, K.-S. Oh, and H.-I. Choi, "Off-line Verification System of the Handwrite Signature or Text using a Dynamic Programming," *ICCSA 2007, LNCS 4705, Part 1*, pp. 1014-1023, 2007.

[33] G. F. Russell and A. B. Jianying Hu, "Dynamic Signature Verification Using Discriminative Training," in *Proceedings of the 2005 Eight International Conference on Document Analysis and Recognition (ICDAR'05)*, 2005.

[34] W. K. Pratt, *Digital Image Processing: PIKS Scientific Inside*. Wiley-Interscience, 2007.

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

[36] V. E. Ramesh and M. N. Murty, "Off-line signature verification using genetically optimized weighted features," *Pattern Recognition*, no. 32, pp. 217-233, 1999.

[37] A. Lumini and L. Nanni, "Over-complete feature generation and feature selection for biometry," *Expert Systems with Applications*, 2007.

[38] K. Huang and H. Yan, "Off-line signature verification using structural feature correspondence," *Pattern Recognition*, no. 35, p. 2467–2477, 2002.

[39] J. Mahmud and C. M. Rahman, "On the power of feature analyser for signature verification," *Proceedings of the Digital Image Computing, Techniques and Applications*, 2005.

[40] S. N. Srihari, A. Xu, and M. K. Kalera, "Learning Strategies and Classification Methods for Off-line Signature Verification," *Proceedings of the 9th Int'l Workshop on Frontiers in Handwriting Recognition (IWFHR-9 2004)*, 2004.

[41] J. Fierrez-Aguilar, S. Krawczyk, J. Ortega-Garcia, and A. K. Jain, "Fusion of Local and Regional Approaches for On-Line Signature Verification," *IWBRS 2005, LNCS 3781*, p. 188–196, 2005.

[42] N. A. Murshed, F. Bortolozzi, and R. Sabourin, "Off-line signature verification, without a priori knowledge of class ω2 A new approach," *Third International Conference on Document Analysis and Recognition (ICDAR'95)*, vol. 1, p. 191.

[43] L. L. Lee and M. G. Lizarraga, "An Off-Line Method for Human Signature Verification," *13th International Conference on Pattern Recognition (ICPR'96)*, vol. 3, p. 195, 1996.

[44] R. Sabourin, J. P. Drouhard, and E. S. Wah, "Shape matrices as mixed shape factor for off-line signature verification," *Fourth International*

Conference on Document Analysis and Machine Intelligence, vol. 22, no. 9, pp. 938-949, Sep. 2000.

[45] Y. Mizukami, M. Yoshimura, H. Miike, and I. Yoshimura, "An Off-line Signature Verification System Using an Extracted Displacement Function," *Fifth International Conference on Document Analysis and Recognition (ICDAR'99)*, p. 757, 1999.

[46] J. L. Camino, C. M. Travieso, M. R. Ciro, and M. A. Ferrer, "Signature Classification by hidden Markov Model," *IEEE*, vol. 5, 1999.

[47] E. J. R. Justino, A. Yacoubi, F. Ortolozzi, and R. Sabourin, "An Off-Line Signature Verification System Using Hidden Markov Model and Cross-Validation," *XIII Brizilian Symposium on Computer Graphics and Image Processing (SIBGRAPI'00)*, p. 105, 2000.

[48] J. K. Guo, D. Doermann, and A. Rosenfeld, "Off-line skilled forgery detection using stroke and sub-stroke properties," *Proceedings of the International Conference on Pattern Recognition*, 2000.

[49] E. J. R. Justion, F. Bortolozzi, and R. Sabourin, "The Interpersonal and Intrapersonal Variability Influences on Off-Line Signature Verification Using HMM," *Proceedings of the XV Brazilian Symposium on Computer Graphics and Image Processing (SIBGRAPI'02)*, 2002.

[50] B. Fang, "Tracking of Feature and Stroke Positions For Off-line Signature Verification," *International Conference on Image Processing*, pp. 965-968, 2002.

[51] K. Ueda, "Investigation of Off-Line Japanese Signature Verification Using a Pattern Matching," *Proceedings of the Seventh International Conference on Document Analysis and Recognition (ICDAR'03)*, 2003.

[52] L. E. Martinez, C. M. Travieso, J. B. Alonso, and M. A. Ferrer, "Parametrization of a Forgery Handwritten Signature Verification System Using SVM," *IEEE 38rd Annual 2004 International Carnahan Conference on Security Technology*, pp. 193-196, 2004.

[53] M. A. Ferrer, J. B. Alonso, and C. M. Travisco, "Offline geometric parameters for automatic signature verification using fixed point arithmetic," *IEEE Transactions on Pattern Analyisis and Machine Intelligence*, vol. 27, no. 6, pp. 993-997, Jun. 2005.

[54] S. Chen and S. Srihari, "Use of Exterior Contours and Shape Features in Off-line Signature Verification," *Proceedings of the 2005 Eight International Conference on Document Analysis and Recognition (ICDAR'05)*, 2005.

[55] S. Armand, M. Blumenstein, and V. Muthukkumarasamy, "Off-line Signature Verification Based on the Modified Direction Feature," *The 18th International Conference on Pattern Recognition (ICPR'06)*, 2006.

[56] V. Nguyen, Y. Kawazoey, T. Wakabayashiy, U. Palz, and M. Blumenstein, "Performance Analysis of the Gradient Feature and the Modified Direction Feature for Off-line Signature Verification," *12th International Conference on Frontiers in Handwriting Recognition*, pp. 303-307, 2010.

[57] M. Blumenstein, M. A. Ferrer, and J. F. Vargas, "The 4NSigComp2010 off-line signature verification competition: Scenario 2," *12th International Conference on Frontiers in Handwriting Recognition*, pp. 721-726, 2010.

[58] H. Lei and V. Govindaraju, "A comparative study on the consistency of features in on-line signature verification," *Pattern Recognition Letters*, vol. 26, p. 2483–2489, 2005.

[59] B. Kovari, I. Albert, and H. Charaf, "A General Representation for Modeling and Benchmarking Off-line Signature Verifiers," *Computational Engineering in Systems Applications*, pp. 72-76, 2008.

[60] B. Kovari. (2006) Signature Database for Off-line Signature Verification. [Online]. http://www.aut.bme.hu/signature/

[61] F. Vargas, M. A. Ferrer, C. M. Travieso, and J. B.Alonso, "Off-line Handwritten Signature GPDS-960 Corpus," *IAPR 9th International Conference on Document Analysis and Recognition*, pp. 764-768, Sep. 2007.

[62] J. Fierrez-Aguilar, N. Alonso-Hermira, G. Moreno-Marquez, and J. Ortega-Garcia, "An off-line signature verification system based on fusion of local and global information," *LNCS, Workshop on Biometric Authentication*, pp. 298-306, 2004.

[63] H. Srinivasan, S. N.Srihari, and M. J. Beal, "Signature verification using Kolmogorov-Smirnov statistic," *International Graphonomics Society Conference (IGS)*, pp. 152-156, Jun. 2005.

[64] J. Fàbregas and M. Faundez-Zanuy, "Biometric dispersion matcher," *Pattern Recognition*, vol. 41, pp. 3412-3426, 2008.

[65] A. K. Jain, A. Ross, and S. Prabhakar, "An introduction to biometric recognition," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 14, no. 1, pp. 4-20, Jan. 2004.

[66] H. Lilliefors, "On the Kolmogorov-Smirnov test for normality with mean and variance unknown," *Journal of the American Statistical Association*, vol. 62, no. 318, pp. 399-402, Jun. 1964.

[67] S. S. Shapiro and M. B. Wilk, "An Analysis of Variance Test for the Exponential Distribution (Complete Samples)," *Technometrics*, vol. 14, pp. 355-370, May 1972.

[68] (2012) Mathematica 8.0. [Online]. http://www.wolfram.com/mathematica/

[69] C. Illes, "Off-line aláírás-hitelesítő algoritmusok (Off-line signature verification algorithms)," *Master's thesis*, 2007.

[70] A. Aishy, "New binary morphological operations for effective low-cost boundary detection," *International journal of pattern recognition and artificial intelligence*, vol. 17, no. 2, pp. 1-13, 2002.

[71] L. Lam, S.-W. Lee, and C. Y. Suen, "Thinning Methodologies-A Comprehensive Survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 14, no. 9, p. 879, Sep. 1992.

[72] M. Couprie, "Note on fifteen 2D parallel thinning algorithms," *IGM2006-01*, 2006.

[73] D. Dori and W. Liu, "Sparse Pixel Vectorization: An Algorithm and Its Performance Evaluation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 21, no. 3, pp. 202-215, Mar. 1999.

[74] J. Peng, "An efficient algorithm of thinning scanned pencil drawings," *Journal of Image and Graphics*, vol. 5, no. 5, pp. 434-439, 2000.

[75] B. Kovari, A. Horvath, Z. Kertesz, and C. Illes, "Off-Line Signature Verification - Comparison of Stroke Extraction Methods," in *ICSOFT, 2nd International Conference on Software and Data Technologies*, Barcelona, 2007, pp. 270-276.

[76] K. K. Lau, P. C. Yuen, and Y. Y. Tang, "Recovery of Writing Sequence of Static Images of Handwriting using UWM," *Proceedings of the Seventh International Conference on Document Analysis and Recognition (ICDAR'03)*, 2003.

[77] B. Talko, "Extracting stroke information off-line for cursive handwriting recognition," *Honours Report, University of Melbourne*, 1995.

[78] A. Pérez-Hernández, A. Sánchez, and J. F. Vélez, "Simplified Stroke-based Approach for Off-line Signature Recognition," *Second COST 275 Workshop*, 2004.

[79] Y. M. Su and J. F. Wang, "A learning process to the identification of feature points on Chinese characters," *IEEE Transactions on Systems, Man, and Cybernetics, Part A: Systems and Humans*, vol. 33, no. 3, May 2003.

[80] R. Cao and C. L. Tan, "A Model of Stroke Extraction from Chinese Character Images," *15th International Conference on Pattern Recognition (ICPR'00)*, vol. 4, p. 4368, 2000.

[81] I. .-J. Kim, C. .-L. Liu, and J. .-H. Kim, "Stroke-guided Pixel Matching for Handwritten Chinese Character Recognition," *Proceedings of the Fifth International Conference on Document Analysis and Recognition*, 1999.

[82] K. Liu, Y. S. Huang, and C. Y. Suen, "Identification of Fork Points on the Skeletons of Handwritten Chinese Characters," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 21, no. 10, Oct. 1999.

[83] Y. Qiao, M. Nishiara, and M. Yasuhara, "A Framework Toward Restoration of Writing Order from Single-Stroked Handwriting Image," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28, no. 11, Nov. 2006.

[84] V. Pervouchine, G. Leedham, and K. Melikhov, "Three-stage Handwriting Stroke Extraction Method with Hidden Loop Recovery," *Proceedings of the 2005 Eight International Conference on Document Analysis and Recognition (ICDAR'05)* , 2005.

[85] J. C. Rush, *The Image Processing Handbook, Fifth edition*. North Carolina State University, 2006.

[86] K. K. Lau, P. C. Yuen, and Y. Y. Tang, "Stroke Extraction and Stroke Sequence Estimation On Signatures," *16th International Conference on Pattern Recognition, Proceedings*, pp. 119-122, 2002.

[87] S. Lee and J. C. Pan, "Offline tracing and representation of signatures," *JC PAN IEEE transactions on systems, man, and cybernetics*, pp. 755-771, 2005.

[88] Y. Kato and M. Yasuhara, "Recovery of Drawing Order from Single-Stroke Handwriting Images," *Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 9, pp. 938-949, Sep. 2000.

[89] M. Vincze, "Vékonyító algoritmusok összehasonlító elemzése," *Bachelor's thesis*, 2010, Supervisor: Bence Kovari.

[90] B. Zitová and J. Flusser, "Image registration methods: a survey," *Image and Vision Computing 21*, pp. 977-1000, 2003.

[91] L. G. Brown, "A survey of image registration techniques," *ACM Computing Surveys*, vol. 24, pp. 326-376, 1992.

[92] J. Valyon and G. Horváth, "A hybrid intelligent system for image matching , used as preprocessing for signature verification," *5th International Conference on Artificial Neural Networks and Genetic Algorithms*, 2001.

[93] G. Gregory, G. Boris, and H. Jing, "Worst Case Analysis of Max-Regret, Greedy and Other Heuristics for Multidimensional Assignment and Traveling Salesman Problems," *Journal of Heuristics*, vol. 14, no. 2, pp. 169-181, Apr. 2008.

[94] H. W. Kuhn, "The Hungarian Method for the assignment problem," *Naval Research Logistics Quarterly*, vol. 2, pp. 83-97, 1955.

[95] ASTM, "E 2195: Standard Terminology Relating to the Examination of Questioned Documents," *Annual Book of ASTM Standards*, vol. 14.02, 2002.

[96] B. Nagy, "Automatizált aláíráshitelesítés alapvonalak összehasonlításával (Automatic signature verification using baseline comparison)," *Master's thesis*, 2010, Supervisor: Bence Kovari.

[97] J. Serra, "Image Analysis and Mathematical Morphology," 1982.

[98] M. K. Haragos, "Automatikus aláíráshitelesítés alakjellemzők összehasonlításával (Automated signature verification by comparing shape descriptors)," *Master's thesis*, 2009, Supervisor: Bence Kovari.

[99] Larman, *Applying UML and Patterns : An Introduction to Object-Oriented Analysis and Design and Iterative Development*. Prentice-Hall, 2004.

[100] T. Kökényesi, "Tollvonások rekonstruálása off-line aláíráshitelesítésben (Reconstruction of pen strokes in off-line signature verification)," *Bachelor's thesis*, 2010, Supervisor: Bence Kovari.

[101] Z. Albert, "Aláíráshitelesítés alapvonalak DTW alapú összehasonlításával," *TDK (Scientific Students' Associations Conference)*, 2009, Supervisor: Bence Kovari.

[102] Á. Horváth, "Automatikus osztályozási módszerek az off-line aláíráshitelesítésben (Automatic classification methods in off-line signature verification)," *Master's thesis*, 2011, Supervisor: Bence Kovari.

[103] G. Kiss, "Vonas alapu algoritmusok fejlesztese off-line alairas-felismero rendszerhez (Development of stroke based algorithm for an off-line siganture verification system)," *Master's thesis*, 2008, Supervisor: Bence Kovari.

[104] Z. Márta, "Aláírások automatikus hitelesítése dinamikus írásjellemzők segítségével (Automatic signature verification using dynamic handwriting features)," *Bachelor's thesis*, 2010, Supervisor: Bence Kovari.

[105] Z. Albert, "Aláíráshitelesítés burkológörbék összehasonlításával (Siganture verification by comparing upper and lower envelopes)," *Master's thesis*, 2010, Supervisor: Bence Kovari.

[106] Z. Berceli, "Írásjellemzők automatikus párosítása az aláíráshitelesítésben

(Automated feature matching in signature verification)," *Master's thesis*, 2012, Supervisor: Bence Kovari.

[107] B. Tóth, "Automatikus osztályozási módszerek az aláíráshitelesítésben (Automated classification methods in signature verification)," *Master's thesis*, 2009, Supervisor: Bence Kovari.

[108] A. Horvath, "Automatikus osztályozási módszerek (Autoamted classification methods)," *Bachelor's thesis*, 2010, Supervisor: Bence Kovari.

[109] B. Kutasi, "Moduláris WPF keretrendszer kézeredet azonosításhoz (A modular WPF framework for signature verification)," *Bachelor's thesis*, 2010, Supervisor: Bence Kovari.

[110] M. Pálfalvi, "Bachelor's thesis," *Interaktív aláíráshitelesítési frontend Silverlight platformon (Interactive siganture verification frontend on Silverlight platform)*, 2012, Supervisor: Bence Kovari.

[111] D. D. Nagy, "Felhasználóbarát felületek WPF platformon (User friendly interfaces on WPF platform)," *Bachelor's thesis*, 2010, Supervisor: Bence Kovari.