

PhD THESIS

*Thesis submitted in partial fulfilment of the requirements
to obtain the title of*

PhD of Université Paris Descartes

UFR Mathématiques et Informatique

Classification of Handwritten Documents : Writer Recognition

by

Imran SIDDIQI

Defended on the 1st of December 2009 in front of the following jury:

Pr.	Thierry	PAQUET	Université de Rouen	Chair
Pr.	Robert	SABOURIN	Université du Québec	Reviewer
Pr.	Christian	VIARD-GAUDIN	Université de Nantes	Reviewer
Pr.	Hubert	EMPTOZ	Insa de Lyon	Examiner
Pr.	Laurent	WENDLING	Université Paris Descartes	Examiner
Pr.	Nicole	VINCENT	Université Paris Descartes	Supervisor

Abstract

The problem of identifying the writer of a handwritten document image has been an active research area over the last few years and enjoys applications in forensic and historical document analysis. We have developed an effective method for automatic writer identification and verification from unconstrained handwritten text images. Our method relies on two different aspects of writing: the presence of redundant patterns in the writing and its visual attributes. Based on the hypothesis that handwriting carries certain patterns that an individual would use frequently as he writes, we look to extract these patterns by analyzing small writing fragments and grouping similar patterns into clusters. In fact this corresponds more to the redundancy of writing gestures than writing shapes. These clusters are determined either for each of the writers separately or, for a group of writers generating a universal set of patterns. The writing in question is then compared to the produced clusters. We next exploit two important visual attributes of writing, the orientation and curvature, which enable to distinguish one writing from another. These attributes are extracted by computing a set of features from writing samples at different levels of observation. Two writings are then compared by computing distances between their respective features. Finally, we combine the two facets of handwriting to characterize the writer of a handwritten sample. The proposed methodology, evaluated on modern as well as ancient writings exhibited promising results on tasks of writer recognition and handwriting classification.

Acknowledgements

This thesis is the result of three years of devoted work which would not have been possible without the support of many. I would here like to express my thanks to people who have been very helpful to me during the course of this work.

First of all, I would like to take this opportunity to gratefully acknowledge the wholehearted supervision of **Pr. Nicole Vincent** during this work. Her dedication, skillful guidance, helpful suggestions and constant encouragement made it possible for me to deliver a dissertation of appreciable quality and standard.

I would also like to thank **Pr. Robert Sabourin** and **Pr. Christian Viard-Gaudin** for accepting to review my thesis and providing me with valuable comments and suggestions to improve the quality of this thesis.

Special thanks are due to the co-director of research group SIP, **Pr. Georges Stamon** for providing me with valuable guidance and support at various stages of my work.

My cordial appreciation extends to **Dr. Nicolas Lomenie** and **Dr. Florence Cloppet** for providing a good research environment and extending worthwhile support and constructive suggestions. My stay at SIP would not have been such a pleasurable one without the presence of friendly colleagues who helped me out in one way or another and exchanged fruitful views from time to time.

I owe my special thanks to my friends **Khurram, Shehzad** and **Hassan** for helping me get through the difficult times, and for all the support, camaraderie and caring they provided.

I would also like to express my deepest gratitude to the **Higher Education Commission** of Pakistan for financially supporting the study and making it possible for me to pursue my research in a prestigious institution of esteemed repute.

I am forever indebted to my **parents** for their patience and understanding, alleviating my family responsibilities and encouraging me to concentrate on my study.

Finally and most importantly, I would like to express special thanks to my **wife** for her support when it was most required. Without her help and encouragement, this study would not have been completed.

Table of Contents

Chapter 1 Introduction	1
Chapter 2 State of the art	7
2.1 Data Sets.....	8
2.1.1 IAM Data Set	8
2.1.2 RIMES Data Set	9
2.2 Text- Dependent Writer Recognition	10
2.3 Text-Independent Writer Recognition	13
2.3.1 Global Methods.....	13
2.3.2 Local Methods.....	19
2.3.3 Writer Recognition based on Handwriting Recognition	24
2.4 Discussion	25
Chapter 3 Writer Characterization: Redundant Patterns of Writing	29
3.1 Data Set	31
3.2 Document Image Binarization	32
3.3 Division of Handwriting.....	33
3.3.1 Regular Division	33
3.3.2 Horizontal Window Sliding	34
3.3.3 Adaptive Window Positioning	34
3.4 Clustering of sub-images.....	37
3.4.1 Representation of sub-images	38
3.4.2 (Dis) Similarity Measure	39
3.4.3 Clustering	40
3.4.4 Representative Clusters	42
3.5 Writing Representation	45
3.6 Writer Identification.....	46
3.6.1 Bayesian Classifier.....	47
3.6.2 Probability Distribution of Redundant Writing Patterns	48
3.7 Writer Verification	49
3.8 Redundant Writing Patterns: Universal Code Book.....	50
3.9 Experimental Results.....	52
3.9.1 Writer-specific code book	52
3.9.2 Universal Code book.....	54
3.9.3 Discussion	55
3.10 Conclusion.....	57
Chapter 4 Writer Characterization: Contour Based Features	59
4.1 Feature Extraction	60

4.1.1 Chain Code Based Features	61
4.1.2 Polygon Based Features	69
4.2 Writer Recognition.....	72
4.2.1 Writer Identification.....	73
4.2.2 Writer Verification	73
4.3 Experimental Results	73
4.3.1 Performance of Individual Features	74
4.3.2 Performance of Feature Combinations.....	76
4.4 Stability of Features	80
4.5 Combining Contour-based Features with Redundant Writing Patterns	82
4.6 Feature Selection.....	83
4.6.1 Genetic Algorithms	84
4.6.2 Analysis of Feature Relevance.....	85
4.7 Conclusion	92
Chapter 5 Applications.....	93
5.1 Classification of Ancient Manuscripts	93
5.1.1 Text Extraction.....	94
5.1.2 Document Binarization	95
5.1.3 Feature Extraction.....	96
5.1.4 Similarity based Incremental Grouping	96
5.1.5 Relevance Feedback.....	97
5.2 Writer Recognition on Arabic Documents.....	102
5.3 Signature Verification	103
5.4 Conclusion	105
Chapter 6 Conclusion and Perspectives.....	107
Appendix A Résumé en Français.....	109
Appendix B Results: Codebook Features	117
Appendix C Results: Contour Based Features.....	119
Appendix D Feature Selection Results	121
Appendix E Sample Images	122
Appendix F Graphem Retrieval Software: User Interface	127
Bibliography	128
Author's Publications	140

List of Figures

Figure 1.1 Copybook styles (a) United States (Zaner Bloser) (b) Chile (c) Germany	2
Figure 1.2 The evolution of writing from Roman square capitals through medieval Gothic to modern handwriting (Image reproduced from [Nickell, 2007]).....	2
Figure 1.3 Lindbergh kidnapping case (1932): Comparison between the known signature of the prime suspect (top) and the signature composed from individual letters in the ransom notes (bottom) was a significant investigation development.....	3
Figure 1.4 Copy book norms and individual deviations (Image: [Nickell & Fischer, 1999]).....	4
Figure 1.5 Factors causing handwriting variability. Image reproduced from [Schomaker, 1998].....	5
Figure 1.6 Writer Identification and Verification models.....	6
Figure 2.1 Distribution of samples per writer in the IAM data set.....	9
Figure 2.2 The morphological opening on the projection function: (a) initial histogram function of the word ‘characteristic’ (b) opening with line structuring element of length 3 (c) opening with line structuring element of length 7. (Image: [Zois & Anastassopoulos, 2000])	12
Figure 2.3 Extraction of the edge-based texture features on letter "a" (Image: [Bulacu & Schomaker, 2006])	15
Figure 2.4 Evolution Graph (Image: [Boulétreau et al., 1998])	16
Figure 2.5 Legibility Graph (Image: [Boulétreau et al., 1998])	17
Figure 2.6 An example manuscript and its Zipf plot (Image: [Pareti & Vincent, 2006])	18
Figure 2.7 Segmentation into graphemes (Image: [Bensefia et al., 2005b]).....	21
Figure 2.8 Samples of invariant clusters extracted from a handwritten page (Image: [Bensefia et al., 2002])	21
Figure 2.9 Segmentation into fraglets (Image: [Schomaker et al., 2004])	23
Figure 3.1 Code book based methods for writer recognition	30
Figure 3.2 Presence of similar loops in two different characters	30
Figure 3.3 Same pattern repeated across different characters	31
Figure 3.4 An overview of the proposed methodology.....	32
Figure 3.5 Regular division of writing and its drawback	33
Figure 3.6 Sliding windows horizontally over text	34
Figure 3.7 Adaptive Window Positioning.....	35
Figure 3.8 Comparison of window positioning algorithms.....	36
Figure 3.9 For the same text written by two writers number of windows as function of (a) number of words (b) window size (in pixels).....	37
Figure 3.10 Patterns translated before calculating the features.....	38
Figure 3.11 Number of clusters and the corresponding area of text pixels covered	43
Figure 3.12 Number of clusters for two (training & test) samples of writers	43
Figure 3.13 Clusters obtained on a document image	44

Figure 3.14 The first five most frequent classes (shown in Figure 3.13) after application of PCA and dimensionality reduced to 2	45
Figure 3.15 FAR and FRR on the distribution of distances	50
Figure 3.16 Examples of the invariant patterns obtained in studies: (a) [Bensefia et al., 2005b] and (b) [Bulacu & Schomaker, 2007] (Images reproduced from the cited references)	50
Figure 3.17 Code book of size 100 generated from 50 samples of the dataset RIMES.....	51
Figure 3.18 Probability distribution for two samples of the same writer when using a universal code book.....	52
Figure 3.19 Writer identification results for different clustering methods (150 writers).....	53
Figure 3.20 Writer identification performance (on 300 writers) as a function of (a) code book size and (b) the number of samples used to generate the code book.....	55
Figure 3.21 Identification rate as function of amount of text on 300 writers of IAM data set	57
Figure 4.1 An example of instinctively distinguishable writings	60
Figure 4.2 Images and their contours.....	60
Figure 4.3 Contours replaced by direction codes (Colours represent codes).....	62
Figure 4.4 Distribution of chain codes on writing samples from two writers.....	63
Figure 4.5 Angle θ_i at pixel p_i	63
Figure 4.6 The four possible L-junctions.....	64
Figure 4.7 Two possible chain code pairs, starting at each of the three pixels of an L-junction	64
Figure 4.8 Two writings and their respective distributions (normalized) of chain code pairs.....	65
Figure 4.9 a) Contour pixel p_j (with code c_j) of a character b) Backward histogram 'b' at p_j c) Forward histogram 'f' at p_j	66
Figure 4.10 a) Windows positioned over text b) Windows positioned over the contour image c) Directional distribution in a window	67
Figure 4.11 Contribution of a window to the stroke direction distribution.....	68
Figure 4.12 Normalized local stroke direction distribution of a writing.....	69
Figure 4.13 Polygonization at different values of T	69
Figure 4.14 Division of Slopes (-90° to 90°) into bins and the corresponding segment classes.....	70
Figure 4.15 Cumulative sum of the slope distribution for three writers (two samples each)	70
Figure 4.16 Curvature (Angle) between two connected segments.....	71
Figure 4.17 Writer identification search on the IAM data set.....	75
Figure 4.18 Distances (d1 & d2) to a query image	76
Figure 4.19 Performance of feature combinations on Writer Identification	78
Figure 4.20 ROC Curves for feature combinations	79
Figure 4.21 Writer identification performance for a weighted combination of f_5 and f_7	80
Figure 4.22 Writer Identification performance as function of (a) Number of writers (b) Amount of text.....	81

Figure 4.23 Key steps in Feature Selection.....	84
Figure 4.24 Features selected in ten runs of the GA	86
Figure 4.25 Frequencies of selected features	87
Figure 4.26 Feature components selected in 10 runs of the GA	89
Figure 4.27 Average number of components selected for each of the features.....	89
Figure 4.28 Feature selection distribution.....	91
Figure 4.29 Frequencies of selected features (reduced dimension)	92
Figure 5.1 Examples of non-text regions in the document images	94
Figure 5.2 Binarization Steps.....	96
Figure 5.3 An example of images retrieved on the Graphem dataset	97
Figure 5.4 Principle of a relevance feedback system	98
Figure 5.5 Retrieval results with and without feedback.....	100
Figure 5.6 Writing classes obtained with k-means for (a) k=3 (b) k=5	101
Figure 5.7 Classification performance (knn) on the Graphem dataset with k=3,5 and 7.....	101
Figure 5.8 Division of a word into sub-images and its polygon approximation	102
Figure 5.9 Identification rates and ROC curves on the IFN/ENIT dataset	103
Figure 5.10 Examples of signatures	103
Figure 5.11 Inertial axis of the signature and the redressed signature	104
Figure 5.12 Distribution of chain codes for 10 signatures of an individual.....	104
Figure 5.13 ROC curve on the signatures	105

List of Tables

Table 2-1 Comparison of writer identification performance	27
Table 3-1 The modified IAM data set.....	31
Table 3-2 Identification and verification performance based on writer specific code book (Clustering: Hierarchical, Classifier: χ^2 Distance)	54
Table 3-3 Identification and verification performance based on the universal code book	54
Table 3-4 Performance comparison of code book based methods.....	56
Table 4-1 Differential Chain Codes and the corresponding Angles	63
Table 4-2 Features and their dimensionalities	72
Table 4-3 Distance measures to compare two distributions.....	73
Table 4-4 Performance of individual features.....	74
Table 4-5 Performance of feature combinations.....	77
Table 4-6 Performance of combined features on the combined dataset (1025 writers).....	80
Table 4-7 Performance of combining contour based features and the code book distribution.....	82
Table 4-8 Performance comparison of writer identification systems on IAM dataset.....	83
Table 4-9 Division of features according to relevance	87
Table 4-10 Performance of selected feature subsets on 650 writers of the IAM data set.....	87
Table 4-11 Performance of selected feature components on 650 writers of IAM data set	90
Table 4-12 Features with reduced dimensions.....	91
Table 4-13 Division of features (reduced dimension) according to relevance	92
Table 5-1 Summary of neighbourhood based thresholding methods.....	95
Table 5-2 Writer recognition performance on the IFN/ENIT database	102
Table B-1 Writer identification rates (on 150 writers of IAM dataset) on different clustering methods for writer-specific and universal codebooks.....	117
Table B-2 Writer identification performance (on 300 writers of IAM dataset) as a function of codebook size.....	117
Table B-3 Writer identification performance (on 300 writers of IAM dataset) as a function of the number of samples (writers) used to generate the codebook	117
Table B-4 Writer identification rates (on 300 writers of IAM dataset) as function of amount of text	118

Chapter 1

Introduction

Handwriting is an essential means of communication in our civilization which has developed and evolved over time. At school, we all learn to write according to a standard writing style: *the copy book* that varies according to the geographical location, temporal circumstances and the cultural and historical backgrounds (Figure 1.1). With the passage of time, we develop individual writing characteristics and our handwriting starts to deviate from the learned style. These unique characteristics serve to distinguish an individual's writing from another's, even if the two writings share the same copy book, making it possible to identify the author for which one has already seen a written text. This automatic writer recognition serves as a valuable solution for the document examiners, palaeographers, graphologists and forensic experts.

In contrast to the electronic or printed text, the handwritten text carries additional information about the personality of the person who has written. The Chinese philosopher King Jo-Hau rightly declared: "*handwriting infallibly shows us whether it comes from a vulgar or a noble minded person*" [Olyanova, 1960]. In fact, among the expressive behaviours of human, handwriting carries the richest information to gain insight into the physical, mental, and emotional states of the writer. Each written movement or stroke reveals a specific personality trait, the neuro-muscular movement tendencies being correlated with specific observable personality features [Baggett, 2004]. Two personalities might share common characteristics but can never be exactly the same and so is the case with handwriting. This explains the stability in the writing style of an individual and the variability between the writings of different writers.

A handwritten text may present varied interests. The ancient manuscripts, for example, could serve to study the evolution of the style and form of writing over time that in turn reflects the historical and cultural changes of the society. Knowledge about individual letters, ligatures, punctuations and abbreviations and, the way they have evolved enables the palaeographers and historians identify the periods in which a manuscript was written (Figure 1.2). The quantity of these ancient manuscripts stored in archives, libraries and private collections is enormous and it will be useful to develop computer systems that could help the palaeographers in manuscripts dating, classification and authentication.

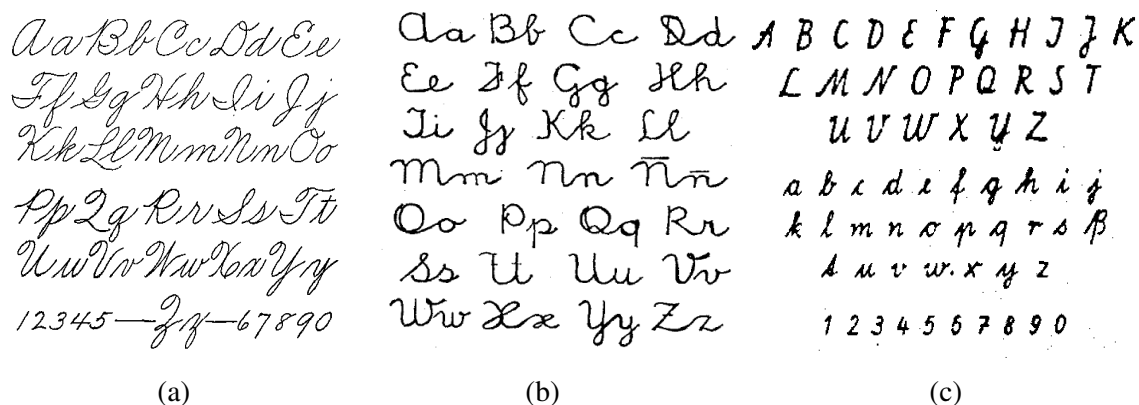


Figure 1.1 Copybook styles¹ (a) United States (Zaner-Bloser) (b) Chile (c) Germany

Handwriting also has an interesting relationship with neuroscience and several neurological disorders (that could affect the writing motor skills) are reflected in handwriting of the patient [Eaton, 1938] [Mergl et al., 2004] [Racine et al., 2008] [Ünlü et al., 2006]. Handwriting could therefore be of diagnostic value for the neurologists, particularly if they have previous writing samples of the patient. As an example, changes in the writing of a person could be analyzed for the diagnosis of the Parkinson’s disease [Ünlü et al., 2006] where the effects of the disorder are manifested in the writing of the patient even in the early stages. The writing of the patient tends to get smaller as he/she writes i.e. the writing is well formed and of normal size as the patients starts writing but as he/she writes across the page, it progressively gets smaller and smaller and at the end the letters might even be unreadable (micrographia) [Sage & Duvoisin, 2001]. The changing patterns in the writing could then be used to follow the progression of the disease.

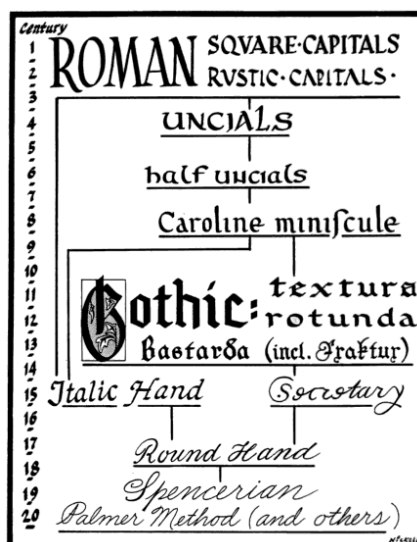


Figure 1.2 The evolution of writing from Roman square capitals through medieval Gothic to modern handwriting (Image reproduced from [Nickell, 2007])

¹ Source : www.handwriting.org

From the view point of graphology, handwriting is an insightful means of personality profiling, highlighting the character traits and tracking the feelings and emotions of a person. As a matter of fact, graphologists claim that handwriting could reveal more than two hundred personality traits including moods, temperaments, thinking patterns, fears, work drive, maturity, social interactions, and the list goes on. May be that is why, handwriting is also termed as *brainwriting*. Not only it reveals one's personality, but changing the way one writes, can result in changing the personality as well – the *grapho-therapy*, that involves identifying and eliminating negative strokes and letter formations from the writing and replacing them by the positive ones, thus eventually developing positive personality traits and getting rid of the unwanted ones.

Moving to the world of forensic analysis, handwriting, being an important component of behavioural biometrics, carries significant importance (Figure 1.3). The forensic documents examiners might need to identify the authorship or authenticity of a questioned document (e.g. a will, a ransom note, a threatening letter, etc.), verify signatures, identify forgeries, detect alterations or analyze indented writings. This analysis is a tiresome procedure for the questioned document examiners, thus developing computerized systems for handwriting analysis could serve as valuable tools in forensic document analysis. These systems take in the scanned document images and employ image processing and pattern classification techniques to solve a given problem. Of course, the results of these systems can not be accepted as evidence in a court case and the intervention of human experts is inevitable; nevertheless, they have still proved to be quite helpful in speeding up the examination process considerably.

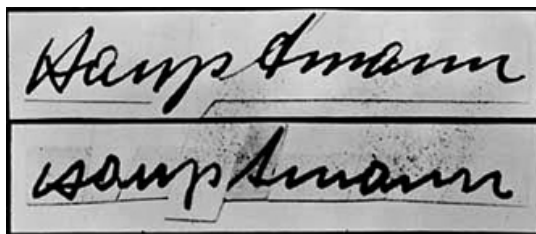


Figure 1.3 Lindbergh kidnapping case² (1932): Comparison between the known signature of the prime suspect (top) and the signature composed from individual letters in the ransom notes (bottom) was a significant investigation development.

Summarizing, we could say that handwriting carries remarkable information about the writing style itself, the author of writing and the personality of the writer. Inspired by this very fact we decided to carry out a study of how to develop algorithms that allow an automatic analysis of handwriting. Since handwriting analysis itself has different aspects, we limited our focus of research to the classification of handwritings, in particular, the recognition of the writer of a handwritten document.

² Source : FBI History – Famous Cases : www.fbi.gov

The problem of writer recognition is related to that of handwriting recognition [Plamondon & Srihari, 2000][Vinciarelli, 2002]. Handwriting recognition aims at eliminating the writer-dependent variations between writings and thus identifying the individual characters and words. Writer recognition on the other hand relies on these writer-specific variations between character shapes (Figure 1.4) which allow to characterize its writer’s hand. Despite this contradiction between the two approaches, writer recognition can be handy in handwriting recognition, exploiting the principle of adaptation of the system to the type of writer [Nosary et al., 1999].

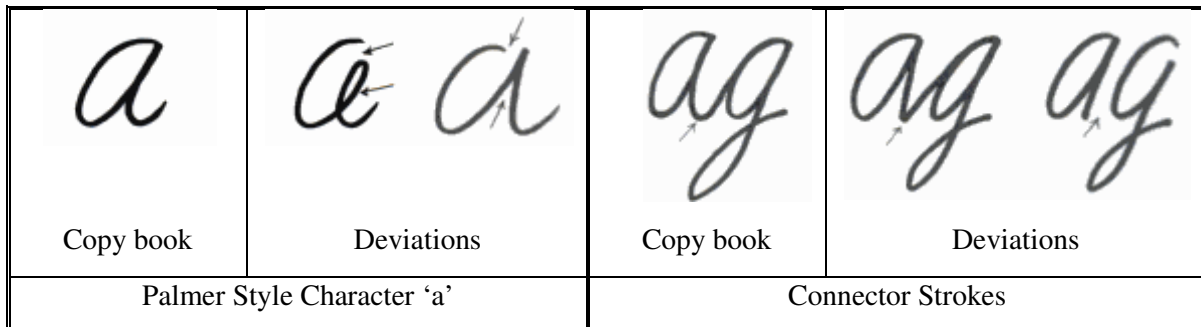


Figure 1.4 Copy book norms and individual deviations (Image: [Nickell & Fischer, 1999])

[Schomaker, 1998] identifies four factors responsible for variations in handwriting (Figure 1.5). These are affine transforms (rotation, translation, scaling etc.), allographic variations (character shapes employed by a writer), neuro-biomechanical variability and sequencing variability (variable order of stroke production). Among these factors, the allographic variations provide the most useful information for automatic writer recognition [Schomaker & Bulacu, 2004]. These variations result in the first place from the copy book style taught and then from the writer-specific preferences in drawing these shapes, developed over time (Figure 1.4). According to one of the greatest handwriting experts of his times, Albert S. Osborn (1858 – 1936): “Only a small portion of the vast variety of forms in writing can be accounted for by tracing them back to a pattern system. Thousands of these characteristics are individual inventions and developments” [Nickell & Fischer, 1999].

Writer recognition is generally divided into writer identification and verification. Writer identification involves a one-to-many search where given a handwritten sample s of unknown authorship and a database with samples of N known authors, the objective is to find the writer (or a likely list of writers) of s in the database. Writer verification on the other hand involves a one-to-one comparison where given two handwritten samples $s1$ and $s2$ the objective is to determine whether the two have been written by the same person or not. The two models have been indicated in (Figure 1.6). Writer identification is generally carried out by calculating a similarity index between the questioned writing and all the writings of known writers and sorting the retrieved results in a hit list with an increasing distance from the query. Choosing appropriate acceptance thresholds on these similarity indices can then be used to perform writer verification.

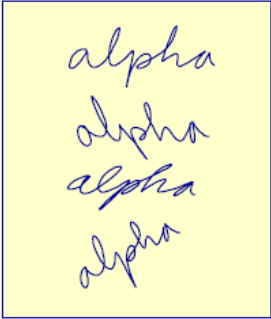
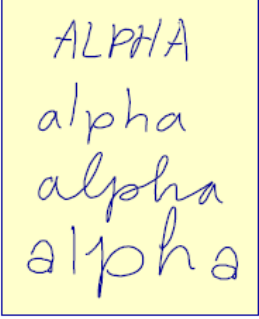
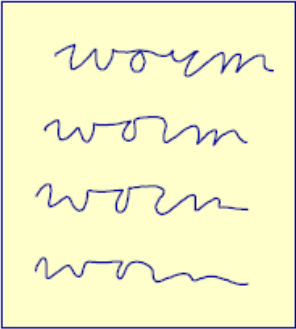
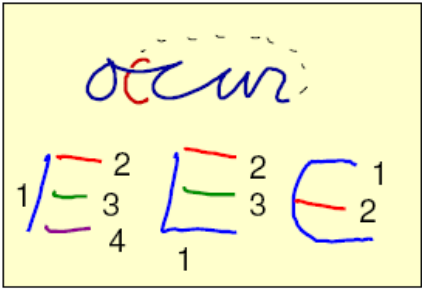
	
Affine transforms	Allographic variations
	
Neuro-biomechanical variability	Sequencing variability

Figure 1.5 Factors causing handwriting variability. Image reproduced from [Schomaker, 1998]

When dealing with large data sets, writer identification can also be employed as a filtering step prior to verification [Bensefia et al., 2005b]. The identification step could extract a sub-set of likely candidate writings from the database each of which can then be compared to the questioned writing either by the verification system or by an expert. The target performance of a writer identification system, as reported in [Schomaker & Bulacu, 2004], aims at attaining a near-100 percent correct identification rate in a hit list of 100 writers, computed from a database of up to 10,000 samples, the size of search sets in current European forensic databases. This performance however, remains far from being achieved for the time being. We have attempted to make a contribution in achieving this goal by developing a system for writer recognition from scanned image of handwritten documents.

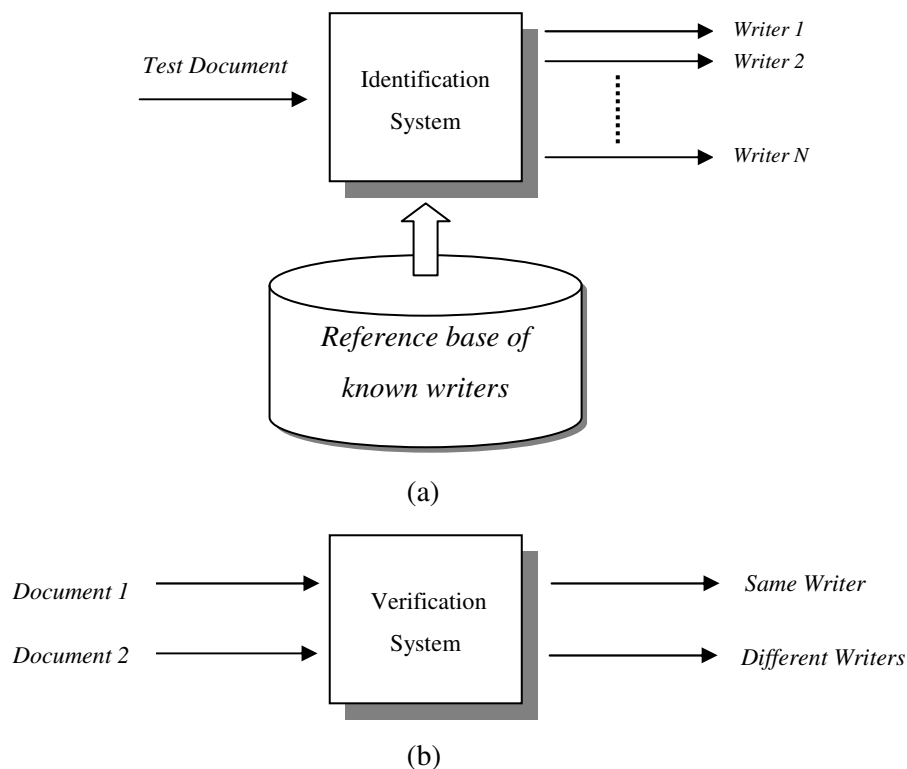


Figure 1.6 Writer Identification and Verification models

Our methodology is based on two facets of handwriting. In the first place we exploit the existence of certain redundant patterns in a writing to characterize its writer. Unlike classical approaches which consider these patterns at the grapheme level, we have introduced much elementary writing fragments which correspond more to the writing gestures than writing shapes. This approach will make the subject of our study in chapter 3. We will next focus on two important visual attributes of writing in chapter 4, namely orientation and curvature that enable distinguishing one writing from another. These properties are captured by a set of features extracted from the contours of handwriting images. We will then explore the effect of combining the two aspects in characterizing the writer of a given sample. Finally, we will demonstrate the application of the proposed methodology on ancient medieval manuscripts, Arabic handwritten documents and signatures. But before proceeding to the discussion of our methodology, we would present an overview of the significant contributions to writer recognition in the next chapter.

Chapter 2

State of the art

Despite the development of electronic documents and predictions of a paperless world, the importance of handwritten documents has retained its place and the problems of identification and authentication of writers have been an active area of research over the past few years. A wide variety of systems that are based on the use of computer image processing and pattern recognition techniques have been proposed to solve the problems encountered in automatic analysis of handwriting and recognition of the writer of a questioned document. A comprehensive survey of the work in writer recognition until 1989 has been presented in [Plamondon & Lorette, 1989]. Here, we will be more interested in surveying the approaches developed in the last several years, thanks to the renewed interest of the document analysis community for this domain.

The techniques for writer identification and handwriting classification are traditionally categorized into two broad classes: text-dependent and text-independent methods.

In text dependent methods the writing samples to be compared require to contain the same fixed text. Signature verification, for example, can be considered in this category. These methods normally use the comparison between individual characters or words of known transcription and thus require the text to be recognized or segmented (manually or automatically) into characters or words prior to writer recognition [Schlapbach, 2007].

The text independent methods on the other hand identify the writer of a document independent of its semantic content. These methods use features extracted from the entire image of a text or from a region of interest. A minimal amount of handwriting is necessary in order to derive stable features insensitive to the text content of the samples [Bulacu & Schomaker, 2007].

Generally speaking, text-dependent methods operate at the character or word level whereas text-independent methods work on the line or paragraph levels. Evidently, text independent methods are less constrained and more useful for practical applications where the human intervention is to be minimized. It should however be noted that text-dependent analysis carries significant importance in forensic document examination where the forensic experts strive to compare certain attributes of individual character shapes. In addition, the discriminating power of different characters may vary as well [Zhang et al., 2003] [Greening et al., 1995] [Tan et al., 2009] [Eldridge et al., 1984]. An ideal solution would therefore be to use automatic systems to find a set of probable candidates from large data sets and then present the reduced list of a manageable size to the document examiner. This first part of finding a subset of likely writers of a questioned

document has been the focus of the research in the document analysis community and the methods proposed in an attempt to achieve this goal will make the subject of this chapter.

We will present in this chapter, an overview of significant contributions to the field of writer identification. It should be noted that we will not explicitly discuss writer verification as we assume that if a (dis)similarity measure has been defined to compare two writings, one can perform writer verification by choosing appropriate thresholds on the calculated (dis)similarity values. It is also important to distinguish offline and online writer identification systems. Contrary to offline documents, online handwriting contains additional information about stroke sequencing, velocity, pressure and pen-up and pen down events etc. For offline documents, we have to rely on the scanned images of handwriting. In our discussion, offline writer identification systems will be the main focus of our study.

We will start with a brief introduction of the commonly used data sets in the domain. We will then give an account of some of the major text dependent methods followed by a discussion of the text independent methods which will constitute the main part of this chapter. Along with writer recognition, we also discuss the closely related area of recognition of writing styles where the objective is not to precisely identify the writer of a given document but to group ‘similar’ writing styles into classes. Finally we will present a comparative analysis of these methods and try to identify where the automatic writer identification stands in terms of performance requisite of real world applications.

2.1 Data Sets

The availability of data sets is one of the fundamental requirements for development and evaluation in any research domain and same is the case with handwriting and writer recognition. Furthermore, over the last few years, the comparison of different methods gained much interest in the handwriting recognition community which led to the development and annotation of standard data sets. Examples of commonly used data sets include IAM [Marti & Bunke, 2002], CEDAR [Hull, 1994], NIST [Wilkinson et al., 1992] and CENPARMI [Suen et al., 1992] in the offline while UNIPEN [Guyon et al., 1994] and IRONOFF [Viard-Gaudin et al., 1999] in the online domain. A relatively new French data set RIMES [Grosicki et al., 2008] is also gaining popularity in the research community. In the following, we will give a brief account of IAM and RIMES data sets, two interesting collections in the perspective of offline writer recognition.

2.1.1 IAM Data Set

The most well known and widely used data set in writer recognition is the IAM³ data set [Marti & Bunke, 2002] that contains forms with handwritten English text of variable content. The images

³ IAM = Institut für Informatik und angewandte Mathematik , University of Bern, Bern, Switzerland

have been scanned at 300 dpi, 8 bits/ pixel, grey-scale. The data set is fully annotated with information about the writer identity, the ground truth text and the segmentation at the line, sentence and word levels. A total of 650 writers have contributed to the data set with a variable number of handwritten samples per writer that varies from one sample (350 writers) to 59 samples (one writer) as illustrated in Figure 2.1 which gives an overview of the samples per writer in the data set. (excluding the exceptional writer with 59 samples).

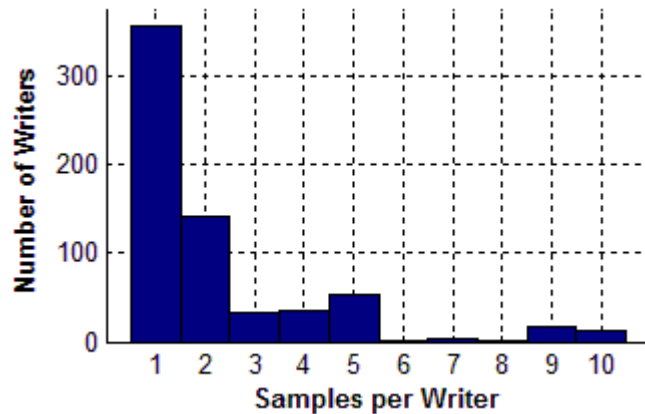


Figure 2.1 Distribution of samples per writer in the IAM data set

2.1.2 RIMES Data Set

RIMES⁴, a relatively new data set, comprises handwritten letters in French text representing the content sent by individuals to companies or administrations. More than 1300 writers contributed to the data collection by writing up to 5 letters, making a total of 5600 letters (more than 12,000 pages if we also count the questionnaire accompanying the letters and the (optional) fax cover sheets). The data set is completely annotated and secondary collections of isolated characters, handwritten words and logos have also been created. Two phases of evaluation campaigns, targeting applications like handwriting recognition, writer identification, layout analysis etc. have been carried out, using a subset of the complete database, in the last two years (2007 & 2008). The same data set has been employed in the ICDAR 2009 handwriting recognition contest. The data set will be made publicly available to the handwritten recognition and document analysis communities after the final phase of evaluations.

After having introduced the data sets, we will now move on to the presentation of writer recognition methods in the following sections.

⁴ Reconnaissance et Indexation de données Manuscrites et de fac similÉS

2.2 Text- Dependent Writer Recognition

The earlier research in writer identification has mainly witnessed text-dependent methods. The writers would copy a (fixed) given text which could vary from an entire page to individual words or characters. In this respect, text-dependent writer identification is quite similar to signature verification. In case of signatures, the writer himself has the freedom to choose the text (signature) whereas for writer recognition, the user might be asked to copy a predefined text. Naturally, text-dependent methods can achieve higher accuracy with small amount of available text as opposed to text-independent methods.

The classical text-dependent methods compared in [Plamondon & Lorette, 1989] have mostly been applied to a very limited number of writers that varied in general from two to ten writers. Only [Naske, 1982] considered a dataset of 100 writers (each having contributed a single word 10 times), achieving an identification rate of 98%. In our discussion, we will mainly concentrate on the recent contributions to text-dependent writer identification. These studies are mainly motivated by forensic applications and aim to design computational algorithms to extract the features used by forensic document examiners [Huber & Headrick, 1999]. Since writer identification is our main focus of study, signature verification methods will not be a part of our discussion.

One of the most comprehensive studies validating the hypothesis of the individuality of handwriting has been presented in [Srihari et al., 2002]. Handwriting samples of 1500 individuals, representative of the U.S. population were collected, each individual copying a source document (the CEDAR letter containing 156 words capturing all the characters and certain character combinations of interest) three times. The authors propose an extraction of a set of macro and micro features from each of the samples. Macro-features are extracted at the document level (entire handwritten manuscript) or at the paragraph, line and word levels while the micro-features are computed at the allograph, or character shape level.

A total of eleven macro-features have been employed which have been grouped into three broad categories: darkness features (entropy of grey values, grey level threshold and number of black pixels), contour features (number of interior and exterior contours and the number of horizontal, vertical, positive and negative slope components) and averaged line-level features (slant and height). Certain of these features are also calculated at the paragraph and word level, the paragraph level features being complemented by two (aspect ratio and indentation) while the word level features by three (upper and lower zone ratios and word length) additional features. The proposed micro-features consist of 512 binary features corresponding to gradient (192 bits), structural (192 bits), and concavity (128 bits) features, calculated at the character level.

The effectiveness of the features has been evaluated on writer identification and verification. For macro-features, the distance between a pair of documents with feature vectors A and B is defined

by the Euclidean distance between A and B while for micro-features, two characters represented by binary feature vectors A and B are compared by the following similarity measure:

$$S(A, B) = A^t B + \frac{\bar{A}^t \bar{B}}{2} \quad (2.1)$$

The accuracy on writer identification was evaluated using a nearest neighbour rule by randomly choosing a number of n writers from a total of 1000; selecting one sample of one of these writers as query and the remaining $(3 \times n - 1)$ samples as reference. This leave-one-out method was performed 1000 times for each n . The macro features were computed at the document, paragraph and word levels (to analyze system sensitivity to the amount of text) while the micro features were calculated for characters of the word ‘referred’. The two feature sets were tested alone and then combined by using the macro-features as a filter that reduces the number of writers from 1000 to 100 and then using the micro-features to identify the writer among the 100 choices. The authors realize an identification in the range of 98% for $n=2$ to 87% for $n=900$. Writer verification is performed using a fully connected three layer artificial neural network yielding an accuracy of 96% once the two sets of features are combined.

Later studies by the same group focused on investigating the discriminatory power of individual characters [Zhang et al., 2003] and numerals [Srihari et al., 2003]. According to their analysis: ‘G’, ‘b’, ‘N’, ‘I’, ‘K’, ‘J’, ‘W’, ‘D’, ‘h’, ‘f’ came out to be the 10 most discriminating characters. In case of numerals, ‘5’ was found to be the most while ‘1’ the least discriminating numeral. The individuality of characters was then compared with that of words [Zhang & Srihari, 2003] by carrying out a study on four characteristics words ‘been’, ‘Cohen’, ‘Medical’ and ‘referred’ using the same set of micro features. Naturally the words were more effective in differentiating writings than the individual characters. The study was then extended to 25 words with the addition of Word Model Recognition [Kim & Govindaraju, 1997], shape curvature and shape context features [Belongie et al., 2002]. The computational details of these features can be found in [Tomai et al., 2004].

[Pervouchine & Leedham, 2007] studies a set of structural micro features extracted from three characters ‘d’, ‘y’ and ‘f’ and grapheme ‘th’. These characters (grapheme) were chosen based on an analysis of character frequencies in several novels and their discriminating powers. The character images were extracted manually from 600 samples of the CEDAR letter [Srihari et al., 2002] contributed by 200 writers producing a total of 30, 24, 24 and 27 samples of the respective character per writer. A set of 31 features was computed from these characters and was meant to represent a subset of the features used by forensic experts [Huber & Headrick, 1999]. These features mainly include the widths, heights, width to height ratios, and depending upon the character, the relative heights of ascender/descender, presence/absence of loop, certain slants and

distances, etc. The classification was carried out using a neural network while normalized Manhattan distance was used as the distance measure between the features:

$$d(\vec{F}^1, \vec{F}^2) = \frac{1}{k} \sum_{i=1}^k \frac{|F_i^1 - F_i^2|}{\max_i - \min_i} \quad (2.2)$$

Where k is the number of features and \max_i and \min_i represent the maximum and minimum values of the i^{th} feature in the data set respectively.

The authors have also carried out an analysis of the usefulness of features by searching the optimal feature sets using a wrapper method. Classification accuracies of 16%, 20%, 26% and 36% were achieved using the characters 'd', 'y', 'f' and 'th' respectively. The highest achieved classification accuracy was 58% when using the optimal subset of all four character features. Out of the 31 features evaluated, 13 were found to be relevant, 4 irrelevant and the rest partially relevant for writer discrimination. The features that came out to be irrelevant included the final stroke angle, the fissure angle of character 'd' and the presence of loop at the upper and lower points of f-stem. The authors however precise that it cannot be concluded that these features are useless as the results might be linked to a faulty formalization of features and, feature values might not have reflected the real situation. Separate contributions dedicated to the study of similar features on letter 'a' [Sutanto et al., 2003], grapheme 'th' [Pervouchine & Leedham, 2006] and numerals [Leedham & Chachra, 2003] can also be found in the literature.

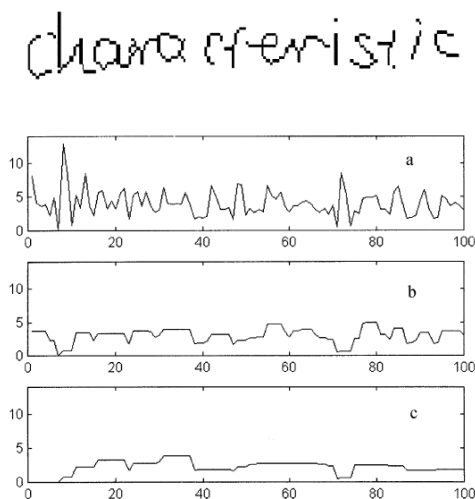


Figure 2.2 The morphological opening on the projection function: (a) initial histogram function of the word 'characteristic' (b) opening with line structuring element of length 3 (c) opening with line structuring element of length 7. (Image: [Zois & Anastassopoulos, 2000])

In addition to the forensic features discussed above, [Zois & Anastassopoulos, 2000] introduced a new feature based on a morphological transform of the vertical histogram function of a word. The binarized and one pixel thinned image of the word is projected onto the horizontal direction.

Taking into account the blank spaces between the letters, two versions of the projection function are created: one with and one without the zero bins. The functions are then resampled to have a total of 100 bins, making it independent of the word length. The obtained curves are morphologically processed (opening) in order to obtain the feature vector. The differences between successive openings denote the amount of information that is removed by the increasing in size structuring element. Figure 2.2 shows the results of applying successive openings to the histogram function with line structuring elements of length 3 and 7 respectively. The final feature vector is created merely by partitioning the representation space into a number of segments and measuring the relative amount of area that the two structuring elements reject in each block.

The discrimination capabilities of the proposed feature were evaluated on a database of 50 writers each having 45 samples of two different words of the same length: an English word and the corresponding (same meaning) Greek one. Employing two different classification schemes (the Bayesian classifier and the neural networks) identification rates in excess of 95% were realized. The authors have also shown that the identification results obtained on the English and Greek words are equivalent, thus highlighting the language independence of the proposed method.

Among the methods presented, most of the features are meaningful only in text-dependent mode. However, some of the features introduced in [Srihari et al., 2002] e.g. (slant and slopes etc.) could also be used to characterize the writer of a document independent of the textual content.

This concludes our discussion on text-dependent automatic writer identification which might achieve very good performance but is very constrained and is mostly not applicable in many practical applications like the identification of the writers of archived handwritten documents and crime suspect identification from large sized forensic data sets etc. [Said et al., 2000]. Therefore, most of the methods developed lately fall in the text-independent analysis of handwriting as discussed in the section to follow.

2.3 Text-Independent Writer Recognition

We will now present the methods that are based on text independent analysis of handwriting. In order to better describe the different approaches for the text-independent writing identification, we have compiled existing methods in three parts. The first part makes reference to the approaches that we have termed as global. We then present local approaches and finally we will describe the methods that are based on handwriting recognition.

2.3.1 Global Methods

These methods identify the writer of a document based on the overall look and feel of the writing. We will discuss these methods under three categories namely texture, fractal and Zipf analysis of writing. Each of these has been detailed in the following sections.

2.3.1.1 Texture Analysis of Handwriting

[Said et al., 2000] presents an algorithm for automatic text-independent writer identification considering the writing of each individual as a different texture. The system comprises three main steps: normalization, feature extraction and identification. During normalization, the detection and correction of the skewed words in the handwriting images is performed using line fitting on the connected components. Then, the space between lines/words and margins are set to a predefined size by means of cell padding to produce a well defined pattern for texture analysis. The texture features are obtained by two established methods namely the multi-channel Gabor filtering [Peake & Tan, 1997] and the grey scale co-occurrence matrix (GSCM) [Tan, 1996], implemented on random non-overlapping blocks (of 128x128 pixels) extracted from the normalized image.

The two Gabor filters are of opposite symmetry and are given by:

$$\begin{aligned}h_e(x, y; f, \theta) &= g(x, y) \cos(2\pi f(x\cos\theta + y\sin\theta)) \\h_o(x, y; f, \theta) &= g(x, y) \sin(2\pi f(x\cos\theta + y\sin\theta))\end{aligned}\tag{2.3}$$

Where g is a 2-D Gaussian function and, f and θ are the radial frequency and orientation respectively which define the location of the channel in the frequency plane. Four frequencies of 4, 8, 16 and 32 cycles/degree have been used and for each of these frequencies, filtering is performed for 4 orientations (0° , 45° , 90° and 135°). This gives a total of 16 output images (4 for each frequency), from which the writer's features are extracted. These features are the mean and the standard deviation of each output image. Therefore, 32 features per input image are calculated.

The matrix of co-occurrence is a square matrix M of size N (number of grey levels) where each element $M(i, j)$ of the matrix represents the number of pairs of pixels separated by a distance d for an angle α , having the grey values i and j respectively. The authors have considered 5 distances (1, 2, 3, 4 and 5) and 4 directions (0° , 45° , 90° and 135°) on binary handwriting images ($N=2$) giving a total of 20 matrices. For each 2×2 matrix there are only 3 independent values due to the diagonal symmetry and they are used directly as features thus giving a total of 60 (20×3) features per handwriting image.

For identification, two classifiers were considered, namely the weighted Euclidean distance (WED) classifier and the nearest neighbour classifier (K-NN). Forty writers were divided into two equal groups and 25 non-overlapping handwritten blocks were extracted for each person. The experiments were carried out by using 10 training and 15 test followed by 15 training and 10 test images per writer for each group of 20 writers. Testing was conducted using different combinations of features under both classifiers and an identification rate of as high as **96%** was achieved with Gabor filtering technique under WED.

Another significant contribution that employs a set of texture level features [Bulacu et al., 2003] proposes a set of contour-based joint directional probability distribution functions to characterize an individual handwriting style. The contour direction distribution is determined by analyzing the fragment between two contour points (A and B as illustrated in Figure 2.3) taken a certain distance apart and finding the angle that the fragment makes with the horizontal. These angles are determined for each of contour points and then counted into a histogram (quantized in 8, 12 or 16 directions) that is finally normalized to a probability distribution $p(\phi_l)$.

The authors have introduced two additional features: the bivariate contour-angle probability distributions $p(\phi_1, \phi_2)$ quantifying the probability of finding two hinged edge fragments oriented at angles ϕ_1 and ϕ_2 respectively and, $p(\phi_1, \phi_3)$: the joint probability distribution of the two contour angles occurring at both ends of a run-length on white [Bulacu & Schomaker, 2003]. These features have been illustrated in Figure 2.3. For the purpose of comparison, the authors have also evaluated three other features widely used for writer identification; the run-length distributions [Arazi, 1983, Arazi, 1977], autocorrelation and entropy.

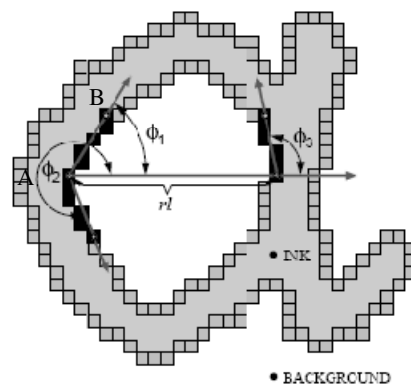


Figure 2.3 Extraction of the edge-based texture features on letter "a" (Image: [Bulacu & Schomaker, 2006])

The features evaluated on 250 subjects of the Firemaker⁵ data set using a leave-one-out approach reported identification rates of 43%, 84% and 63% for the upper case while 48%, 81% and 75% for lower case writings for the three proposed features respectively. On the 650 writers of IAM database, the authors achieved identification rates of 46%, 81% and 77% respectively. In later studies, the authors combined these texture level features with allographic features which will be discussed shortly.

2.3.1.2 Fractal Analysis of Handwriting

The fractal dimension, as defined by B. Mandelbrot [Mandelbrot, 1975], is a “number which measures the degree of irregularity or of fragmentation of a set”, or the measure of the complexity

⁵ A private data set produced at the Nijmegen Institute for Cognition and Information, Netherlands

of the studied set. The fractal behaviour of handwritings was first proved by N. Vincent in [Vincent & Emptoz, 1995]. Later studies show that under certain conditions of observation, the fractal parameters are stable and discriminate enough to establish a handwriting classification according to styles [Boulétreau et al., 1998]. The fractal dimension calculus applied was based on the measure of the Minkowski-Bouligand dimension which is given for a set X as:

$$D(X) = \lim_{r \rightarrow 0} \left[2 - \frac{\log(A(X_r))}{\log(r)} \right] \quad (2.4)$$

Where $A(X_r)$ is the area of the optimal covering of X by balls of radius r . For a fractal curve, the behaviour of $\log[A(X_r)/r]$ versus $\log(r)$ is linear and the corresponding graph termed as *evolution graph*. The evolution graph (as indicated in Figure 2.4) is determined by computing the area of the covering of X by balls of radius r which is implemented by dilating X , r times by a ball with radius 1. Three zones with different slopes can be identified in the graph; each characterizes a particular behaviour and corresponds to a different scale of observation.

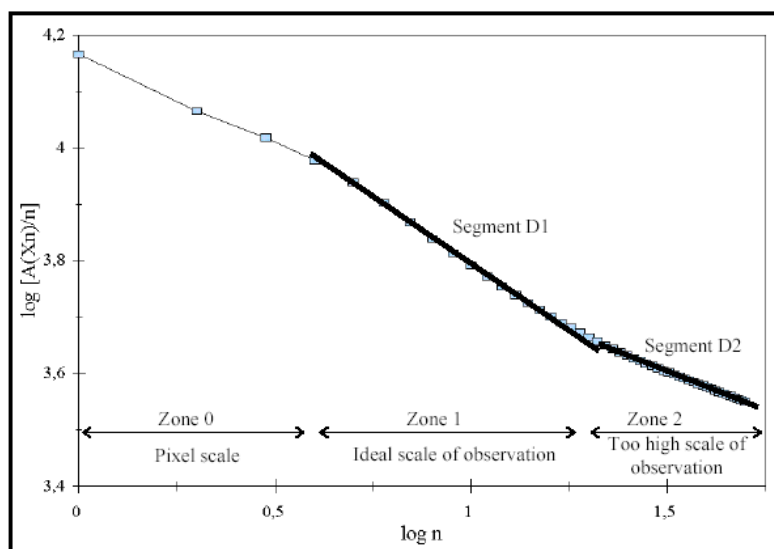


Figure 2.4 Evolution Graph (Image: [Boulétreau et al., 1998])

The authors then define two parameters: fractal dimension and secondary dimension of handwriting from the slopes of these zones. The *fractal dimension* (FD) is computed from the slope of zone 1 while the *secondary dimension* (D2) is calculated from the slope of zone 2. These two parameters correspond to a perception of writing from an adapted and a remote distance of observation respectively.

The authors tested the stability of fractal dimension towards physical constraints linked to both writing and acquisition [Boulétreau et al., 1995]. It was concluded that the use of different writing instruments influenced only on the first few points of the evolution graph (zone 0) which is not considered for the computation of the FD. It has also been demonstrated [Boulétreau, 1997] that

resolution changes produce a translation of the different zones of the graph without affecting the slopes of zone 1 and 2 that are used in the computation of the proposed parameters. Representing the handwritings in the plane, FD vs. D2, the authors suggest that the distribution is linked to legibility. The representation is termed as legibility graph and is illustrated in Figure 2.5.

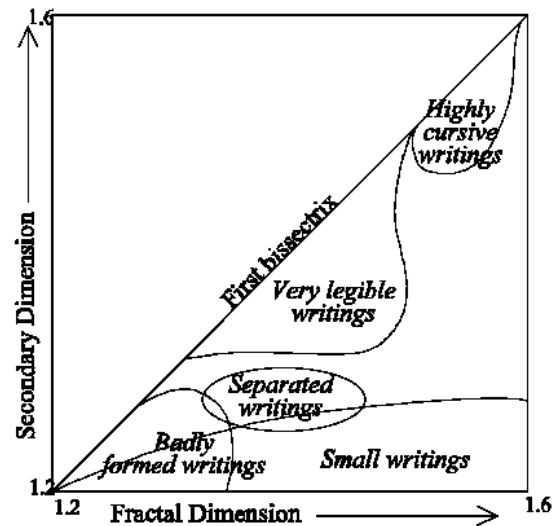


Figure 2.5 Legibility Graph (Image: [Boulétreau et al., 1998])

The legibility graph based on fractal dimension represents a writing in a two dimensional space which is good enough for classifying writing styles but might not be sufficient for problems like writer identification where assigning a writing to a particular class (writer) requires more precision.

Later studies on fractal analysis of handwriting include [Seropian, 2003] presenting a writer identification system based on fractal compression [Fisher, 1995]. The approach is based on the auto-similarity properties present in the writing of an individual. A set of invariant patterns extracted from a writing appears along a fractal compression process where the objective is to find the transformations that are part of a partitioned iterative function system (PIFS) having the initial image as fixed point. To construct the PIFS, the image is partitioned into square sub-images R called *ranges* which are considered to be a result of a transformation of *domains* D (also sub-images of initial image). The fractal compression process hence replaces an image by the system of transformations and the position of associated R and D . A reference base is thus generated for each writer representing inner similarities contained in the writing.

In the decompression step the transformations are iteratively applied to all sub-images of an image until the difference between two successive images of the sequence is small enough. To identify the writer of a query document, one proposition is to use the reference bases of the known set of writers and compare the quality of the images of the new text after fractal compression and decompression steps [Seropian & Vincent, 2002]. Another solution proposed by the authors is to partition the query image in ranges and employ a pattern matching process with the elements of the reference base. The questioned text is then attributed to the writer whose reference base resulted in

the highest number of correspondences [Seropian et al., 2003]. Evaluations carried out on images from 20 different writers reported an identification rate of more than 85%.

This method of fractal compression/decompression of writing is closely related to the code book based methods, both the approaches exploiting the auto-similarities within a writing. The code book based methods will be presented later in the chapter.

2.3.1.3 Zipf Law on Handwriting

In an attempt to index and identify manuscripts, [Pareti & Vincent, 2006] presents a method based on the Zipf law [Zipf, 1949] that models the occurrence of distinct objects in sorted collections, originally used in mono-dimensional domains. The law states that when a given set of symbols (patterns) is sorted with respect to decreasing occurrence frequency, the following relation can be observed:

$$N_{\sigma(i)} = k \times i^a \quad (2.5)$$

Where $N_{\sigma(i)}$ represents the occurrence number of the pattern with rank i , and k and a are constants. This power law is characterized by the value of the exponent a while k is more linked to the length of the studied symbol sequence. The relation is not linear but a simple transform leads to a linear relation between the logarithm of N and the logarithm of the rank.

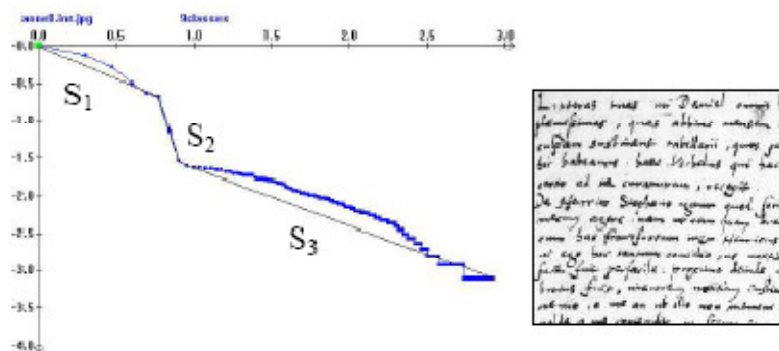


Figure 2.6 An example manuscript and its Zipf plot (Image: [Pareti & Vincent, 2006])

Application of this law to images requires some kind of encoding. The authors have chosen to quantize the grey values to k -levels, k being set to 9 and then to 3. A 3×3 mask (for $k=9$) and a 4 connectivity mask ($k=3$) is employed yielding 9^9 and 3^5 possible patterns respectively. Although Zipf law does not hold for the whole image, Zipf curves are built and approximated by some straight line segments. The authors have chosen to consider in each curve up to three different linear zones (Figure 2.6). The splitting point in a curve segment is defined as the farthest point from the straight line linking the two extreme points of the curve to be split. The three slopes and abscissa of the extremities of each segment are considered as features for document representation and two images are compared by the Hamming Distance in the feature space. The method tested on

a collection of 16th century documents by 20 different writers reported a writer identification rate of as good as 80%.

The method is quite generic in the sense that its applicability is not limited to document images only. The discriminative power of the six primitives however is questionable. Since the mask size is very small, the most frequent patterns in the Zipf curve, which influence the first slope and the respective abscissa value, generally correspond to combinations which might not characterize a writing, e.g. the patterns that are completely white (background) or completely black (contained within the text).

2.3.2 Local Methods

These methods identify the writer based on the localized features of writing. Instead of the overall writing, the focus generally lies on individual characters or allographs and the way a particular writer would draw them. In our discussion, we will first briefly present writer identification from low level features followed by classification of writing styles using allograph modelling. Finally, we will discuss in detail the well-known code book based methods.

2.3.2.1 Low Level Features

[Marti et al., 2001] extracts a set of 12 features (that mainly correspond to visible characteristics of the writing) from handwritten lines of text, which are then classified using the k-nearest neighbour and a feed forward neural network classifier, for writer recognition. Employing the projection profiles, the handwritten text image is first segmented into lines which are then binarized. For each line, a set of features is extracted which mainly includes the height of the three main writing zones and their ratios, width of characters, writing slant and the inter word distances. Additionally, features based on the fractal behaviour of the writing, which are correlated with the writing's legibility, are also employed. These are the same features as the ones presented in [Boulétreau et al., 1998] and correspond to the fractal dimension and secondary dimension as illustrated in Figure 2.4.

The system was evaluated on 100 pages of 20 different writers with a total of 912 text lines divided into five disjoint sub-sets (4 used in training and 1 in testing). An average identification rate of 87.8% is achieved using 7 of the 12 features with the k-nearest neighbour classification and that of 90.7% is measured for the neural network classifier using the complete set of features.

The writer identification rates achieved with these features are promising but they are based on an evaluation protocol where four sample images per writer have been used for training. However, in practical problems, one might not have that much text available for each of the writers leaving the effectiveness of these features debatable.

2.3.2.2 Allograph Modelling

[Gilloux, 1994] described a method for improving the handwriting recognition rate, including the style of writing. A method of writer adaptation is proposed based on several models of writing estimated by the Hidden Markov Models (HMMs). Each model is meant to reflect a particular style which is known beforehand. Markov models assume that the image can be represented as a sequence of observations. In addition, they require these observations to be independent when the hidden state is known. The author uses three ways to classify styles to ensure that the observations are mutually independent when the hidden states of HMMs are known. The first step is normalization of writing, thereby ensuring a better generalization of the markovien model. Other aspects of the writing style are taken into account in the HMM by using several sub models.

The detection of style is carried out during the phase of feature extraction. This step segments the connected components of the words into upper and lower contours and then finds the loops and extensions of each segment of letters. A different symbol is assigned to each possible configuration of loops, extensions and their respective sizes. Taking into account spaces, these characteristics can be summarized in a set of 27 symbols. It is the arrangement of these symbols that makes it possible to classify the different styles of writers.

In another major and well-known contribution [Crettez, 1995], the author proposes an analysis of the variability of handwritings with the objective of identifying the family of the handwriting style before initiating the text recognition process. At this pre-recognition level, the author presents a first degree handwriting characterization on the basis of two kinds of observations. The first category includes a group of three classical measures which are semantically independent: thickness of the tracing, main body of the word and spatial density of characters. The thickness is linked to the pen sharpness and writing pressure, the main body refers to the middle of the three vertical zones of text while the spatial density is proportional to the number of letters per unit length.

The second kind of observations is linked to the successive directions of the tracing. Based on the idea that a given author takes some directions more frequently and more intensively than the others, the normalized histogram of different straight line parts of the tracing is drawn as a polar diagram which is then segmented into four directional lobes. The set of the eight observations defined by the three structural measures, the respective directions of the four lobes and also by the relative intensity of the first directional lobe constitutes an eight dimensional space.

These observations are determined on 3788 words extracted from a set of 980 literal amounts of cheques. The set of handwriting families is then determined employing a fuzzy classification.

These classification methods, although developed primarily for improving the handwriting recognition rate, present some interesting features which could be adapted for writer recognition systems as well.

2.3.2.3 Code book Based Methods

The concept of writer's invariants introduced in [Nosary et al., 1999] was employed by [Bensefia et al., 2002] who proposed a system for writer identification based on template matching of the graphemes extracted from the writings to compare. The morphological redundancy of individual patterns, defined as writer's invariants (code book), is a set of similar patterns or graphemes extracted from the segmentation of handwriting. This allows compression of the handwritten text while maintaining a good identification rate. The connected components of the document are first extracted and segmented into graphemes which are actually elementary patterns of handwriting that are produced by a segmentation based on the analysis of the minima on the upper contour [Nosary et al., 1999] as indicated in Figure 2.7. In later studies, the authors also introduced the concept of bi-grams (tri-grams) obtained as a result of concatenation of two (three) adjacent graphemes (graphemes i and $i+1$) [Nosary et al., 2002].

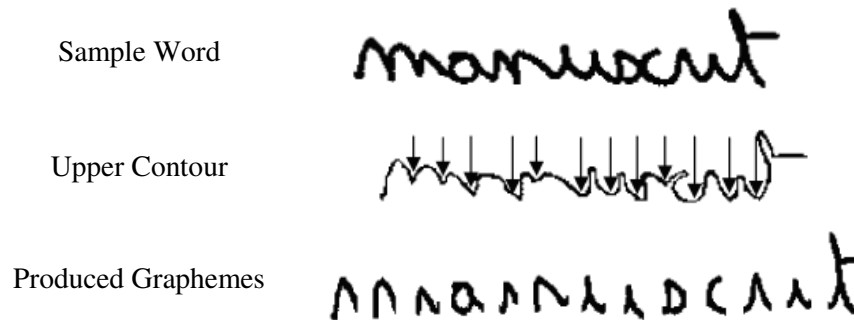


Figure 2.7 Segmentation into graphemes (Image: [Bensefia et al., 2005b])

Once the graphemes have been segmented, morphologically similar ones are grouped together using a sequential clustering algorithm [Friedman & Kandel, 1999] where two graphemes are compared using a correlation similarity measure. The authors also propose to carry out multiple sequential clusterings with a random selection order of the elements to be less sensitive to the order in which the elements are presented for clustering. Finally, only the elements that are always grouped together, when iterating the clustering procedures, are retained to constitute a cluster, termed as an invariant cluster (Figure 2.8).

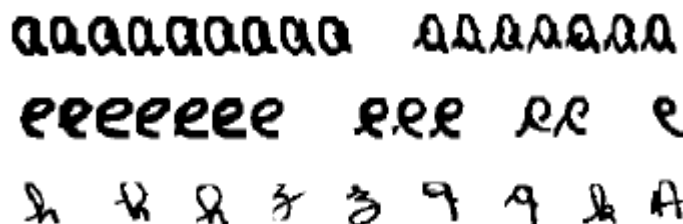


Figure 2.8 Samples of invariant clusters extracted from a handwritten page (Image: [Bensefia et al., 2002])

The handwritten document image is represented by the set of extracted graphemes and the similarity between two documents Q and D is defined as:

$$SIM(Q, D) = \frac{1}{card(Q)} \sum_{i=1}^{card(Q)} \max_{y_j \in D} (sim(x_i, y_j)) \quad (2.6)$$

The experiments were carried out on the PSI database (88 writers – data set constituted in the local lab asking each writer to copy one letter chosen among two suggested ones comprising 107 and 98 words respectively) to evaluate the influence of the text representation (with or without invariants) on writer identification. An identification rate of 97.7% is reported when both the reference and test documents are represented by the complete set of graphemes. Representing only the questioned text by its invariant clusters achieves almost equally good results. For the other two combinations (query document represented by complete set of graphemes and reference documents by invariant clusters or both the documents represented by invariant clusters), the authors report slightly lesser identification rates of approximately 96.5%. It was also shown that samples of 40 to 50 graphemes of the writing allow significant identification rates of nearly 90% on average.

Extending the same idea in the later studies [Bensefia et al., 2003] [Bensefia et al., 2005a] [Bensefia et al., 2005b], instead of determining the invariant clusters from the graphemes of each writer, the authors cluster all the graphemes of the database thus defining a common feature space over the entire dataset. The invariants obtained are viewed as binary features and a feature is considered all the more pertinent as it belongs to a low number of writers.

The writer identification task is formulated within the framework of Information Retrieval for which the authors employ the well known Vector Space Model (VSM). During the indexing phase a document D is represented by a set of weights that are assigned to the features (invariants) as:

$$\vec{D} = (a_0, a_1, \dots, a_{m-1})^T \quad (2.7)$$

With:

$$a_i = FF(\varphi_i, D)IDF(\varphi_i)$$

Where $FF(\varphi_i, D)$ is the feature frequency in document D while $IDF(\varphi_i)$ is the inverse document frequency that is the inverse of the number of documents that contain this feature φ_i .

In the retrieval phase, the similarity between two documents is defined by the normalized inner product of the two vectors e.g. by the cosine of the angle between the two vectors describing the query Q and the document D_j from the reference base. The system was evaluated on the PSI database as discussed earlier and a subset (150 writers) of the IAM database, achieving identification rates of around 93% and 86% respectively. At the bi-gram level the system performance on the IAM database falls, while it rises on the PSI data set (96%). In both cases

trigrams show the same significant decrease in identification performance because of being more dependent on the textual content.

Using a similar approach, Schomaker and Bulacu consider that an individual writer is characterized by a stochastic pattern generator, producing a family of writing shapes. This approach was first applied to isolated uppercase handwriting [Schomaker & Bulacu, 2004] and later it was extended to lowercase cursive writing [Schomaker et al., 2004] by using a segmentation method, the resulting graphemes being termed as fraglets by the authors. The segmentation is based on finding the minima of the lower contour with the added condition that the distance to the upper contour is in the order of the ink-trace width (Figure 2.9). In a later study [Bulacu & Schomaker, 2005], the contour representation of these fraglets was replaced by normalized bitmaps thus making the approach quite similar to [Bensefia et al., 2005b].



Figure 2.9 Segmentation into fraglets (Image: [Schomaker et al., 2004])

Using an independent training set (comprising images from the Unipen [Guyon et al., 1994] dataset), a code book of connected component contours (fraglets in case of lowercase cursive writing) is computed using a Kohonen [Kohonen, 1988] self-organizing feature map (SOFM). The generated code book is then used to compute the probability distribution of these shapes in a given handwriting sample and this distribution is used to characterize the writer. The system evaluated on the 250 writers of upper and lower case Firemaker data set and 650 writers of the IAM data set achieved identification rates of 65%, 75% and 80% respectively.

In later studies [Bulacu & Schomaker, 2006] [Bulacu & Schomaker, 2007], the authors combine these code book based features with the texture level features [Bulacu & Schomaker, 2003] [Bulacu et al., 2003] presented in section 2.3.1.1. The authors investigated a number of feature combinations (by distance averaging) achieving identification rates of as high as 86%, 83% and 89% for the three sets respectively. The authors also combined the lower case Firemaker and the IAM data sets to generate a large data set of 900 writers and achieved an identification rate of up to 87%.

Resuming, one may categorize these methods as being based on one of the two types of code books, universal and writer-specific. The methods based on a universal code book are generally efficient in terms of computational cost however a new code book is to be generated if the script changes. On the other hand, writer specific code books have high computational costs but they could present a generic framework independent of the alphabet under study.

It would also be interesting to compare the two universal code book based approaches [Bensefia et al., 2005b] and [Bulacu & Schomaker, 2007]. While [Bensefia et al., 2005b] generates the code book from the dataset under study, [Bulacu & Schomaker, 2007] computes it from an independent dataset. Another important difference between the two is that [Bulacu & Schomaker, 2007] employs a fixed size (400) code book whereas [Bensefia et al., 2005b] makes use of a sequential clustering algorithm where the size of code book is not fixed *a priori*. It is however difficult to conclude which of these is better as the two methods do not use the same (dis)similarity measure to compare two writings. These methods will be discussed further in the next chapter.

2.3.3 Writer Recognition based on Handwriting Recognition

While writer identification has been employed in handwriting recognition systems (adapting the system to the type of writer) [Nosary et al., 1999], the inverse has also been investigated, that is: using handwriting recognition for writer identification. Schlapbach [Schlapbach & Bunke, 2004] presents a system for writer identification using Hidden Markov Model (HMM) based recognizers. For each writer in the considered population, an individual HMM based handwriting recognition system is trained using data from that writer only. Thus for n different writers one can have n different HMMs. Given an arbitrary line of text as input, each HMM based recognizer outputs a transcription of the input and a recognition score. Based on the assumption that correctly recognized words have a higher score than incorrectly recognized words, and that the recognition rate of a system is higher on input from the writer the system was trained on than on input from other writers, the scores produced by the different HMMs are used to decide who has written the input text line.

Each text line presented to the system is first normalized with respect to slant, skew, baseline location and height. A sliding window is used to transform a normalized handwritten text line into a sequence of feature vectors. The window is one pixel wide and shifted from left to right over a line of text. At each position of the window, nine geometrical features are extracted which represent the following geometrical quantities: number of black pixels in the window, centre of gravity, second order moment, position and contour direction of the upper and lower-most pixel, number of black-to-white transitions in the window, and fraction of pixels between the upper and lower-most pixel. Hence the input to the HMM is a sequence of nine dimensional feature vectors of variable length.

A text line to be classified is presented to the HMM of each writer which outputs a transcription of the input text line together with its log-likelihood score. The authors have also implemented a rejection mechanism calculating the difference between the log-likelihood of the best and the second best ranked writer and normalizing it by the length of the text line; thus defining a

confidence measure (cm) for a given text line. Once this confidence value has been computed, the authors reject an input text line if the value of cm is smaller than a given threshold.

The system is evaluated on 50 writers (2,207 text lines including 4,210 different words) from IAM database. For each writer, the set of available text lines is split into four disjoint subsets to perform full four-fold cross validation experiments. In each of the four runs, three subsets are used to train each of the handwriting recognition systems whereas the remaining text lines form the test set. An average identification rate of 94.47% is achieved which rises to 100% using the confidence measure and rejecting 15% of the results.

This work was followed by the use of Gaussian Mixture Models (GMMs) to model a person's handwriting instead of HMM based recognizers [Schlapbach & Bunke, 2006a]. For each writer, one GMM is trained using (the same nine geometrical) features extracted from text lines coming from the specific writer only. The training is carried out using the Expectation–Maximization (EM) algorithm as a result of which, the authors obtain for each writer a GMM that is specially adapted to the individual handwriting style of that writer. During the identification phase, a text line to be classified is presented to the GMM of each writer and is then assigned to the best ranked writer.

The system evaluated on 4,103 text lines from 100 different writers in the IAM database produced an identification rate of 98.46% as compared to 97.03% with HMM based recognizers on the same data set. The authors conclude that while the GMM based system is conceptually much simpler and takes substantially less time to train than the HMM based system, it achieves a significantly higher writer identification rates. A detailed comparison of the two methods can be found in [Schlapbach & Bunke, 2006b].

Although the authors succeed in achieving very high identification rates, a significant amount of text has been used to train the GMM (HMM) of a writer, which might not be on hand in real world problems.

This concludes our discussion on writer recognition methods. In addition to writer recognition and writing style classification on modern handwritings, studies on ancient manuscripts have also been carried out. Some recent contributions involve [Joutel et al., 2007a] and [Eglin et al., 2007] where the authors strive to carry out a classification of ancient manuscripts from different historical periods. [Joutel et al., 2007a] employs a Curvelets transform while [Eglin et al., 2007] proposes the use of Hermite and Gabor transforms for noise reduction and handwriting classification.

Having presented an overview of some notable studies in the area, summing up, we will now provide a comparative analysis of these methods in the following section.

2.4 Discussion

From the discussion of text-dependent and text-independent methods, one can conclude that in general higher identification rates are achievable with the former type but at the cost of the

requirement to have some fixed text or human intervention to extract the elements (characters or words) to be compared. Text-independent methods are much more useful and applicable. These methods, however, require a certain minimum amount of text to produce acceptable results.

Resuming, we could say that the research on writer recognition that started with the analysis of very constrained writings and very few writers has matured really well over time. Regarding the methods developed, in general, many of the global methods (e.g. Gabor filters, fractal analysis and Zipf law) work reasonably good on a small number of writers. The performance of these approaches however begins to degrade as the number of writers increases. Only the directional features introduced in [Bulacu et al., 2003] maintain a good identification rate when evaluated on larger data sets. Among the local methods, in addition to the structural and statistical features, code book generation has emerged as a very popular as well as effective method for writer identification. These code books could be computed universally for the entire set of writers or for each of the writers separately. Finally, the methods developed for the classification of writing styles could well be integrated with writer identification systems where a query document is compared only to a subset of writings that belong to the same writing style class as that of the document in question.

After a qualitative comparison of methods, we will now summarize the quantitative performance of some of the methods discussed earlier, on writer identification. The comparison that we provide is based on the results that have been reported in the literature. As it can be noticed from Table 2-1, although identification rates of as high as 98% have been reported, they are based on a smaller number of writers. Only the studies [Srihari et al., 2002] and [Bulacu & Schomaker, 2007] have been evaluated on significantly large data sets (900 writers). The data set in the former one however comprises the same text (a letter) written three times by each of the authors hence the effectiveness of the system on text-independent analysis is yet to be explored. Therefore, we can conclude that Bulacu and Schomaker [Bulacu & Schomaker, 2007] currently hold the best performance results reading around 87% on 900 writers. Considering the size of the forensic search data sets (as discussed in the 1st chapter) and the performance demands, the methods reported in the literature are still far from matching these requirements and there is a lot more to do in the field of automatic writer recognition.

Table 2-1 Comparison of writer identification performance

	Writers	Data set	Samples/writer	Sample Size	Performance
[Said et al., 2000]	20*2	-	25	Text blocks of few lines	95.3%/96%
[Zois & Anastassopoulos, 2000]	50	-	45	One word (English/Greek)	92.48%/92.63%
[Marti et al., 2001]	20	IAM	5	Paragraph of 5 to 11 lines	90.7%
[Srihari et al., 2002]	100	CEDAR Letter	3	156 words	94%
	900	CEDAR Letter	3	156 words	87%
[Bensefia et al., 2005b]	150	IAM	2	Paragraph/3-4 words	86%/68%
	88	PSI	1	107 words	96%
[Schlapbach & Bunke, 2006a]	100	IAM	5	Paragraph (8 lines on avg.)	98.46%
[Bulacu & Schomaker, 2007]	250	FireMaker Uppercase	1	Paragraph	86%
	250	Firemaker lower case	2	Paragraph(2 lines to full page)	83%
	650	IAM	2	Paragraph	89%
	900	IAM+Firemaker	2	Paragraph	87%

Chapter 3

Writer Characterization: Redundant Patterns of Writing

In the previous chapter, we discussed the significant contributions to the field of writer recognition over the last two decades. Lately, the focus of research in the domain has shifted towards the extraction of writer specific patterns in writing. As an individual writes, he draws similar characters using the same basic shapes that are generated by more or less the same gesture of hand. This leads to a certain redundancy in the writing of an individual with certain patterns more frequent than others. These redundant patterns, termed as writer's invariants, have been analyzed at the grapheme level in [Nosary et al., 1999] and have shown effective performance on writer recognition [Bensefia et al., 2002] and [Bensefia et al., 2005b]. These redundant shapes, generally termed as '*code book*' could either be writer specific where the frequent writing forms extracted separately for each writer [Bensefia et al., 2002] or universal where these forms are extracted globally (either from the dataset under study [Bensefia et al., 2005b] or from an independent data set [Bulacu & Schomaker, 2007]). The different possible code books explored in the literature have been illustrated in Figure 3.1. In each of these studies, the code book is computed from graphemes which could represent over, under or well segmented characters [Heutte et al., 1998].

The approach that we present is inspired by the same idea of frequent writing shapes; we however, chose to work on a much smaller scale of observation. The methods [Bensefia et al., 2002] [Bensefia et al., 2005b] [Bulacu & Schomaker, 2007], although text independent, are more linked to the way the letters are drawn and segmented and the extracted graphemes might also carry some semantic information. We think the recognition of the writer is more linked to the physical way the lines or loops are produced and hence the observation scale may be inferior to that of a grapheme. The fragments that we consider are small parts of handwritten text that do not carry any semantic information and are obtained by a segmentation of handwriting into small windows. The redundant patterns characterizing an individual are then extracted by grouping similar patterns into clusters.

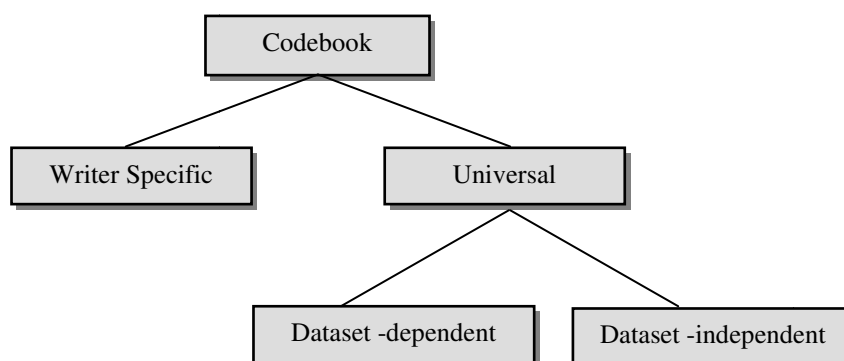


Figure 3.1 Code book based methods for writer recognition

The main advantage of working on small observation windows is that it allows studying the frequent shapes that might be parts of different characters/graphemes. An individual who draws a particular shape (e.g.; loops) in a specific way is expected to always employ the same (similar) patterns when drawing that shape, irrespective of the character being written. As an example, Figure 3.2 shows two loops that are very much similar but come from two different characters ('*h*' and '*k*' respectively) written by the same author. This redundancy may or may not be captured at the grapheme level depending upon the segmentation scheme employed as well as the characters these loops belong to.

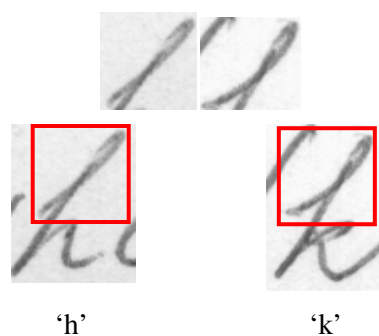


Figure 3.2 Presence of similar loops in two different characters

Developing the same idea and reducing the observation scale further, one can identify the presence of redundancy at the level of even smaller fragments. This is explained by the fact that a writer might use the same gesture of hand, and hence the same pattern, while writing the letters that share similar basic forms. A writer-specific curved fragment, for example, is likely to be produced in a similar form for a number of characters as illustrated in Figure 3.3 that shows four small fragments which are extracted from four different characters but they are all very close to one another. This redundancy of fragments hence is different from the one at the grapheme level, and will be the main focus of our study in this chapter. We will show how these frequent forms can be determined from a handwritten sample. We will first introduce the proposed segmentation

scheme and then discuss the various possibilities of grouping similar fragments (represented by a set of features) together. We will then show how these groups (clusters) of basic writing patterns can be used to characterize the author of a document. We will develop the methodology using the writer specific code book example and then apply the same idea to a global code book. But before presenting the method, we would like to give an overview of the data set that we will be working on.

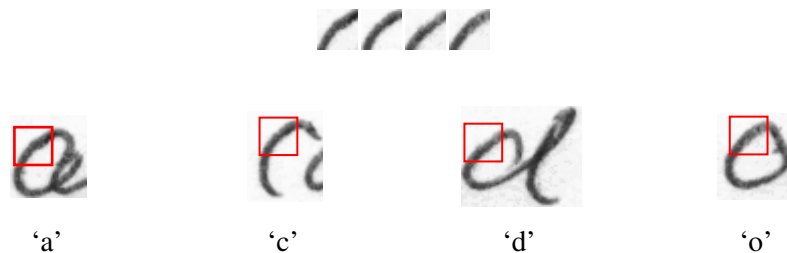


Figure 3.3 Same pattern repeated across different characters

3.1 Data Set

We mainly conducted our experimental evaluations on the IAM data set [Marti & Bunke, 2002]. As discussed in section 2.1.1, this data set contains handwritten texts written by a total of 650 writers with 350 writers having contributed only one page, 300 writers with at least two and 125 writers with at least four pages. In order to fit in all the 650 writers in our study, we kept only the first two images for the writers having more than two pages and split the image roughly in half for writers who contributed a single page thus ensuring two images per writer, one used for training while the other for testing. Furthermore, in order to have an independent validation set, we used the pages ‘three’ and ‘four’ of the 125 writers having at least four samples in the actual data set. Table 3-1 gives an overview of the modified IAM data set employed in our study. Certain experiments are carried out on 300 writers (part 1) while others conducted on the entire set of 650 writers (part1 & part2).

Table 3-1 The modified IAM data set

Data set part	Number of Writers	Contributing Samples
1	300	Sample 1 & Sample 2
2	350	Two halves of the page
3 (Validation)	125	Sample 3 & Sample 4

After having described the data set, we now present our approach for characterizing the writer of a handwritten text. We start with binarization of the document image and extract small writing fragments by dividing the writing into small windows. The frequent writing patterns are then

extracted by grouping similar fragments into clusters which then serve to characterize the writer. These steps have been summarized in Figure 3.4 and each of these has been discussed in the following sections.

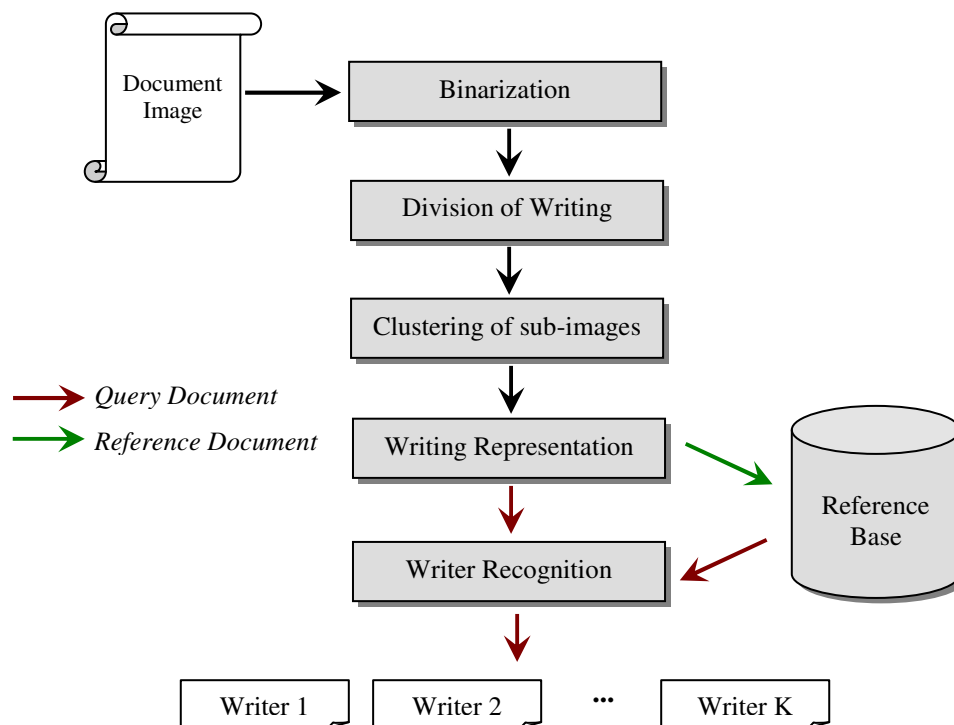


Figure 3.4 An overview of the proposed methodology

3.2 Document Image Binarization

Before proceeding to the analysis of small writing fragments, we binarize the document image. Although the grey level image might carry additional information (e.g. pen pressure) we chose to work on the binarized writing images which simplify the representation and comparison of two forms. Since we mainly focus on modern writings and the corresponding digitized images that we use are not very noisy, a global thresholding using Otsu's algorithm [Otsu, 1979] is applied to generate a binary image of the writing. All the subsequent steps are then carried out on the binarized image.

In order to extract the patterns that a writer employs frequently as he writes, we first need to carry out a division (segmentation) of writing into small sub-images (fragments). This division is carried out by positioning small windows over the writing as discussed in the next section where we investigate the different ways in which the writing could be divided.

3.3 Division of Handwriting

The division of handwriting is an important step as a ‘good’ division would allow to exploit most of the redundancy in writing. For our problem, a ‘good’ division is the one that produces writing fragments in a way that allows meaningful comparison of these fragments. We have chosen to carry out this division employing square windows of size n . This size should be large enough to contain ample information about the style of the author and small enough to ensure a good identification performance. Ideally, the window size should be adjusted according to the writing details (e.g. ink thickness, character height etc.), however we have worked on a fixed size (13x13) that is determined empirically [Siddiqi & Vincent, 2007].

Once the size has been fixed, we proceed to the positioning of windows over text. We have studied a number of window positioning algorithms including regular, horizontal window sliding and adaptive division. They are discussed and compared in the following.

3.3.1 Regular Division

The simplest technique of dividing the writing would be to divide the entire image regularly from left to right and top to bottom and eliminating the windows which do not contain any part of text. The method is simple but the problem is that the resulting windows are not ‘well’ positioned over the text as illustrated in Figure 3.5 where certain fragments that could have easily been contained in a single window, are in fact divided in two different windows. This implies that we will have many windows containing writing fragments which might not be useful to characterize the writer. In addition, the number of windows that we need to process will be relatively larger than required. We need to have a division mechanism that produces comparable writing fragments and ensures the invariance of the trace position within the window. Naturally, this problem does not arise in cases where the text has been divided into allographs or graphemes but in our case, the image (component) is taken in its totality and the study is not based on an intelligent segmentation but on a focalisation process, extracting comparable patterns. We thus seek to address these issues by employing more sophisticated window positioning as presented in the following.

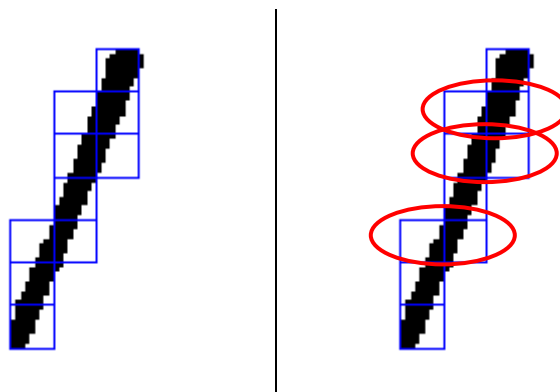


Figure 3.5 Regular division of writing and its drawback

3.3.2 Horizontal Window Sliding

In an attempt to carry out a division that is more relevant for handwriting, we next employ an improved version of the technique proposed in [Vincent et al., 2005]. For each connected component in the text, we fix the vertical origin and divide the component into equally spaced lines separated by a distance of n pixels (n represents the window size). We then slide a window on each of the lines, from left to right, to find the first text (black) pixel that is not already contained within a window. While [Vincent et al., 2005] shifts the entire grid horizontally, we do it for each individual window, thus achieving a better window positioning. The method has been illustrated in Figure 3.6. In general, this approach achieves better positioning as compared to the regular division, however the vertical axis still remains divided in a regular fashion which could result in problems similar to the ones in regular division. We thus need to have an adaptive method that could adjust the position of a window with respect to the writing trace.

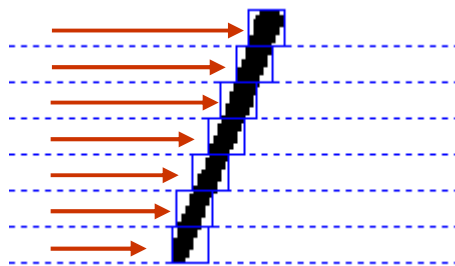


Figure 3.6 Sliding windows horizontally over text

3.3.3 Adaptive Window Positioning

We now propose a division technique that is more adapted to the ink trace and strives to address the problems that are linked to the lack of invariance of line position within the window. Since the images are offline, it is not possible to follow the stroke trajectory that was followed by the writer. Nevertheless, we will seek to follow the ink trace with the objective of achieving an optimal window positioning that is based on the analysis of the skeleton with respect to the drawing.

Given a connected component, we first find its skeleton by removing the pixels on the boundaries of the component without allowing it to break apart (Matlab Image Processing Toolbox). We then place the first window on the original component by finding the first text pixel in top-bottom, left-right scanning (Figure 3.7 (a)) and copy the same window on the skeleton image as well. For each window, we define four flags namely: East, West, North and South, the respective flag being set if the skeleton exits from that particular side (Figure 3.7 (b)). If the skeleton exits from the E (or W), we place the next window towards the right (respectively left) of the current window (on the original component), and displace it in the vertical direction (up and down) to find its ‘best’ position with respect to the text. On the other hand, if the skeleton exists

from the N (or S), we place the next window on top (respectively bottom) of the current window and move it horizontally (left or right) so that it is ‘well placed’ over the text. In cases where the skeleton exists from more than one side, we treat each of the branches separately.

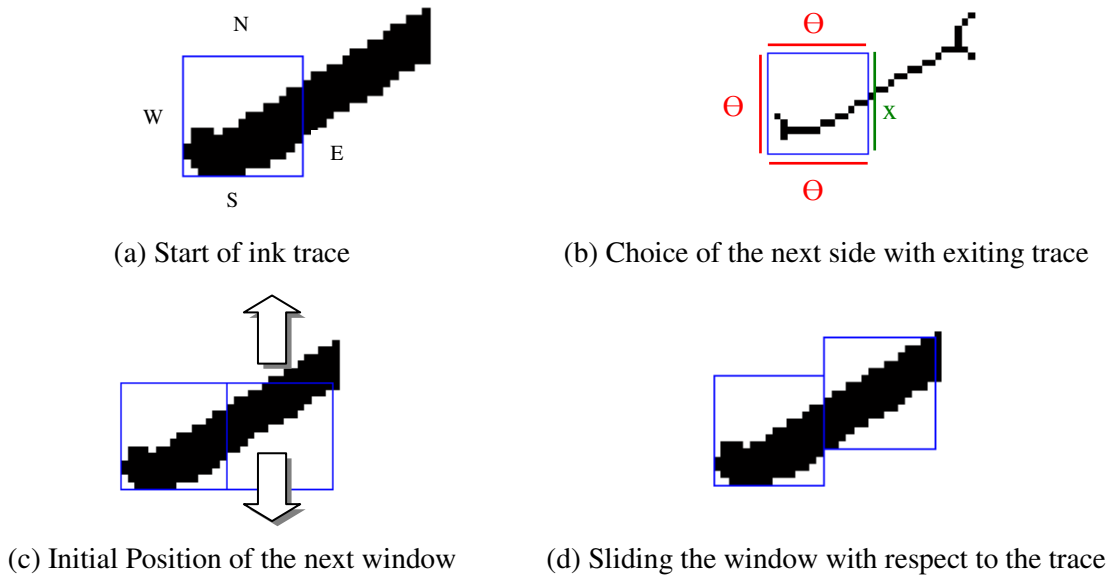


Figure 3.7 Adaptive Window Positioning

Algorithm 3.1 : Window Positioning on text

Place the *firstWindow* and **set** its *flags*

Push the *firstWindow* on the *stack*

while (*stack* is NOT empty)

previousWindow = **Pop**

Place the *currentWindow* on the first set flag *f* of *previousWindow*

Reset the flag *f*

Set the *flags* for the *currentWindow*

if (Any *flag* of *previousWindow* is Set)

Push *previousWindow*

if (Any *flag* of *currentWindow* is Set)

Push *currentWindow*

The term ‘well positioned’ could be explained by the example in Figure 3.7 (c) where a window is placed to the right of an existing window and needs to be moved in the vertical direction to find its final position. Evidently, moving it upwards would result in gaining text pixels in the empty

rows and achieving a better positioning with respect to the stroke whereas moving it downwards would eventually end up in exiting the text pixels of the image. The method is summarized in Algorithm 3.1. It does not guarantee an ‘*ideal*’ division of writing; nevertheless it attempts to work out the issues with the methods discussed earlier.



(a) Regular Division: 84 Windows



(b) Horizontal Sliding: 67 Windows

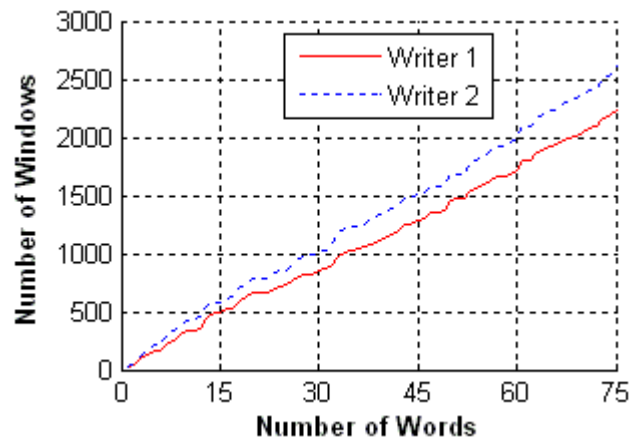


(c) Adaptive Positioning : 54 Windows

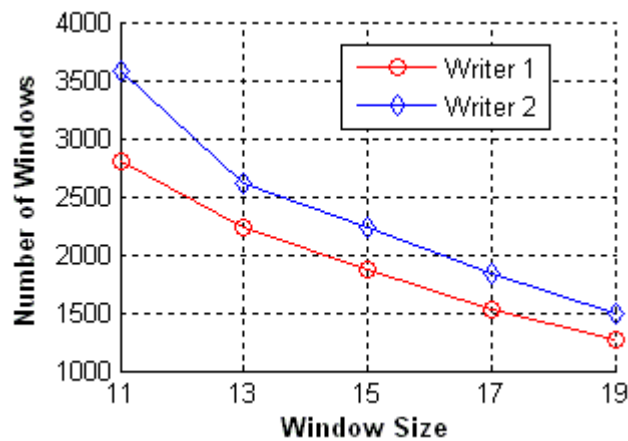
Figure 3.8 Comparison of window positioning algorithms

The window positioning algorithm is applied component by component to the complete image thus resulting in a division of handwriting into small sub-images. A comparison of the division by regular partitioning, sliding the windows horizontally, and positioning them adaptively has been illustrated for the word ‘*headlines*’ in Figure 3.8. It can be noticed that the last method significantly improves the window positioning with respect to the text pixels and as a result of which, though not our primary goal, the number of windows containing text pixels is also reduced. A quantitative comparison of these (last two) methods in terms of performance on writer identification also validates the effectiveness of adaptive window positioning [Siddiqi & Vincent, 2008].

It would be interesting to analyze how the number of windows evolves with respect to the number of words and whether this number is writer dependent or not. Figure 3.9 (a) shows the number of windows as function of the number of words for images of the same text (of 75 words) written by two different writers. Naturally the two curves are quite close to each other in the start and as the number of words increases the gap between the two curves widens, generating about 18% more windows for writer 2 than for writer 1 for the same text. The difference is more or less consistent on different window sizes as illustrated in Figure 3.9 (b) where the number of windows is plotted against the window size.



(a)



(b)

Figure 3.9 For the same text written by two writers number of windows as function of (a) number of words (b) window size (in pixels)

As we have seen, some writer-specific information has already started to show up just by dividing the writing into sub-images. Our objective now is to identify the characteristic (frequent) sub-images (writing fragments) which could eventually characterize the writer of a sample. This is accomplished by implementing a clustering of the sub-images that would group similar fragments into classes as detailed in the following section.

3.4 Clustering of sub-images

Clustering, also known as unsupervised classification is the process of constructing a set of clusters or classes from unlabeled data according to a proximity measure. The objective of clustering in our case is that the sub-images that have been produced by the same gesture of hand are grouped in the same classes (clusters). These clusters would then represent the set of redundant patterns for an individual, the redundancy of a particular pattern being proportional to the (relative) cardinality of

the cluster it belongs to. [Jain & Dubes, 1988] defines the following key steps in a clustering procedure:

- Data representation
- Definition of a (dis)similarity measure
- Clustering / Grouping

We will discuss each of these steps with respect to our problem that involves clustering of sub-images.

3.4.1 Representation of sub-images

The comparison between the sub-images can be made either on the images directly (pattern matching) or by first extracting a set of features and representing the images in a feature space. Directly comparing pixel values is simple but suffers from the disadvantage of keeping n^2 pixel values (for a window size of n) and in addition, the resulting comparisons might not be robust to noise and distortions. We therefore chose to represent the patterns by a set of features. The position of the trace with respect to window is first adjusted (by moving the shape towards the upper-left corner of the window: Figure 3.10) so that the features computed do not depend upon the way a window was positioned over a fragment. It is however not desirable in our case to have rotation invariance as a pattern and its rotated version are not produced by the same gesture of hand and thus should not be grouped in the same class. As far as the scale is concerned, within the same sample, we do not expect a writer to change the writing scale; therefore two sub-images can be compared using the features that are not necessarily scale invariant. Across different samples of a writer, if there is a very significant change in the text size or if the images have been scanned at different resolutions, we assume that the writings have been normalized prior to window positioning and feature computation.

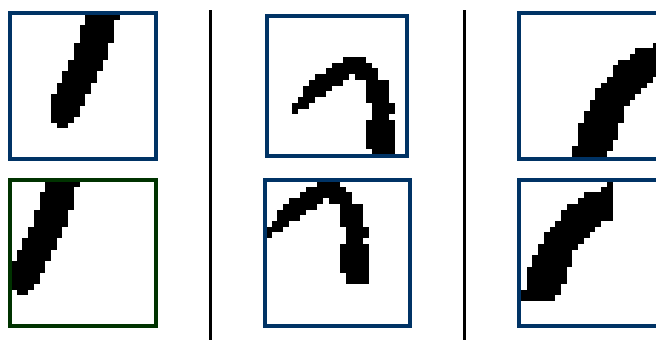


Figure 3.10 Patterns translated before calculating the features

The set of features that we compute for each window includes:

- Horizontal Histogram:** The number of text pixels in each row of the sub-image.

- b) **Vertical Histogram:** The number of text pixels in each column of the sub-image.
- c) **Upper Profile:** The distance of first text pixel from the top of each window.
- d) **Lower Profile:** The distance of the last text pixel from the top of each window.
- e) **Orientation:** The overall direction of the shape: the angle (ranging from -90° to 90°) between the x-axis and the major axis of the ellipse that has the same second-moments as the shape.
- f) **Eccentricity:** The eccentricity of the ellipse that has the same second-moments as the shape. It is the ratio of the distance between the foci of the ellipse and its major axis length and takes on a value between 0 and 1. (An ellipse whose eccentricity is 0 is actually a circle, while an ellipse whose eccentricity is 1 is a line segment.)
- g) **Rectangularity:** The ratio between, area of the object and area of the bounding box.
- h) **Elongation (Aspect Ratio):** The ratio of the height and width of the minimal bounding box. It is defined by the ratio between the shape's minor and major axes.
- i) **Perimeter:** The number of pixels in the boundary of the shape.
- j) **Solidity:** The proportion of the pixels in the convex hull that are also in the shape, and is computed as: *shape area/convex hull area*.

These features are known classically [Costa & Jr, 2001] and have been employed in a variety of applications. The profiles and histograms, for example, have shown good performance on character recognition [Heutte et al., 1998], word spotting [Rath & Manmatha, 2007] and font recognition [Zramdini & Ingold, 1993] while the shape descriptors are known to be effective on tasks like image retrieval [Sarfraz & Ridha, 2007] and shape grouping [Peura & Iivarinen, 1997] etc. The sub-images/writing fragments that we consider can be viewed as parts of characters, each having a specific shape and thus a combination of profiles and shape descriptors is likely to perform well in grouping these patterns.

Once all the features have been calculated, the values are normalized in the interval $[0, 1]$ and hence each window is represented by a vector of dimension $d=4n+6$, where n is the window size. In our case, let S be the set of vectors representing the sub-images, thus we have:

$$S = \{S_i\} \text{ with each } S_i = (s_i^1, s_i^2, \dots, s_i^d) \quad (3.1)$$

3.4.2 (Dis) Similarity Measure

After having represented the sub-images by a set of features, we need to choose a (dis)similarity measure that enables us to compare two sub-images. For the direct comparison of raw pixels, among the several similarity measures defined in the literature [Duda & Hart, 2000], the following

correlation measure has shown good results, for example, in comparing graphemes in the studies [Bensefia et al., 2002, Nosary et al., 1998]:

$$sim(X, Y) = \frac{n_{11}n_{00} - n_{10}n_{01}}{[(n_{11} + n_{10})(n_{01} + n_{00})(n_{11} + n_{01})(n_{10} + n_{00})]^{1/2}} \quad (3.2)$$

With n_{ij} being the number of pixels for which the two sub-images X and Y have values i and j respectively, at the corresponding pixel positions. This measure will be close to 1 if the two compared sub-images are similar and in extreme case it will have a value equal to 1 indicating that the two shapes are exactly the same.

In our case, the sub-images have been represented by a set of features and we calculate the dissimilarity between two patterns by using a distance measure (Euclidean distance) defined on the feature space with dimension d .

$$d(X, Y) = \sqrt{\sum_{k=1}^d (x_k - y_k)^2} \quad (3.3)$$

3.4.3 Clustering

A wide variety of clustering algorithms have been proposed and a comprehensive review can be found in [Jain et al., 1999] where different methods of data clustering have been described. Among the available options, we need to choose a clustering algorithm that does not need to know *a priori* the number of clusters to retain as this number would vary from one writing to another. We have experimented with a number of algorithms that include sequential clustering, multi-phase sequential clustering, hierarchical clustering and minimum spanning tree clustering. We will present them one by one and discuss the pros and cons of each.

3.4.3.1 Sequential Clustering

Sequential clustering [Friedman & Kandel, 1999] is the simplest and the most natural way to cluster data once the number of classes is not known. This method has been employed in [Nosary et al., 1999] for the clustering of graphemes. The algorithm starts with the choice of a proximity threshold and the first element as the centroid of the first cluster. For each of the subsequent patterns, the similarity between the current element and each of the clusters is calculated. The element is then either attributed to the nearest cluster or, in case, it is not close enough to any of the clusters (with respect to the threshold), a new cluster is created.

For our implementation, the (dis)similarity between an element S_i and a cluster C_j is calculated as the Euclidean distance between S_i and the mean (μ_j) of C_j :

$$d(S_i, C_j) = \sqrt{\sum_{k=1}^d (s_{i,k} - \mu_{j,k})^2} \quad (3.4)$$

Every time an element is added to a cluster, the mean of the cluster is also updated. The process is repeated until all the patterns have been assigned to clusters. This sequential assignment of elements to clusters seems promising but the procedure suffers from the drawback of being sensitive to the order in which the patterns are presented. The problem is addressed in [Nosary et al., 1999] [Bensefia et al., 2005a] by carrying out a multi-phase sequential clustering.

3.4.3.2 Multi-phase Sequential Clustering

In this method, in order to be less sensitive to the order of presentation, multiple phases of sequential clustering with random selection of the data points are iterated. Each of the clustering phases thus provides a set of clusters and the final clusters are defined as the groups of patterns that are always clustered together during each sequential clustering phase. While in [Nosary et al., 1999] the authors create single element clusters for the patterns that are not always assigned to the same cluster, we assign them to the nearest clusters (provided the proximity threshold constraint is satisfied, otherwise we also create new clusters).

3.4.3.3 Hierarchical Clustering

This approach comprises a series of partitions, which normally runs from N clusters each containing a single data point to a single cluster containing all the data points (agglomerative hierarchical clustering). At each stage the method merges together the two closest clusters. Differences between methods arise because of the different ways of defining distance (or similarity) between clusters. Depending upon whether the distance between two clusters is calculated as the minimum, maximum or the average of all pair wise distances between data points in the two clusters, the methods are termed as single-link, complete-link and average-link respectively, the three most commonly used hierarchical clustering methods. One may decide to stop clustering either when the clusters are too far apart to be merged (distance criterion) or when there is a sufficiently small number of clusters (number criterion).

For the clustering of sub-images, we have employed the average-link method and the number of clusters retained is determined using the distance criterion as the number of clusters is not known *a priori* in our case.

3.4.3.4 Minimum Spanning Tree Clustering

Among the graph theoretical clustering methods the most well-known algorithm is the minimum spanning tree (MST) clustering [Zahn, 1971] that has been widely used in a variety of applications

[Päivinen, 2005] [Xu & Uberbacher, 1997]. A spanning tree T of a (connected) weighted graph G is a connected sub graph of G such that:

- T contains every vertex of G and,
- T does not contain any cycle.

A *minimum spanning tree* is a spanning tree with the minimum total weight i.e. it connects all the given data points at the lowest possible cost. From the perspective of clustering, if the weights of the edges represent the distances between the data points, removing edges from the MST leads to a collection of connected components which can be defined as clusters [Nadler & Smith, 1993].

For our set of sub-images, we define the weighted (undirected) graph $G(S) = (V, E)$ as follows:

- The vertex set $V = \{S_i | S_i \in S\}$ and,
- The edge set $E = \{(S_i, S_j) | S_i, S_j \in S \text{ and } i \neq j\}$

Each edge $(u, v) \in E$ has a weight that represents the (Euclidean) distance $d(u, v)$, between u and v . A spanning tree T of the graph $G(S)$ is a connected sub graph of $G(S)$ such that T contains every vertex of $G(S)$, and the one with the minimum total distance is the MST. The clustering procedure has been summarized in the following:

- Construct a fully connected graph G of S
- Construct a minimum spanning tree T of G
- Remove all edges with weights $>$ *threshold value*
- Retain the connected components as clusters

The MST is constructed using Prim's algorithm [Prim, 1957] and the clustering is based on the idea that two data points with a short edge-distance should belong to the same cluster (sub tree) and data points with a long edge-distance should belong to different clusters and hence be cut. The number of clusters obtained, naturally, is sensitive to the threshold value chosen. The algorithm works quite well provided the inter-cluster edge-distances are clearly larger than the intra-cluster edge-distances.

3.4.4 Representative Clusters

The sub-images are grouped into clusters using one of the methods discussed above. We next need to identify the clusters that would characterize the writer of a given sample. We thus sort the classes with respect to their cardinality and keep only those having '*sufficient*' number of elements. These classes would then correspond to the frequent patterns occurring in a writing. The number of elements per class, however, depends upon the amount of text in the writing sample, so the '*sufficient*' number can not be fixed value. We therefore compare the number of clusters against the area of text pixels covered and pick the top most important M classes which allow to cover 90% of text pixels in the image as illustrated in the plot of Figure 3.11. It can also be noticed that the number of clusters retained (M) for two different samples of the same writer is quite close

which could serve as a useful parameter in the identification phase (as we will see later on). Figure 3.12 compares the number of clusters on the training and test images of the first few writers in the validation set, the two curves having more or less similar forms.

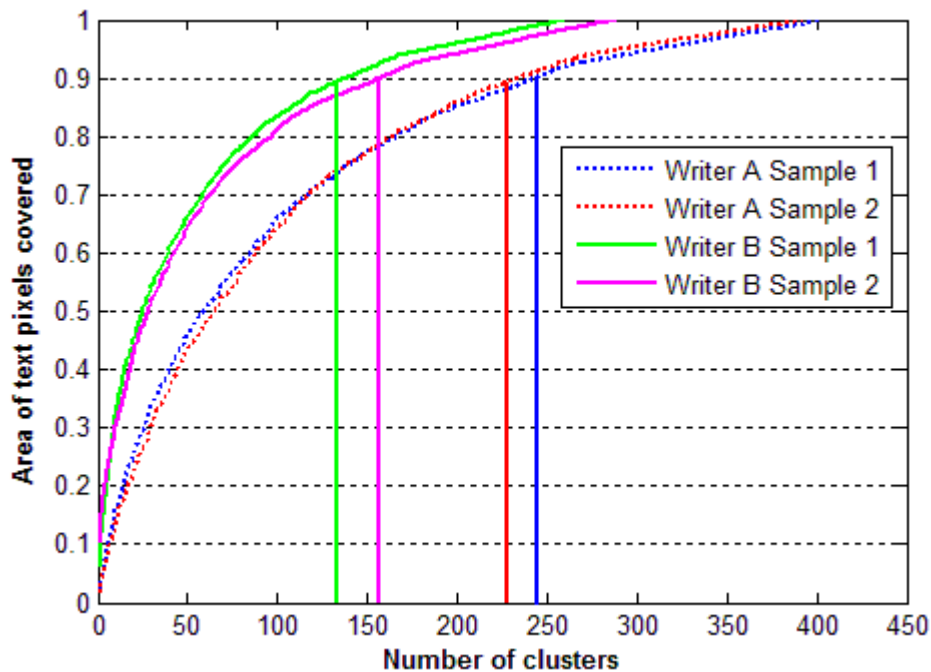


Figure 3.11 Number of clusters and the corresponding area of text pixels covered

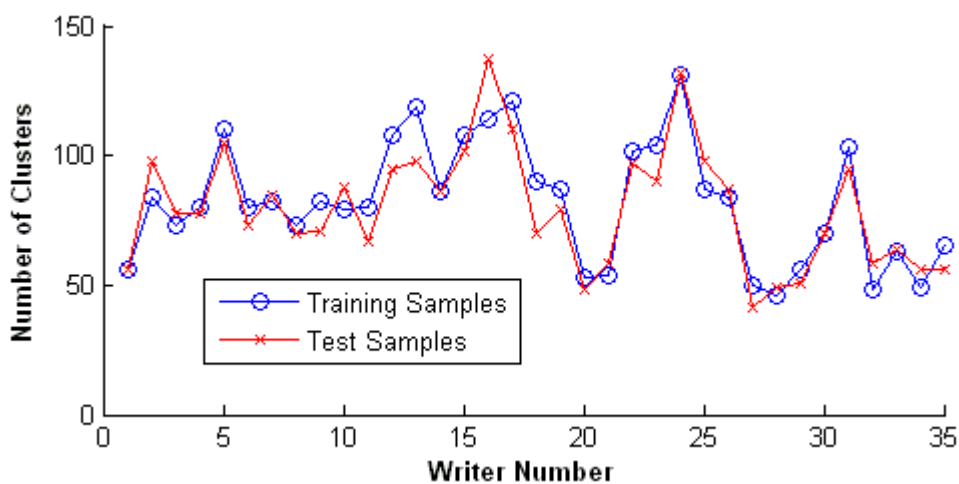


Figure 3.12 Number of clusters for two (training & test) samples of writers

An example of clusters obtained on a document image has been illustrated in Figure 3.13 along with the first five most frequent clusters (after applying PCA and reducing the dimensionality to 2) in Figure 3.14 showing that the clusters are well separable using the proposed representation.

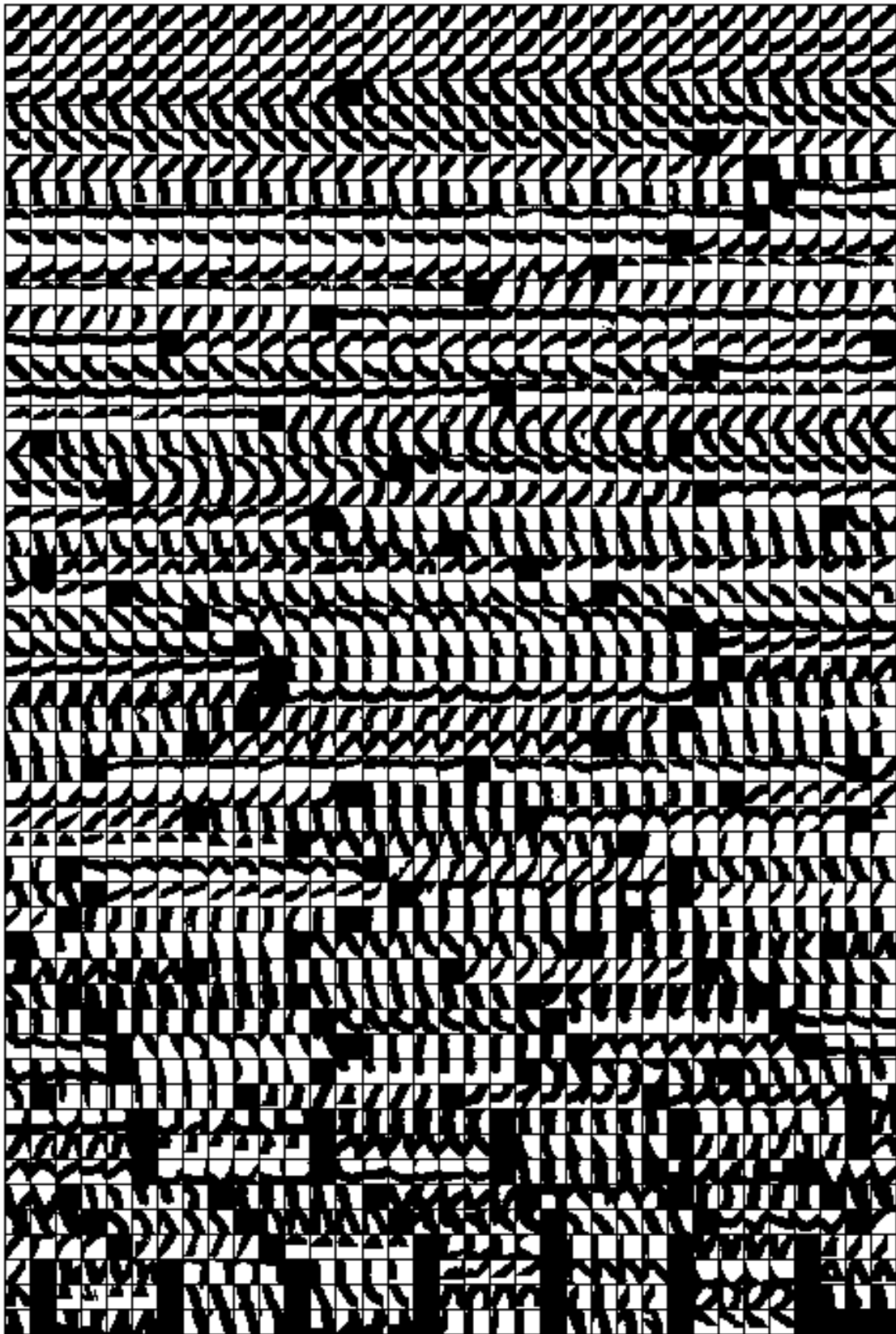


Figure 3.13 Clusters obtained on a document image

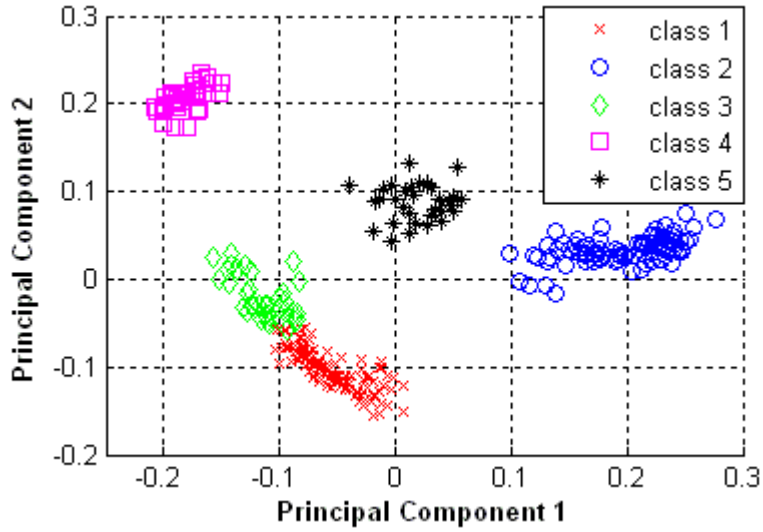


Figure 3.14 The first five most frequent classes (shown in Figure 3.13) after application of PCA and dimensionality reduced to 2

After having extracted the representative clusters that correspond to the frequent patterns of the writer of a document, we now need determine how this information could be employed to represent a writing. This representation would also depend upon the classification method used to compare two writings. We have chosen a representation that comprises certain properties of the extracted classes (clusters) and thus not dealing with the individual elements of each class, as explained in the following section.

3.5 Writing Representation

A document image D , having retrained M^D classes is represented by the corresponding set of classes as:

$$C^D = \{C_i | 1 \leq i \leq M^D\} \quad (3.5)$$

For each class C_i in C^D , we have:

$$C_i = \{S_{1,i}, S_{2,i}, \dots, S_{m,i}\}, \quad m = \text{card}(C_i) \quad (3.6)$$

$$\text{And each } S_{j,i} = (s_{j,i}^1, s_{j,i}^2, \dots, s_{j,i}^d)$$

We then estimate for each class, the probability of its occurrence $P(C_i)$:

$$P(C_i) = \frac{\text{card}(C_i)}{\sum_{j=1}^{\text{card}(C^D)} \text{card}(C_j)} \quad (3.7)$$

the mean vector \bar{S}_i :

$$\bar{S}_i = \frac{\sum_{j=1}^{card(C_i)} S_{j,i}}{card(C_i)} \quad (3.8)$$

and the covariance matrix Cov_i :

$$Cov_i = (X_i - \bar{S}_i)(X_i - \bar{S}_i)^T \quad (3.9)$$

With X_i being the matrix whose columns are the elements of class i , each sub-image is represented as a $d \times 1$ column vector, with $d=4n+6$. A document D is thus represented as:

$$D^r = \{F_i, 1 \leq i \leq card(C^D)\} \quad (3.10)$$

$$where: F_i = \{P(C_i), Cov_i, \bar{S}_i\}$$

We extract these features for each of the N documents in the training set and thus create a reference base R :

$$R = \{D_i, 1 \leq i \leq N\} \quad (3.11)$$

3.6 Writer Identification

In the previous sections, we presented the methodology for characterizing a writer by the frequent writing patterns. We would now like to study how good these patterns are in identifying the writer of an unseen text, written by any of the writers in the reference base. This is carried out by finding a similarity index between the questioned document and all the documents in the reference base, the writer of the unknown sample being identified as the writer of the document that maximizes the similarity index.

The first step towards writer identification is the extraction of features from the document whose writer is to be identified. We start with a binarization of the test image T followed by the division of text into small sub-images and then their clustering, as discussed in the previous sections. As a result of these steps, the questioned document T is represented by the frequent patterns of its author, a set of clusters C^T :

$$C^T = \{C_j | 1 \leq j \leq M^T\} \quad (3.12)$$

M^T being the number of clusters (classes) retained for document T .

We have seen in section 3.4.4 that for two samples written by the same author, the number of classes obtained by clustering is quite close to each other. Using this observation, if the difference between the number of classes of T and the reference document D is above a certain threshold, we straightaway discard D and proceed to the next document of the reference base.

$$\frac{|card(C^T) - card(C^D)|}{card(C^T)} < \eta \quad (3.13)$$

The parameter η is chosen large enough not to introduce any errors at this stage, i.e. we do not want that a reference document written by the same author as the test document be discarded by this filter. A value of $\eta = 0.5$ has been used in our experimentation. If a reference document satisfies condition 3.13, we proceed to the classification which is carried out using a nearest neighbour rule. We will present two approaches; we first employ the Bayesian classifier (to assign patterns in the test document to classes in the training document) and then use the probability distribution of the redundant patterns in a writing to characterize its writer.

3.6.1 Bayesian Classifier

The Bayesian Classifier is based on the assumption that decision problem can be specified in probabilistic terms and that all of the relevant probability terms are known. To find the probability that class i is present when feature X is observed, Baye's formula is given as:

$$P(C_i|X) = \frac{p(X|C_i)P(C_i)}{p(X)} \quad (3.14)$$

With $P(C_i|X)$ the posterior probability, $p(X|C_i)$ the class conditional probability (likelihood), $P(C_i)$ the prior probability and $p(X)$ the normalizing factor.

To find the class i that maximizes the probability of pattern X belonging to class C_i , the Baye's decision rule can be re-written as:

$$i_{Bayes} = \arg \max_i (P(C_i|X)) = \arg \max_i p(X|C_i) P(C_i) \quad (3.15)$$

In our case, the prior probability of each class $P(C_i)$ is known from the available training set while we assume the class-conditional probability density to have a Gaussian distribution for each class C_i .

$$p(X|C_i) = \frac{1}{(2\pi)^{d/2} |Cov_i|^{1/2}} \exp \left(-\frac{1}{2} (X - \mu_i)^T Cov_i^{-1} (X - \mu_i) \right) \quad (3.16)$$

$\mu_i = \text{Mean of class } i \text{ (} d \times 1 \text{ vector)}$

$Cov_i = \text{Covariance matrix of class } i \text{ (} d \times d \text{ matrix)}$

We thus employ Baye's decision theory under the assumption of multivariate Gaussian densities and the classifier is commonly known as a *Gaussian Classifier*. The discriminant function in equation 3.15 can thus be reformulated as:

$$d_{i(X)} = \log p(X|C_i) P(C_i) \quad (3.17)$$

Which is equivalent to:

$$d_{i(x)} = -\frac{d}{2} \log 2\pi - \frac{1}{2} \log |Cov_i| - \frac{1}{2} (X - \mu_i)^T Cov_i^{-1} (X - \mu_i) + \log P(C_i) \quad (3.18)$$

Leaving out the constant factor $-\frac{d}{2} \log 2\pi$, we are left with the following expression to maximize:

$$d_{i(x)} = -\frac{1}{2} \log |Cov_i| - \frac{1}{2} (X - \mu_i)^T Cov_i^{-1} (X - \mu_i) + \log P(C_i) \quad (3.19)$$

Coming back to our problem, we have the test document represented by the mean vector of each class as:

$$T^r = \{\bar{S}_j, 1 \leq j \leq \text{card}(C^T)\} \quad (3.20)$$

These vectors represent the patterns that have to be attributed to one of the classes of a reference document, defining the overall similarity between two documents as:

$$SIM(T, D) = \frac{1}{\text{card}(T^r)} \sum_{j=1}^{\text{card}(T^r)} \text{Max}_{C_i \in C^D} P(C_i | \bar{S}_j) \quad (3.21)$$

That is: for each \bar{S}_j in T^r , the objective is to find the class i of document D that maximizes the probability of \bar{S}_j belonging to class C_i . $P(C_i | \bar{S}_j)$ is estimated by 3.19 which can be re-written for our problem as:

$$d_{i(\bar{S}_j)} = -\frac{1}{2} \log |Cov_i| - \frac{1}{2} (\bar{S}_j - \bar{S}_i)^T Cov_i^{-1} (\bar{S}_j - \bar{S}_i) + \log P(C_i) \quad (3.22)$$

Where

$\bar{S}_i = \text{mean of class } i \text{ (} d \times 1 \text{ vector)}$

$Cov_i = \text{Covariance matrix of class } i \text{ (} d \times d \text{ matrix)}$

We calculate the similarity index between document T and all the documents in the reference base R and identify the writer of the questioned document as the author of the document maximizing the index (nearest neighbour classification, k -nn with $k=1$).

$$Writer(T) = Writer(\text{Arg max}_{D_i \in R} (SIM(T, D_i))) \quad (3.23)$$

3.6.2 Probability Distribution of Redundant Writing Patterns

We have the reference documents represented by a set of clusters, a family of basic redundant shapes with a known probability of emission for a particular writer. For a document D , we may consider these occurrence probabilities as a probability distribution h^D , where each bin in h^D would represent the emission probability of the respective shape by the author of document D . For the questioned document T , we may opt not to carry out the clustering procedure and assign each of

the segmented sub-images to one of the clusters in the reference document. To compare the query document T with a reference document D with M^D clusters, we build a histogram h^{DT} with one bin allocated to each cluster of D . For every pattern p (represented by its respective feature vector) in the test document, its nearest cluster is found using the Euclidean distance and the occurrence is counted in the respective histogram bin:

$$b = \arg \min_j \left(\text{dist}(p, C_j) \right); \quad h_b^{DT} \leftarrow h_b^{DT} + 1 \quad (3.24)$$

$$j = 1, \dots, M^D; \quad M^D = \text{card}(C^D)$$

Thus in fact, the questioned document is represented in the feature space of the reference document and the distance between the two documents is computed by calculating the (χ^2) distance between the respective distributions h^D and h^{DT} :

$$D(D, T) = \sum_{j=1}^{\text{card}(C^D)} \frac{(h_j^{DT} - h_j^D)^2}{h_j^{DT} + h_j^D} \quad (3.25)$$

The writer of T is finally identified as the writer of the document D_i that reports the minimum distance.

3.7 Writer Verification

For writer verification, we compute the distance (or similarity) between two given images. Distances that are less than a predefined decision threshold are viewed as sufficiently low for considering that the two samples have been written by the same person. Beyond the threshold value, we consider the samples to be written by different writers. We can thus have, like for all biometric verification tasks, two types of errors:

False Acceptance: Concluding that two samples are written by the same person when, in fact they are not.

False Rejection: Concluding that two samples are written by different persons when, in fact they are written by the same person.

The respective error rates are termed as False Acceptance Rate (FAR) and False Rejection Rate (FRR). Evidently, the FAR and FRR are dependant on the chosen threshold. Increasing the threshold value, the FAR will increase, while FRR will decrease and vice versa (Figure 3.15). This relationship between the FRR and FAR as a function of the threshold value is normally depicted graphically by Receiver Operating Characteristic (ROC) curves. The ROC curves are computed by varying the acceptance threshold, the performance of the biometric system being quantified by the Equal Error Rate (EER): the point on the curve where the False Acceptance Rate (FAR) equals the False Rejection Rate (FRR). Smaller the Equal Error Rate (EER), more precise the system is. For

our writer verification system, we will present the equal error rates and, in some cases, illustrate the complete ROC curves.

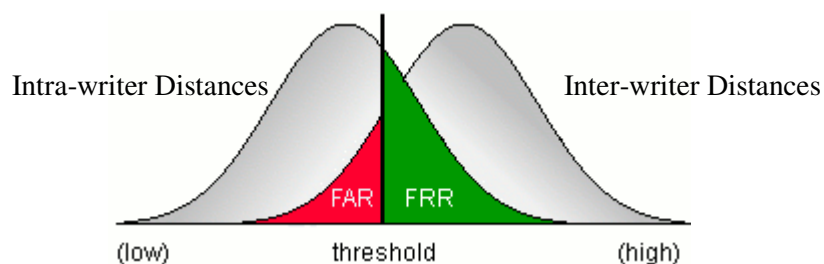


Figure 3.15 FAR and FRR on the distribution of distances

3.8 Redundant Writing Patterns: Universal Code Book

After having presented the study of frequent writing patterns for an individual leading to the generation of a writer specific code book, we will now extend the same principle to a universal set of basic shapes as proposed at the grapheme level in [Bulacu & Schomaker, 2007] and [Bensefia et al., 2005b]. As we discussed earlier (in section 2.3.2.3), the main difference between the two approaches is that in [Bulacu & Schomaker, 2007] the authors have generated a fixed size (set to 400) code book from an independent dataset (of 65 writers) while in [Bensefia et al., 2005b] the authors have clustered all the graphemes of the database under study using a sequential clustering algorithm. Examples of redundant graphemes in the two studies have been illustrated in Figure 3.16. It should be noted that the code book is not meant to represent an exhaustive list of all possible allographic shapes of a particular script and alphabet. Rather, its objective, as described in [Bulacu & Schomaker, 2007], is to *'span a shape space and act as a set of nearest-neighbour attractors for the graphemes extracted from a given writing sample'*.

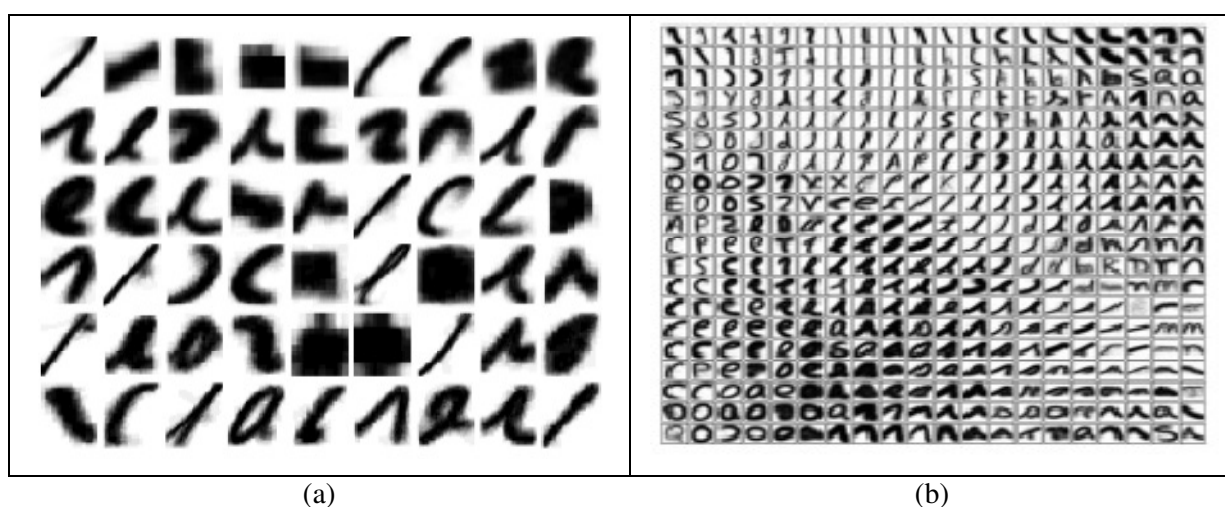


Figure 3.16 Examples of the invariant patterns obtained in studies: (a) [Bensefia et al., 2005b] and (b) [Bulacu & Schomaker, 2007] (Images reproduced from the cited references)

We will apply the principle of universal code book generation on the small writing fragments, as per the segmentation method discussed in section 3.3. The code book is produced from 50 handwritten samples (selected from the data set RIMES [Grosicki et al., 2008]) and the fragments are clustered using k-means algorithm in the feature space (section 3.4.1) with k being varied from 50 to 750 and finally fixed to 100 after evaluations on the validation set. Figure 3.17 shows an example code book (grey levels correspond to intra-cluster variability) and comparing it with the ones in Figure 3.16 where one can notice the presence of graphemes that correspond to complete characters (e.g. ‘a,’ ‘e’, ‘c’, ‘l’ ‘s’, ‘p’ etc), the shapes that we study are much more elementary.

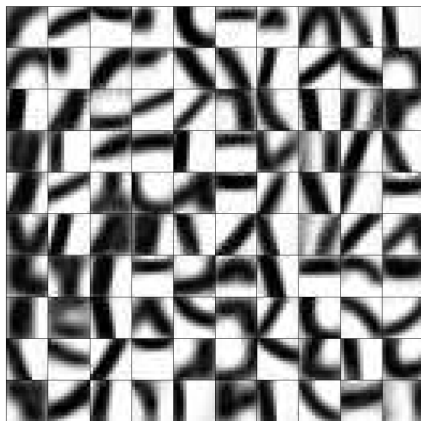


Figure 3.17 Code book of size 100 generated from 50 samples of the dataset RIMES

For a document image, we initialize a histogram with all bins set to zero, the number of bins being equal to the size of the code book. For each of the extracted sub-images (patterns) in the document, we find its nearest prototype in the code book and count it in the respective histogram bin as explained in section 3.6.2, the only difference being that instead of comparing the patterns to the code book of each writer, we now compare them to a common code book. The distribution is used to characterize the writer and is computed for all the documents in the training set.

$$b = \arg \min_j \left(\text{dist}(p, C_j) \right); h_b^D \leftarrow h_b^D + 1 \quad (3.26)$$

$$j = 1, 2, \dots, \text{Code book size}$$

For a query document, we repeat the same procedure and find its corresponding distribution h^T . Two documents are then compared by computing the (χ^2) distance between their respective distributions. Figure 3.18 shows the distribution of the probability of occurrence of the code book entries for two different text samples written by the same writer. The two curves more or less follow similar forms exhibiting peaks and valleys for the same code book entries and showing that their comparison might be useful in identifying the writer of a document.

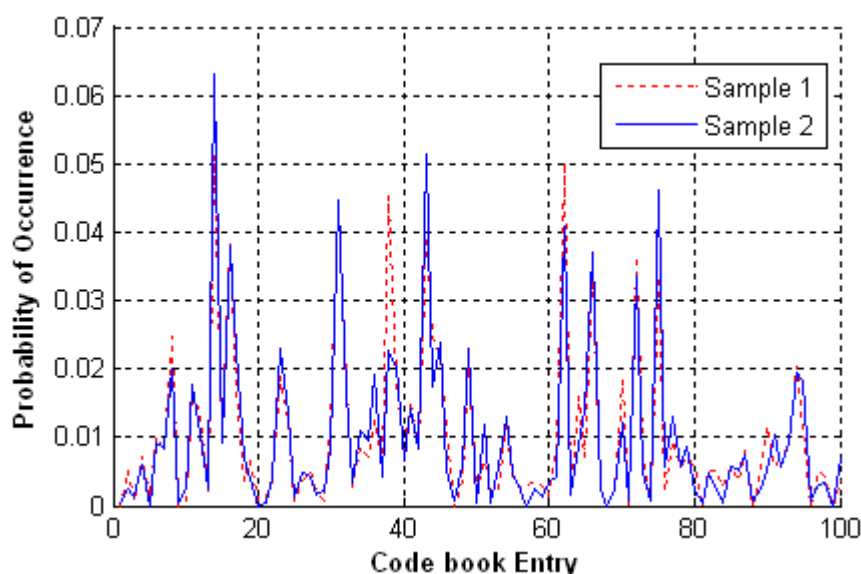


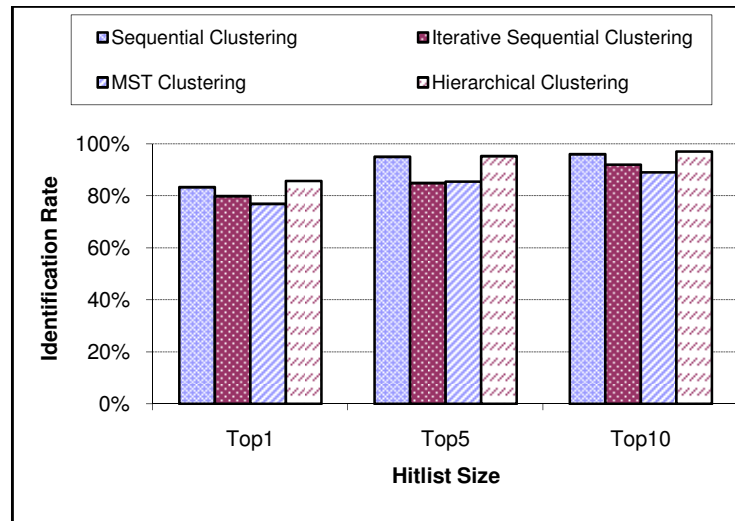
Figure 3.18 Probability distribution for two samples of the same writer when using a universal code book

3.9 Experimental Results

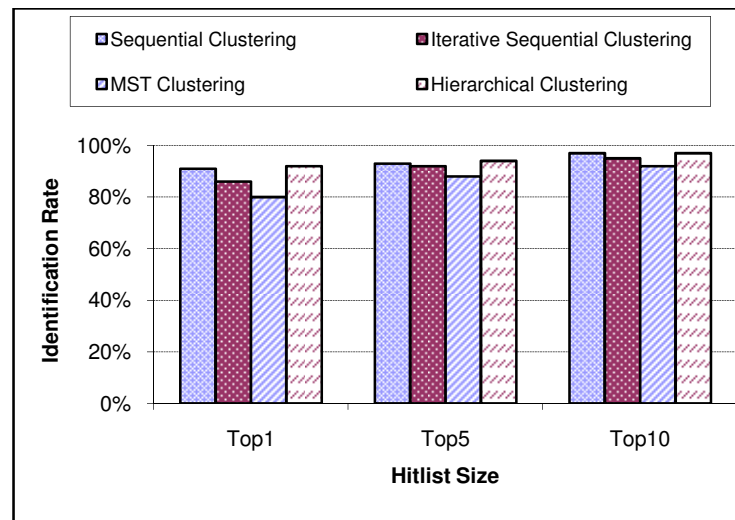
We will now present the experiments that we conducted in order to evaluate the proposed methodology on writer recognition. These experiments have been carried out on the IAM dataset as presented in section 3.1. We will first report the results on writer-specific code book approach and then present the system performance employing the universal code book. It should also be noted that these results represent only a subset of our series of experiments.

3.9.1 Writer-specific code book

We first evaluate the performance of different clustering methods using the two classifiers. The threshold value for each of the methods was optimized on the validation set and the method was then evaluated on the first 150 writers of the IAM database. The results as presented in Figure 3.19 reveal that although the overall identification rates vary for the two classifiers, the performance of the clustering methods is more or less consistent with hierarchical clustering out performing the rest achieving an identification rate of 86% (Bayesian classifier) and 92% (χ^2 distance) on the two classifiers. The sequential clustering performs approximately equally good (identification rates of 83% and 91% respectively) while the multi-phase (iterative) sequential clustering, that produced very fine clusters, falls behind the two. The relatively degraded performance of the MST clustering might be due to the existence of small inter-class distances, grouping two or more clusters into one big cluster and thus resulting in a relatively weak performance.



(a) Bayesian Classifier



(b) χ^2 Distance

Figure 3.19 Writer identification results for different clustering methods (150 writers)

Regarding the two classifiers, comparing the distribution of redundant patterns across two writings using χ^2 distance exhibits significant performance improvements. Of course an improved performance is also linked to the fact that while representing the questioned writing in the reference document space (computing the probability distribution of writing fragments in the query writing) we employed the complete set of patterns while in case of Bayesian classifier, only the mean vector of each cluster of the test document is attributed to one of the candidate classes in the reference document. Employing the same protocol across the two classifiers reveals that χ^2 distance still outperforms the Bayesian classifier hence we carry out the subsequent experiments on the complete data set by employing hierarchical clustering as the regrouping method and χ^2 distance between the probability distributions of writing fragments as classifier. The results are summarized

in Table 3-2 where we achieve an identification rate of 81% and an equal error rate of 5.44% on the entire data set.

**Table 3-2 Identification and verification performance based on writer specific code book
(Clustering: Hierarchical, Classifier: χ^2 Distance)**

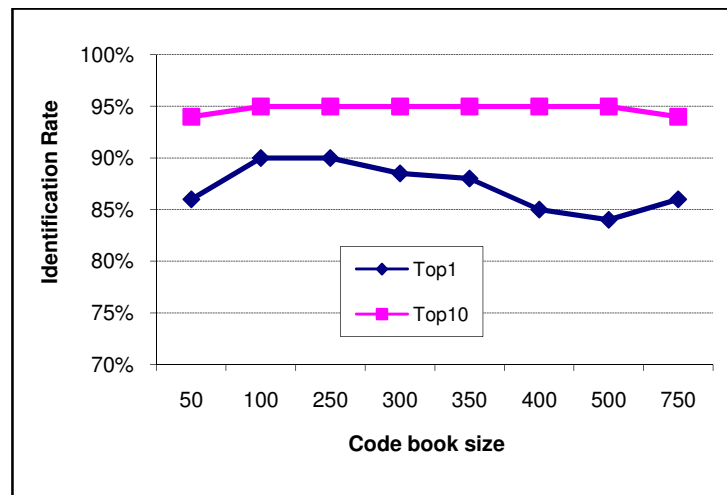
Performance	Identification		Verification
	Top1	Top10	EER
150	92	97	5.27
300	85	94	4.67
650	81	94	5.44

3.9.2 Universal Code book

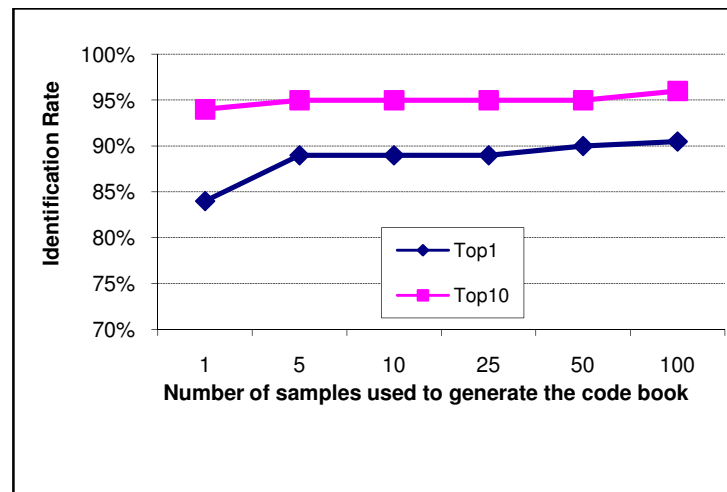
For the universal code book, Table 3-3 summarizes the identification and verification results. An identification rate of 84% and an equal error rate of 4.49% are realized when using the complete set of 650 writers. These results have been achieved by generating the code book of size 100 from 50 different writing samples from the RIMES data set. We also study the performance evolution with respect to the number of writing samples used to generate the code book as well as the size of code book. This is carried out (on 300 writers of the data set with two samples each) by fixing one of these two parameters and varying the other as illustrated in Figure 3.20. It can be noticed that performance shows a consistence decrease as the size of the code book increases beyond 250. Another interesting observation is that the number of samples used to generate the code book does not cause a dramatic change in the performance. Employing only one writing sample to produce the code book results in an identification rate of 84% which further validates the argument that the code book needs not to be exhaustive.

Table 3-3 Identification and verification performance based on the universal code book

Performance	Identification		Verification
	Top1	Top10	EER
150	93	97	4.26
300	90	95	5.12
650	84	96	4.49



(a) Code book generated from 50 sample image



(b) Size of code book fixed to 100

Figure 3.20 Writer identification performance (on 300 writers) as a function of (a) code book size and (b) the number of samples used to generate the code book

3.9.3 Discussion

We have presented the results of the two approaches to exploit the redundancy in writing, a writer-specific and a universal code book. A comparative analysis of the performance of the two reveals that representing writings into a common code book space leads to better results on writer identification and verification as compared to representing them in a writer-specific space (identification rate of 84% against 81% and EER of 4.49 % against 5.44%). However, as we discussed earlier, the methods based on a writer-specific code book are more adaptive to alphabet change as opposed to the ones based on a universal code book.

It would also be interesting to compare our results with the ones achieved by generating a code book at the grapheme level. Table 3-4 presents an overview of the performance on writer identification task in these studies. For the writer-specific code book [Bensefia et al., 2002] reports

an identification rate of 98% on 88 writers but since the dataset employed is not the same, the comparison would not be meaningful. Later studies [Bensefia et al., 2005b] by the same authors achieve an identification rate of 86% on 150 writers employing a code book generated from the entire data set. The best performance so far has been reported in [Bulacu & Schomaker, 2007] where the authors achieve an identification rate of 80% with a code book generated from an independent data set. (It should be noted that since our objective is to compare the performance of different code book based methods, this 80% represents the identification rate achieved by using the code book and not the highest rate achieved by authors by combining different other features. The performance of their complete system would be compared in the next chapter).

By changing the observation scale from grapheme to small fragments, we achieve an identification rate of 84%. It is however, important to precise that the evaluation criterion in [Bulacu & Schomaker, 2007] is not the same as ours. We have distinguished the training and test sets while in [Bulacu & Schomaker, 2007] the authors have used a leave-one-out approach on the entire data set. Thus, in order to present an honest comparison we also carried out a similar experimentation and achieved an identical identification rate of 80%. Our method however relies on a much smaller code book size and the patterns contributing to the code book have been issued from a very generic segmentation scheme.

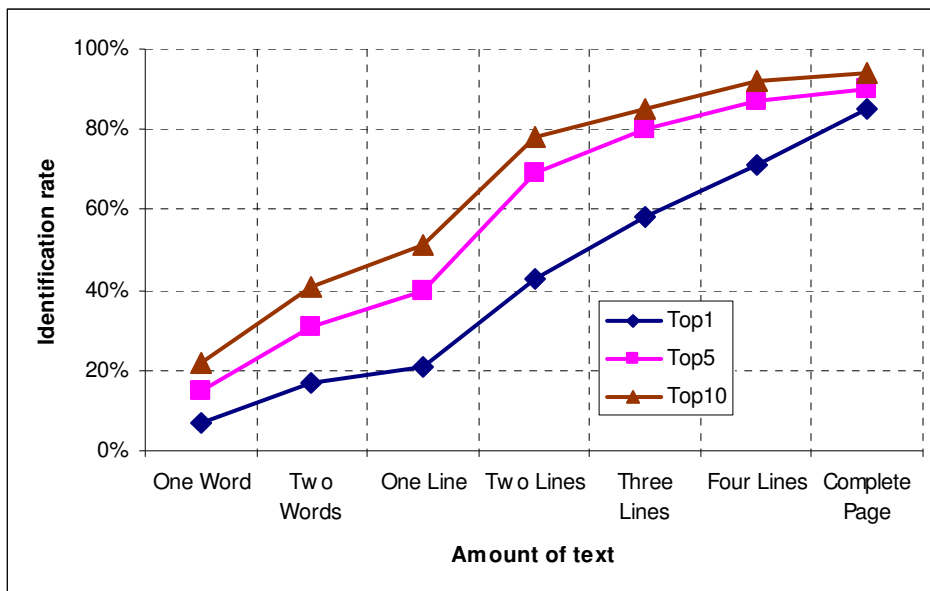
Table 3-4 Performance comparison of code book based methods

	Code book	Dataset	Writers	Performance
[Bensefia et al., 2002]	Writer-Specific	PSI	88	98%
[Bensefia et al., 2005b]	Universal	IAM	150	86%
[Bulacu & Schomaker, 2007]	Universal	IAM	650	80%*
Our method	Writer-Specific	IAM	150	92%
			650	81%
	Universal		150	93%
			650	84% / 80%*

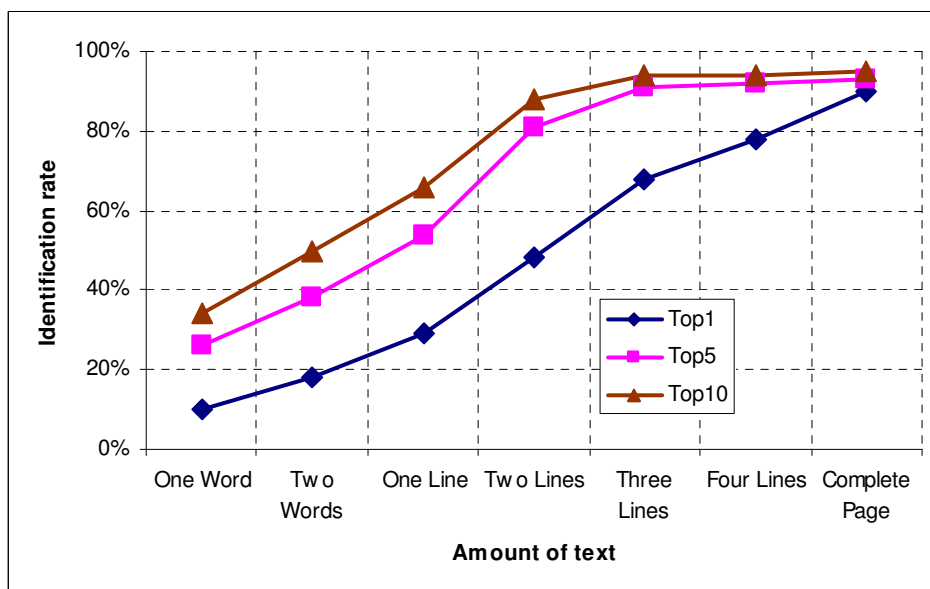
Finally, we would be interested to study how the performance of the two approaches varies with respect to the amount of text present (for training and testing). This study is carried out on the 300 writers of the IAM base who contributed at least two pages. This allows varying the amount of text from one word to complete page (instead of half a page if we also use the 350 writers having contributed only one page). The words and lines are extracted by using the ground truth information provided with the data base. The corresponding writer identification rates for writer-specific and universal code books have been summarized in Figure 3.21. Naturally, the universal

* k - nn using a leave-one-out approach

code book performance is relatively less sensitive to the amount of text where the identification rates rise from about 10% for a single word to 90% for the complete page which may vary from 5 to 10 lines on the average. The performance begins to stabilize a bit from three lines of text onwards, specially the Top-5 and Top-10 performances.



(a) Writer specific code book



(b) Universal code book

Figure 3.21 Identification rate as function of amount of text on 300 writers of IAM data set

3.10 Conclusion

We have shown, in this chapter, how the frequent writing patterns can be employed to characterize the writer of a handwritten text sample. Contrary to the classical approaches which exploit this

redundancy at the grapheme level, we studied writings at a much smaller observation scale extracting clusters of small writing fragments first for each of the individuals separately (writer specific code book) and then for the entire set of writers (universal code book). Thus the notion of redundancy in our case is more linked to the redundancy of writing gestures rather than that of the graphemes or allographs. The performance of the proposed methodology on writer recognition task is comparable to the best results reported so far in the literature for methods based on the idea of redundancy of writing patterns. We have however achieved these results employing a code book of much smaller size. In addition, the segmentation scheme that we employed is quite generic and could be applied to non-Latin scripts as well. In an attempt to improve the recognition rates further, we will now try to employ the visual attributes of writing to characterize its author. The features based on these attributes will make the subject of next chapter.

Chapter 4

Writer Characterization: Contour Based Features

After having studied the redundant patterns of writing and their effectiveness in characterizing the writer, we will now turn to a different aspect of writing and analyze the different visual attributes of writing that allow to capture the writing style of its author. The most important of these attributes is the overall writing orientation (slant) which is known to be a stable parameter [Maarse & Thomassen, 1983] with the assumption that the writing under consideration represents the natural writing style of the writer and is not a forged one (as the most common disguising ploy is to change the writing slant [Nickell, 2007][Koppenhaver, 2007]). In most of the cases, merely by looking at two handwritten samples, one can instinctively conclude that they come from different writers, without the need to read what is written and, in some cases, even without the alphabet knowledge of the script written (Figure 4.1). Besides orientation, curvature is known to be another fundamental characteristic of handwriting [Lee, 1999] and the features based on curvature have shown effective performance in characterizing writing styles [Joutel et al., 2007b] and writers [Bulacu et al., 2003] and, recognition of characters [Legault & Suen, 1992] [Miura et al., 1997] and numerals [Yang et al., 2005]. Inspired by the power of these attributes of handwriting, we endeavoured to design a set of features that would capture these characteristics and allow us to characterize the writer of a handwritten text sample. This chapter is devoted to the discussion of these features.

We will first introduce the proposed features that are computed from the contours of handwritten images by employing different representations of contours. The strength of these features for writer recognition is then highlighted by carrying out a series of evaluations, first employing the individual features and then their various combinations. A statistical study on the stability of these features with respect to certain parameters is then presented. We also explore the combination of these contour based features with the code book based features introduced in the previous chapter. Finally, we study the relevance of the proposed features by employing a feature selection mechanism.

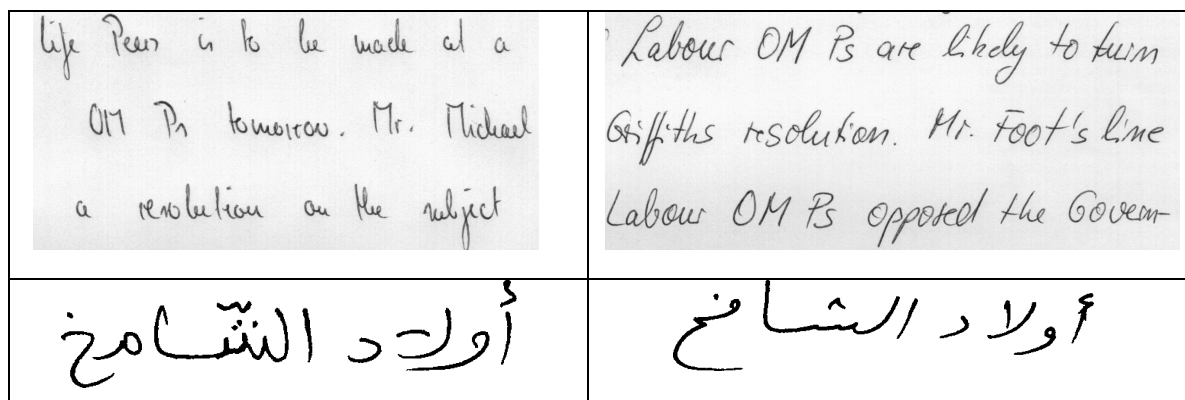


Figure 4.1 An example of instinctively distinguishable writings

4.1 Feature Extraction

Features in automatic analysis of handwriting are generally based on two categories: skeleton and contours, and over the years each of these has received considerable research attention [Chung & Wong, 1998] [Madhvanath & Govindaraju, 1997] [Chiu & Tseng, 1997] [Ke Liu & Suen, 1999]. For handwriting recognition, writer-dependent variations between the character shapes need to be eliminated and skeletonization generally is a good choice to represent a writing. Writer recognition on the other hand relies on these variations which are preserved by the contours, encapsulating the writing style of its author. We therefore chose to extract our features from the contours of the handwritten text images. In addition, human eye, that can instinctively distinguish two different writings, is mostly sensitive to contours and changes, validating our choice of working on the contours.

In order to extract the contours, we first need to binarize the document image which is carried out using Otsu's global thresholding (as discussed in section 3.2). We then perform connected component detection (using 8-connectivity) and for each of the connected components we extract the interior and exterior contours (Figure 4.2). Each contour is a sequence of consecutive points located on the ink-background boundary:

$$Contour_i = \{p_j | 1 < j \leq M_i, p_1 = p_{M_i}\} \quad (4.1)$$

With M_i being the length of contour i .



Figure 4.2 Images and their contours

Once the writing shapes are replaced with their respective contours we need to represent the extracted contours in a form that will facilitate the extraction of features. We have chosen to represent the contours in two ways that correspond to two different levels of writing details :

- i) By the well known Freeman chain codes [Freeman, 1974]
- ii) By a set of polygons approximating the contours

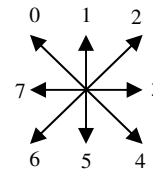
We first find the chain code sequence of the contours in a handwritten text image and extract a set of features at a global as well as a local level of observation. We also approximate each contour by a polygon and compute features from the straight line segments estimating the contours. The features extracted from the two representations are a set of distributions and as discussed earlier, they are mainly aimed to capture the orientation and curvature information in a writing, the two most important visual aspects that enable humans instinctively discriminate between two writings. We will now discuss the proposed features and the pros and cons of each in the sections to follow.

4.1.1 Chain Code Based Features

Chain codes have shown effective performance for shape registration [Ahmad et al., 2003] and object recognition [Bandera et al., 1999] and since the handwritten characters issued by a particular writer have a specific shape, chain code based features are likely to work well on tasks like writer recognition as well.

We calculate the Freeman chain code associated with each contour, thus representing it by the sequence:

$$\{c_j | 1 < j \leq M_i - 1\} \text{ where } c_j \in \{0,1, \dots, 7\}$$



The boundary pixels in the original binary image I are then labelled by their respective codes (Figure 4.3). We then proceed to the extraction of features from the newly formed image I^c . The contours are analyzed both at global and local levels. At global level, to remove errors due to a false ordering of the contour pixels, we employ the histograms of chain codes and their variants. Since only a global analysis might not be enough to distinguish two writings, at the local level, we analyze small handwritten fragments and compute certain features. Finally the set of extracted features is used to characterize a handwritten sample.

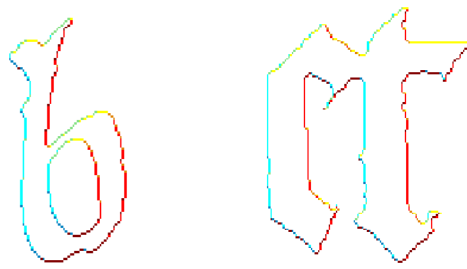


Figure 4.3 Contours replaced by direction codes (Colours represent codes)

4.1.1.1 Global Features

We will now present the chain code based global features which are in fact a set of distributions (histograms) as discussed in the following.

a) Distribution of Chain Codes

In order to capture the overall orientation information in the writing, we compute the well-known slope density function (histogram of all the chain codes/slopes f_I) of the contours. The eight bins of the histogram represent the percentage contributions of the eight principal directions in an individual's writing. Since the images are offline, we do not have the drawing order of the strokes hence whether a stroke is considered forward or backward is dependent on the way the contour is traced. A solution could be to quantize the histogram into four bins representing the four principal stroke directions: horizontal, vertical, left-diagonal and right-diagonal. Our experience however has shown that keeping the eight bins and being always consistent in the way a contour is traced is a better choice. Figure 4.4 illustrates the distribution of chain codes computed from two samples each, coming from two different writers. One can instinctively notice the overall vertical orientation in samples 'a' and 'b' which is reflected by two peaks at the respective bins of the corresponding histograms. Similarly, for samples 'c' and 'd' the peaks can be observed at the bins corresponding to the right-diagonal directions.

b) Distribution of Differential Chain Codes

The histogram of chain codes is invariant towards different deformations but the most obvious limitation of the chain code histogram is that two totally different shapes can have similar histograms. This problem is dealt with by encoding not only the direction, but also the differences in successive directions: differential chain codes, computed by subtracting each element of the chain code from the previous one and taking the result modulo d , where d is the connectivity (8 in our case). Thus we get more information on the contour curve. The differential chain code at pixel p_i represents the angle θ_i (as indexed in Table 4-1) between the vectors $p_{i-1}p_i$ and $p_i p_{i+1}$ (as shown in

Figure 4.5) and their histogram f_2 (also known as curvature density function) is used as the second feature to represent a handwritten text.

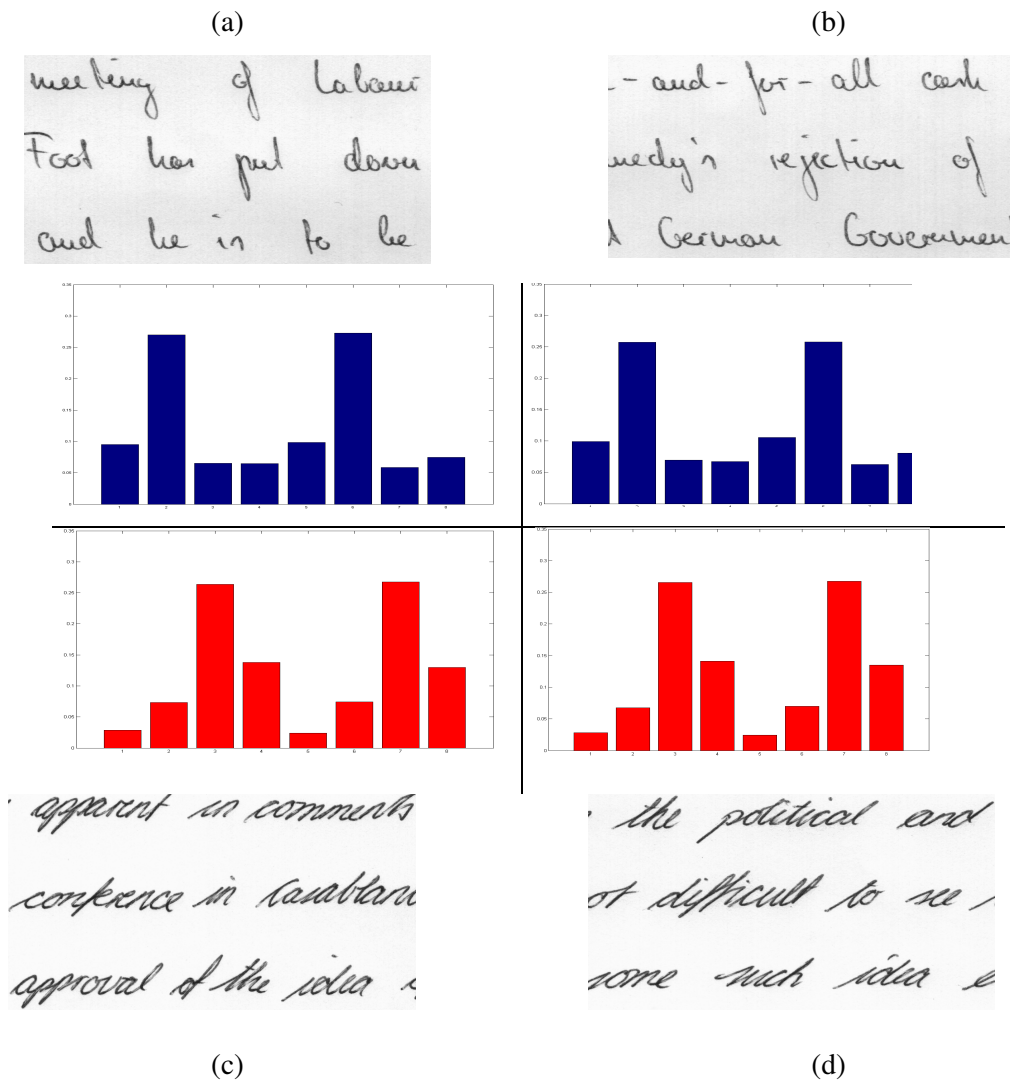


Figure 4.4 Distribution of chain codes on writing samples from two writers

Table 4-1 Differential Chain Codes and the corresponding Angles

$(c_{i+1}-c_i) \bmod 8$	0	1	2	3	5	6	7
θ_i	180°	135°	90°	45°	315°	270°	225°

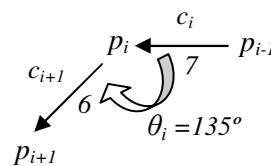
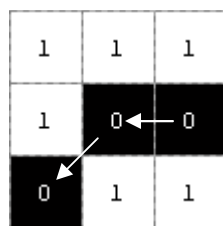


Figure 4.5 Angle θ_i at pixel p_i

Employing the same principle, we also compute the histogram f_3 from the second order derivative of the chain code C , which gives us an estimate of the variation of the angles as the contour progresses.

c) Distribution of Chain Code Pairs and Triplets

The slope and curvature histograms could serve well to provide a crude idea about the handwritten shapes but are insufficient to capture the fine details in writing. We thus propose to count not only the occurrences of the individual chain code directions but also the chain code pairs. We scan the chain code sequence and for each pair (i,j) we increment the bin (i,j) of the histogram f_4 . We next employ the same principle on chain code triplets, that is; we define a three dimensional histogram (f_5) where the bin (i,j,k) of the histogram represents the percentage contribution of the triplet i,j,k in the chain code sequence of the contour of a handwriting image. This gives us two matrices of sizes 8×8 and $8 \times 8 \times 8$ respectively. All the possible 64 pairs and 512 triplets, however, cannot exist while tracing the contours. Consider for example, the four types of L-junctions in Figure 4.6 that may arise while tracing a contour. Starting at any of the three junction pixels, depending upon the way the directions are prioritized while contour scanning, we can have only one of the two possible pairs, as explained for one of the junctions in Figure 4.7 (For simplicity, we have replaced the chain code sequence 0–7 by 1–8). Similarly, during the contour trace, moving from pixel p_{i-1} to p_i , we cannot move back from p_i to p_{i-1} and thus the corresponding chain code pairs do not exist.

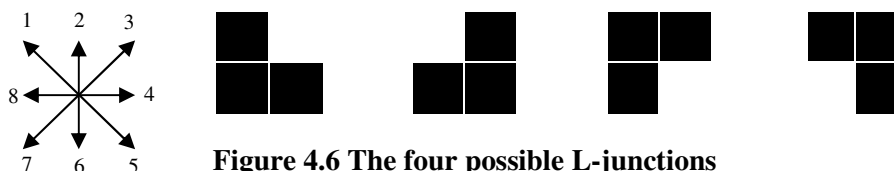


Figure 4.6 The four possible L-junctions

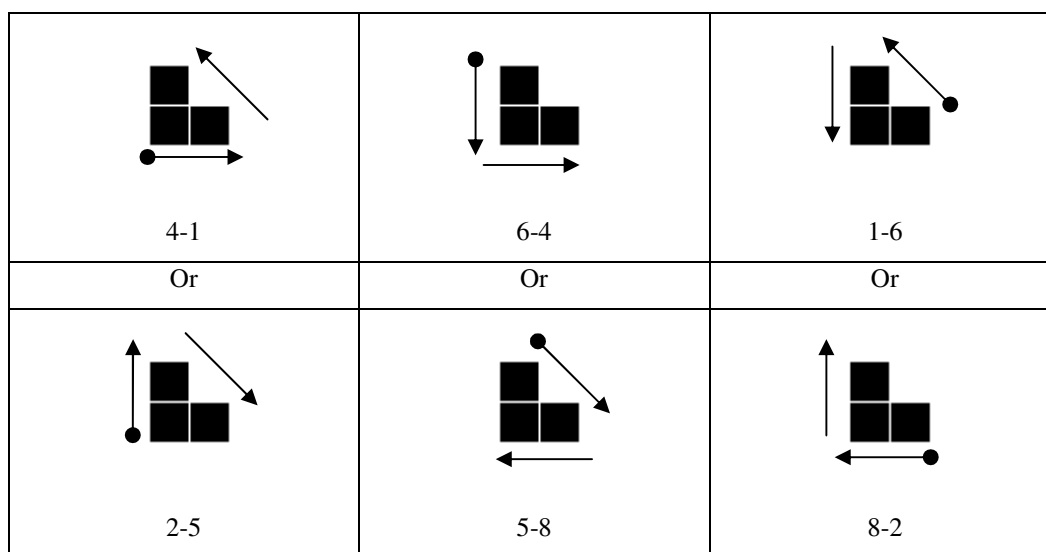


Figure 4.7 Two possible chain code pairs, starting at each of the three pixels of an L-junction

Summarizing, we can have a total of 44 ($64 - (12 \times 3 + 8)$) possible chain code pairs. In a similar fashion, for the chain code triplets, without presenting the details, we would like to resume by precisizing that out of the 512 possible triplets, only 236 exist. An example of two writings and their respective distributions of chain code pairs has been illustrated in Figure 4.8.

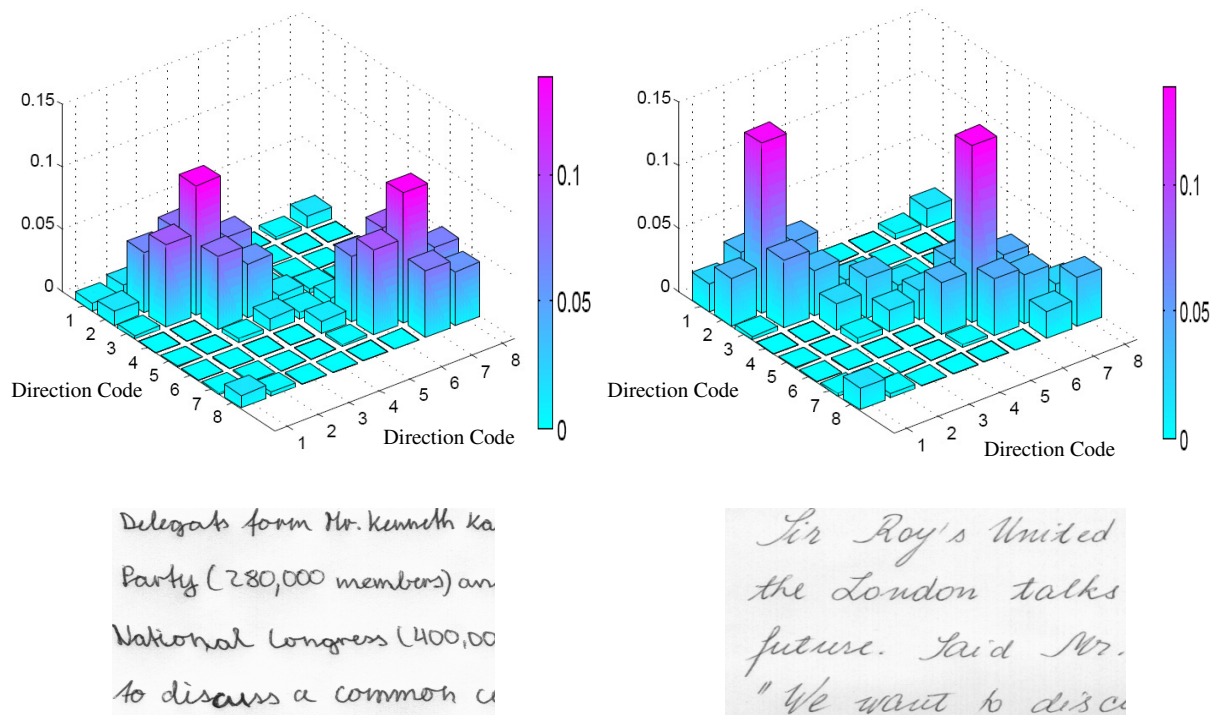


Figure 4.8 Two writings and their respective distributions (normalized) of chain code pairs

It is important to precise that there exists a partial redundancy between the distribution of first order differential chain codes ($f2$) and the distribution of chain code pairs ($f4$) as well as between the distributions of second order differential codes ($f3$) and the code triplets ($f5$). This will be discussed towards the end of this chapter.

d) Distribution of Curvature Indices

To get some visual feeling from the text, curvature is an interesting property to study and is very much linked with the physiological way the strokes are written, the strength involved by the muscles and the way they are operated. The estimate of curvature that we achieve from the differential chain codes and the code pairs and triplets is very local (span of few pixels only). Therefore, an approximation of curvature that is based on a relatively larger neighbourhood of a contour pixel would be a better indicative of the stroke curves. We thus employ a chain code histogram based estimate of curvature, presented in [Bandera et al., 1999] for object recognition where a correlation measure between the distributions of directions on both sides (forward and backward) of a contour pixel p_j is used to approximate the curvature at p_j .

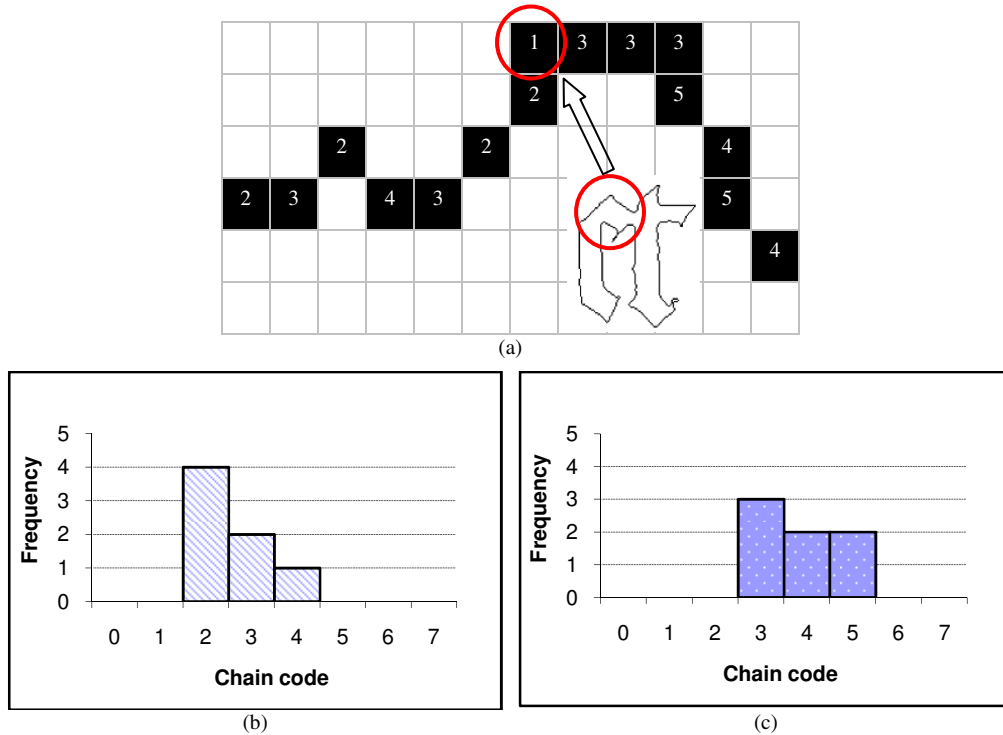


Figure 4.9 a) Contour pixel p_j (with code c_j) of a character b) Backward histogram ‘ b ’ at p_j c) Forward histogram ‘ f ’ at p_j

For each c_j in image I^c , we take K forward and K backward neighbours (K being linked to the height of the character base line and fixed to 7 in our case) and calculate two histograms (f and b) representing the orientation of the segments on both sides of c_j (Figure 4.9). The curvature index at c_j is then estimated by the reciprocal of the correlation coefficient between the forward and the backward histograms.

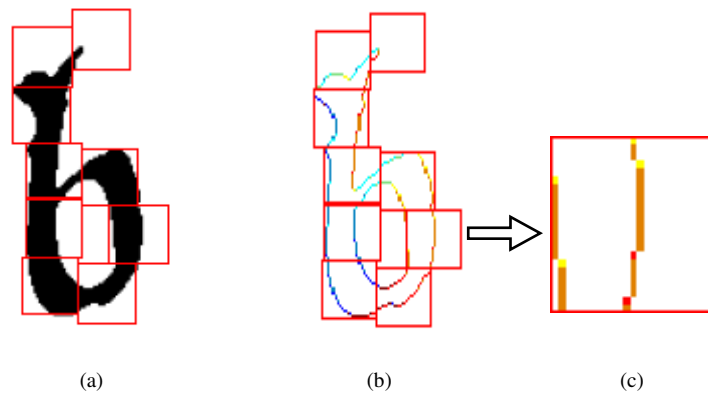
$$\rho = \frac{\sum_{i=0}^7 (f(i) - m_f)(b(i) - m_b)}{\sqrt{\sum_{i=0}^7 (f(i) - m_f)^2 (b(i) - m_b)^2}} \quad (4.2)$$

With f and b being the forward and backward histograms while m_f and m_b their mean values respectively. A high value (close to 1) of the correlation coefficient characterizes similar histograms and hence a low curvature index and vice versa. The correlation coefficients are calculated for each point of the contour sequence and are counted in a histogram $f6$. The histogram is partitioned into 11 bins determined empirically on the validation set.

After having defined the features that bring some global information on the directions of the strokes, the evolution of directions and on the curvature of drawings, we now introduce some local features.

4.1.1.2 Local Features

The features $f1 - f6$, although computed locally, capture the global aspects of writing thus the relative stroke information is lost. We therefore chose to carry out an analysis of small stroke (contour) fragments as well. These fragments are chosen in a similar fashion as presented in section 3.3. The fragments we consider are parts of the handwritten text image contained in small windows that are positioned over the image employing an adaptive window positioning algorithm (section 3.3.3), the fragments under consideration thus do not carry any semantic information. The windows positioned over a text image and the corresponding contour image (pixels labelled by respective codes) have been illustrated in Figure 4.10.



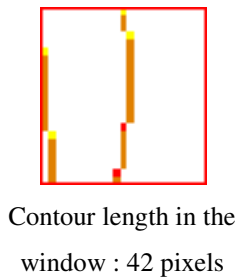
**Figure 4.10 a) Windows positioned over text b) Windows positioned over the contour image
c) Directional distribution in a window**

For each window, we determine how the 8 directions are distributed with respect to the total contour length within the window, the percentages being divided into p intervals. We build an accumulator (local stroke direction distribution $f7$) which is a two dimensional $d \times p$ array where d is the connectivity (8 directions). The accumulator is initialized with all bins set to zero. For each window w , containing the chain code sequence C^w , the bins (i,j) of the histogram (accumulator) are incremented by 1 if the direction i is represented in the j^{th} interval, where j is given by:

$$j = \text{ceil}\left(\frac{\text{card}(C_i^w)}{\text{card}(C^w)} \times 100\right) \quad (4.3)$$

$$\text{With: } C_i^w = \{c_k \in C^w | c_k = i\} \text{ And } i = 0, 1, \dots, 7$$

The process has been illustrated in Figure 4.11 where the three directions encountered in the window result in incrementing the respective bins of the distribution $f7$. The distribution is finally normalized by the sum of all the entries as illustrated for a handwritten image in Figure 4.12. Naturally the first half of percentage axis is much denser than the second one.



Direction (c)	Number of Pixels in Direction c	Percentage of Contour Length	Interval	Bin to Increment
5	4	9%	1	5,1
7	2	5%	1	7,1
6	36	86%	9	6,9
	42	100%		

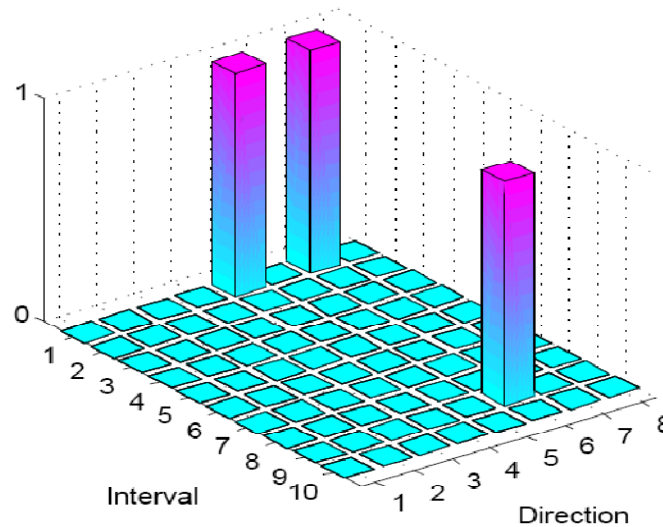


Figure 4.11 Contribution of a window to the stroke direction distribution

The number of intervals p should be large enough to capture the difference in the distribution of directions in the windows and small enough to allow marginally different distributions contribute to the same intervals of $f7$. We have made p vary from 2 to 20 on the validation data set and finally chosen a value of p equal to 10 for our system.

Using the same principle, we calculate the distributions $f8$ and $f9$, superimposing the windows on images $I^{c'}$ and $I^{c''}$ generated by labelling the contour pixels by their first and second order differential chain codes respectively. These distributions could in fact be viewed as the local variants of $f2$ and $f3$.

Summarizing, these chain code based features compute the orientation and curvature information of writing. However, these estimates are computed from raw pixels and it would be interesting to carry out a similar analysis at a different observation level. We therefore propose to estimate the contours by a set of polygons and then proceed to feature extraction (a set of global features) which not only corresponds to a distant scale of observation but the computed features are also more robust to noise distortions. These features have been discussed in the following section.

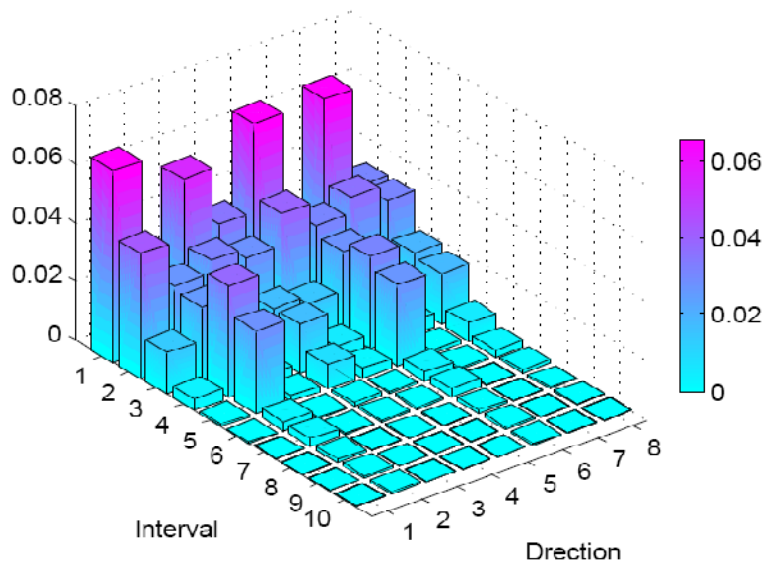


Figure 4.12 Normalized local stroke direction distribution of a writing

4.1.2 Polygon Based Features

These features are aimed at keeping only the significant characteristics of writing (enough to characterize its author) discarding the minute details. For this purpose, we carry out an estimation of the writing contours by a set of line segments, employing the sequential polygonization algorithm presented in [Wall & Danielsson, 1984]. The algorithm requires a user defined parameter T that controls the accuracy of approximation. Larger values of T create longer segments at the cost of character shape degradation and vice versa. Figure 4.13 shows the polygon estimation of the contours of a handwritten word for different values of T . For our system, we have used a value of T equal to 2, chosen empirically on the validation set. We then extract a set of features from these line segments.

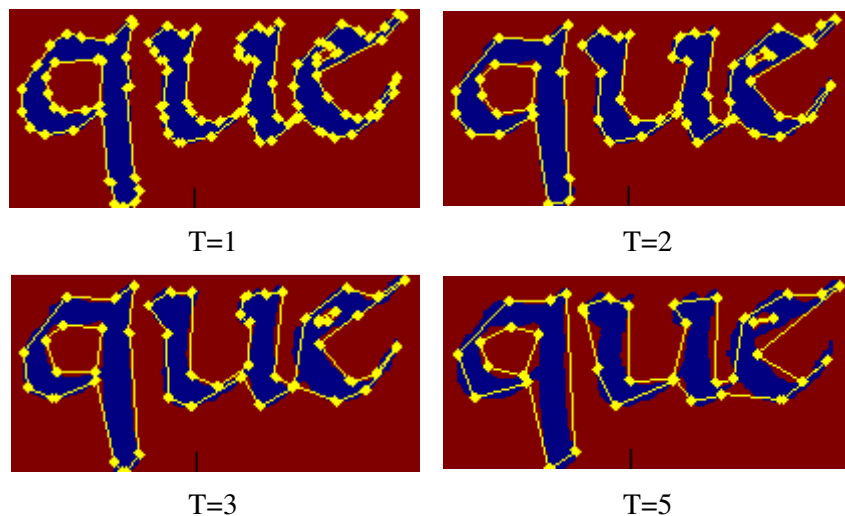


Figure 4.13 Polygonization at different values of T

a) Distribution of Slopes

From the polygonized contours of the handwritten image, we first compute the slope of each of the line segments and employ their distribution (*f₁₀*) for characterizing the writer. Each line is identified as belonging to one of the bins (classes) illustrated in Figure 4.14. These bins are chosen in such a way that the lines having nearly the same orientations as the principal directions (vertical, horizontal etc.) fall in their respective classes. For example, all the segments in the range -12° to 12° are classified as (nearly) horizontal and so on.

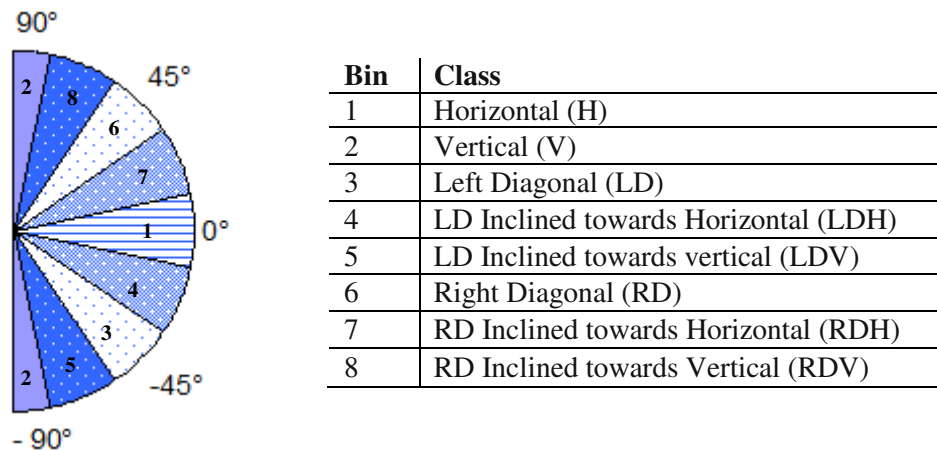


Figure 4.14 Division of Slopes (-90° to 90°) into bins and the corresponding segment classes

The writer specificity of the distribution of slopes can be seen from Figure 4.15 where the cumulative sum of the slopes' distribution is shown for two different samples from three writers. The contribution of each bin (segment class) for the two samples of the same writer is quite close as opposed to the samples from other writers, showing the effectiveness of *f₁₀* in characterizing the writer.

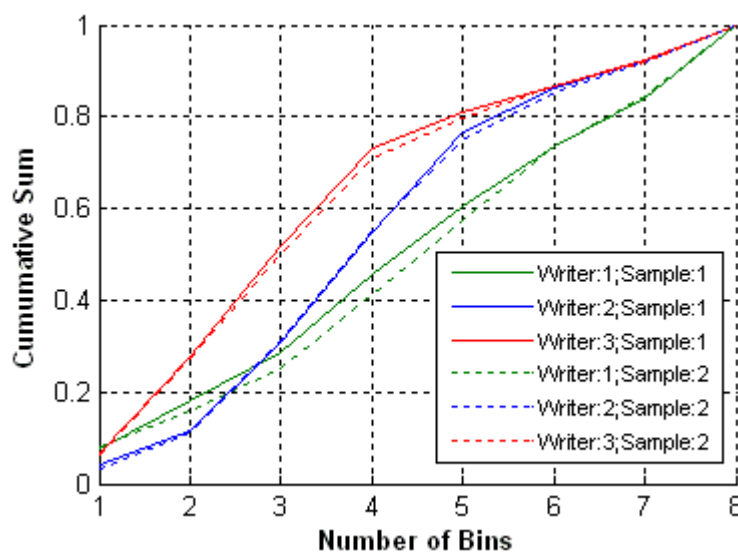


Figure 4.15 Cumulative sum of the slope distribution for three writers (two samples each)

Not only the number of slopes in a particular direction is important but their corresponding lengths as well, so in order to complement the distribution f_{10} , we also compute a length-weighted distribution of slopes (f_{11}), where for each segment at slope i , the bin i in f_{11} is incremented by the length of the segment. The distribution is finally normalized by the total length of segments in the image.

The distributions f_{10} and f_{11} can thus be viewed as the *number* and the *length* of segments belonging to the pre-defined segment classes respectively.

b) Distribution of Curvatures

A histogram based estimate of curvature, calculated from the chain code representation of the contours has been presented in section 4.1.1. From the polygonized version of writing, we compute the angle measurement between two connected straight segments as:

$$\alpha_i = \pi - \cos^{-1} \frac{V_i \cdot V_{i+1}}{|V_i| |V_{i+1}|} \quad (4.4)$$

With V_i and V_{i+1} being the vectors from (x_{i-1}, y_{i-1}) to (x_i, y_i) and from (x_i, y_i) to (x_{i+1}, y_{i+1}) respectively as illustrated in Figure 4.16.

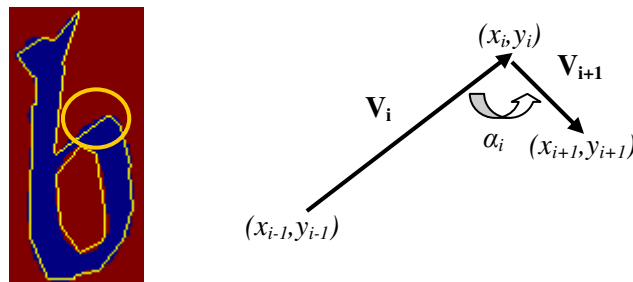


Figure 4.16 Curvature (Angle) between two connected segments

The distribution of these angles is then employed as our next feature (f_{12}). The angle bins ($0^\circ - 180^\circ$) are partitioned in a similar fashion as for the slopes. Similarly, in order to take into account the lengths of the segments forming a particular angle, a length-weighted version of f_{12} , f_{13} is also computed where each bin of f_{13} is incremented by the sum of lengths of the two segments forming the respective angle. The distribution is then normalized to have a sum equal to 1.

c) Distribution of Segment Lengths

Finally, irrespective of the orientation, it would be interesting to analyze the distribution of the lengths of segments in a writing. Generally, smooth strokes will lead to longer and fewer segments while shaky strokes will result in many small segments, thus the straight segment lengths could be useful in distinguishing the writings of different authors. We therefore use the distribution of these lengths (f_{14}) as a writer specific feature. An important issue here is how to determine the number

and partitioning of the bins in *f14* as the lengths are not normalized. This is done by studying the variation of lengths on the validation data set and setting an upper limit on the allowed segment lengths max_L , above which the segments are considered to be spurious and are discarded. In our case, this value was set to 100 pixels.

Summarizing, we extract a set of fourteen (normalized) distributions to represent a document image. The distributions for which the number of bins is not discussed explicitly have been partitioned empirically on the validation set. Table 4-2 summarizes the proposed features with the dimensionalities of each.

Table 4-2 Features and their dimensionalities

Feature	Description	Dimension
<i>f1</i>	Distribution of chain codes	8
<i>f2</i>	Distribution of 1 st order differential chain codes	7
<i>f3</i>	Distribution of 2 nd order differential chain codes	8
<i>f4</i>	Distribution of chain code pairs	44
<i>f5</i>	Distribution of chain code triplets	236
<i>f6</i>	Distribution of curvature indices	11
<i>f7</i>	Local stroke direction distribution	80
<i>f8</i>	<i>f2</i> computed locally	70
<i>f9</i>	<i>f3</i> computed locally	80
<i>f10</i>	Distribution of segment slopes	8
<i>f11</i>	Length-weighted distribution of segments slopes	8
<i>f12</i>	Distribution of curvatures	8
<i>f13</i>	Length-weighted distribution of curvatures	8
<i>f14</i>	Distribution of segments lengths	10
<i>Total:</i>		586

4.2 Writer Recognition

Once the handwriting samples have been represented by their respective features, we need to compute the distances between respective features to define a (dis)similarity between two handwriting samples. We tested a number of distance measures including: Minkowski (order 1 to 5), χ^2 , Bhattacharyya, (Non-) Intersection and Hamming distance as summarized in Table 4-3 (where p and q represent the two histograms to be compared, p_i and q_i are the entries in bin i of the histograms and dim represents the total number of bins in the histogram). In our series of experimentation χ^2 distance performed the best and the results that we will report in the subsequent sections would be based on χ^2 distance unless stated otherwise.

Table 4-3 Distance measures to compare two distributions

Minkowski Distance	$M^r(p, q) = \sum_{i=1}^{dim} (p_i - q_i ^r)^{1/r}$
χ^2 Distance	$\chi^2(p, q) = \sum_{i=1}^{dim} \frac{(p_i - q_i)^2}{p_i + q_i}$
Bhattacharyya Distance	$B(p, q) = \sqrt{1 - \sum_{i=1}^{dim} \sqrt{p_i q_i}}$
Hamming Distance	$H(p, q) = \sum_{i=1}^{dim} p_i - q_i $
(Non-) Intersection Distance	$I(p, q) = 1 - \sum_{i=1}^{dim} \min(p_i, q_i)$

4.2.1 Writer Identification

Writer Identification is performed by computing the distance between the query image Q and all the images in the training data set using a selected feature, the writer of Q being identified as the writer of the document that reports the minimum distance. This corresponds to the nearest neighbour classification (k -nn with $k=1$). For a query document, we not only find the nearest neighbour (Top-1) but a longer list up to a given rank (Top-K) thus increasing the chance of finding the correct writer in the retrieved list.

4.2.2 Writer Verification

For writer verification, we compute the distance between two given samples and consider them to be written by the same person if the distance falls within a predefined decision threshold. Beyond the threshold value, we consider the samples to be written by different writers. As discussed in section 3.7, varying the acceptance threshold the ROC curves are computed and the verification performance is quantified by the Equal Error Rate (EER), the point on the curve where the False Acceptance Rate (FAR) equals the False Rejection Rate (FRR). The lower the equal error rate value, the higher the accuracy of the system. The identification and verification results are presented in the following section.

4.3 Experimental Results

The experiments were mainly carried out on writing samples of 650 writers in the IAM data set presented in section 3.1. In addition to that, being a part of the campaign RIMES [Grosicki et al., 2008], we had the opportunity to access this data set, so we also evaluated our features on 375

writers from the RIMES database. One sample of each writer is contributed to the training set while the other to the test set. For writer identification we report the Top-1 and Top-10 identification rates while for writer verification we present the Equal-Error-Rate (EER). These results are presented first for the individual features and then for their various combinations.

4.3.1 Performance of Individual Features

We first present the performance of the individual features (shown in Table 4-4) that we discussed in the above sections. The results have been grouped according the feature type (global, local or polygon-based) while the numbers represent percentages. Although the performance of the features varies significantly, it can be noticed that, for a chosen feature, the performance is more or less consistent across the two data sets. The distribution of chain code triplets (*f5*), the local stroke distribution (*f7*) and the length-weighted distribution of segment slopes (*f11*) perform the best among their respective feature groups, with *f5* achieving the overall maximum identification rate on IAM (**79%**) and *f7* on RIMES (**78%**). Another interesting observation is that the local variants (*f7, f8 & f9*) of the chain code based features outperform their global counter parts (*f1, f2 & f3*). For writer verification, we achieve equal error rates of as low as **3.86%** and **6.70%** for the two data sets respectively.

Table 4-4 Performance of individual features

Data Set		IAM 650 Writers			RIMES 375 Writers		
Feature Class	Feature	Top1	Top10	EER	Top1	Top10	EER
<i>Global</i>	<i>f1</i>	36	74	7.23	48	77	10.32
	<i>f2</i>	34	76	6.89	50	74	11.40
	<i>f3</i>	42	81	6.56	52	76	11.47
	<i>f4</i>	67	88	5.67	68	85	8.05
	<i>f5</i>	79	93	4.64	75	91	6.70
	<i>f6</i>	43	77	6.96	55	77	10.87
<i>Local</i>	<i>f7</i>	77	93	3.86	78	92	7.02
	<i>f8</i>	46	83	7.10	58	82	10.25
	<i>f9</i>	42	79	7.95	55	79	11.39
<i>Polygon Based</i>	<i>f10</i>	55	86	5.82	64	86	8.21
	<i>f11</i>	58	87	5.42	65	88	7.98
	<i>f12</i>	37	75	6.97	52	76	11.12
	<i>f13</i>	40	78	6.56	52	79	10.29
	<i>f14</i>	31	72	7.51	51	75	12.04

4- Writer Characterization: Contour Based Features

Query Image

ily intruding members of Earl Russell's nu
 ming Committee of 100, the Committee clai
 day. It said pressure was being put on
 bers and associates all over the counry

Writer ID:029 Image:a02-053.png

Images Retrieved

er, to do the honours as host, in wich capac held a reception tonight in Accra's Ambassa POLICE, on direct orders from the Cabinet,	now there was more trouble bra those unbearable Dallases. Dall was detained in Paris by se
Rank:1 Writer ID: 029 Image:a02-053.png Dist:0.057707	Rank:2 Writer ID: 354 Image:g07-065.png Dist:0.18695
fresh in both their minds to be a comforta sect for discussion. "H's a command," he s can do with the extra money. It'll just a Jazzy's school fees." "What shall we do	on they observe his visible behaviour and then give theoretical interpretations of what lies behind behaviour. This is no more queer than the u
Rank:3 Writer ID: 582 Image:n04-015.png Dist:0.18706	Rank:4 Writer ID: 256 Image:i01-143.png Dist:0.21503
much as B for a lace collar. But this n't mean he was prepared to do as m his lady, for he records testily: 'My w	icular contraction. During cutting or punctur entricles, there is often forcible expul ool from the wound. Arterial diastole
Rank:5 Writer ID: 210 Image:e01-014.png Dist:0.23981	Rank:6 Writer ID: 305 Image:g02-065.png Dist:0.25179

(a) A successful search with the true writer found at rank 1

Query Image

she favour the abolition of the mo
 ds, but while it remains Labour has
 an adequate number of members. The
 val African Nationalist Parties of Na

Writer ID:003 Image:a01-007x.png

Images Retrieved

- was we probably facing in their zeal to get their orders and to out. Just what these orders will be, m . Now, because of prior hypnosis we la	be that as Labour OM B's opposed the Gov Bill which brought life peers into existence, not now put forward nominees. He be
Rank:1 Writer ID: 513 Image:m01-104.png Dist:0.094674	Rank:2 Writer ID: 003 Image:a01-003x.png Dist:0.097519
when he found cock-fighting going on in third made a positive approach. He set out to in use of reverence. The Prayer Book was to be re ce was the office of a bishop. Altars should	was a lady who, like her Uncle C the highest pride in keeping her word. ld see Zorander St'ing, " he went on i
Rank:3 Writer ID: 201 Image:d05-040.png Dist:0.18948	Rank:4 Writer ID: 570 Image:n02-157.png Dist:0.22704
d hardly believe it and blinked several ti don't I get a drink tonight?," she asked urse--- anything you like," he murmure	owth over two half-lives (ten days). Therefor a further forty days, the bismuth daught be within 0.1 sources are required for use
Rank:5 Writer ID: 480 Image:i04-005.png Dist:0.235	Rank:6 Writer ID: 400 Image:j04-008.png Dist:0.23657

(b) True writer found at rank 2 in the retrieved list of writers

Figure 4.17 Writer identification search on the IAM data set

The writer identification rates on individual features (as a function of hit list size) and the corresponding ROC curves have been presented in Appendix C.

4.3.2 Performance of Feature Combinations

After having evaluated the performance of individual features, we combine them by computing the distance between two writings as an average of the distances between the individual features. But before we could add up the distances together, we first need to carry out a normalization. Although we normalize each of the features to the range $[0, 1]$, the individual distances d_i can be of quite different dynamic ranges and combining these distances, the distance with larger magnitude might dominate the others. For example, Figure 4.18 illustrates the distances d_1 and d_2 of a query image to 100 images in the reference base. Evidently, adding the two will cause d_1 to overshadow d_2 , we therefore need to normalize each of the distances before proceeding to their combination.

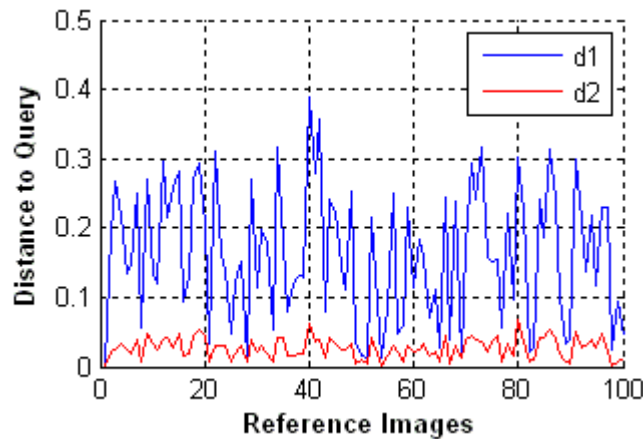


Figure 4.18 Distances (d_1 & d_2) to a query image

The simplest normalization technique would consist in finding the maximum and minimum values in d_i and normalize the sequence to the range $[0,1]$. This normalization, however, suffers from the drawback that an outlier (e.g. an abnormally high value) can take away most of the $[0,1]$ range, leaving a very narrow range for the rest of the values. We thus employ the Gaussian normalization, with the assumption that the values in d_i are distributed normally.

In the training set of N writing samples, we compute the individual distances (d_i) between each pair of images I_m and I_n . For N images, there are $N \times (N-1)/2$ possible values for each of the distances. Treating each d_i as a data sequence we find its mean μ_{d_i} and standard deviation σ_{d_i} . These values are computed offline and are based on the assumption that N is large enough so that the calculated values are good estimates of the true mean and standard deviation of the distances (d_i) between images. These values are then used in the normalization procedure.

When a questioned document Q is presented to the system we first compute the raw distances, between Q and the documents in the database, according to each of the features. These distances are then normalized as follows:

$$d'_i(D_j, Q) = \frac{d_i(D_j, Q) - \mu_{di}}{3\sigma_{di}} \quad (4.5)$$

$$j = 1, 2, \dots, N$$

This normalization produces 99% of all the distances in the range [-1,1] which is finally shifted to [0,1] as:

$$d''_i(D_j, Q) = \frac{d'_i(D_j, Q) + 1}{2} \quad (4.6)$$

The distances having a value greater than 1 are of course very dissimilar to the query and can be discarded without affecting the results.

Once the distances have been normalized, we proceed to feature combination which is performed by averaging the distances of the features participating in the combination. Among the various combinations tested, we will report a subset of results only that corresponds to the natural combinations: combining the individual features within each class of features, combining two feature classes and, combining all the features, summarized in Table 4-5. The identification rates as function of hit list size have been illustrated in Figure 4.19 while the corresponding ROC curves in Figure 4.20.

Table 4-5 Performance of feature combinations

Data Set	IAM 650 Writers			RIMES 375 Writers		
	Top1	Top10	EER	Top1	Top10	EER
<i>Global</i>	81	93	4.08	77	92	6.18
<i>Local</i>	81	95	3.76	77	89	7.85
<i>Polygon Based</i>	83	97	2.77	81	93	5.16
<i>Global & Local</i>	83	96	3.81	80	91	6.65
<i>Global & Polygon Based</i>	85	96	3.32	82	92	5.92
<i>Local & Polygon Based</i>	87	97	3.03	83	93	5.11
<i>All Features</i>	89	97	2.46	85	93	4.87

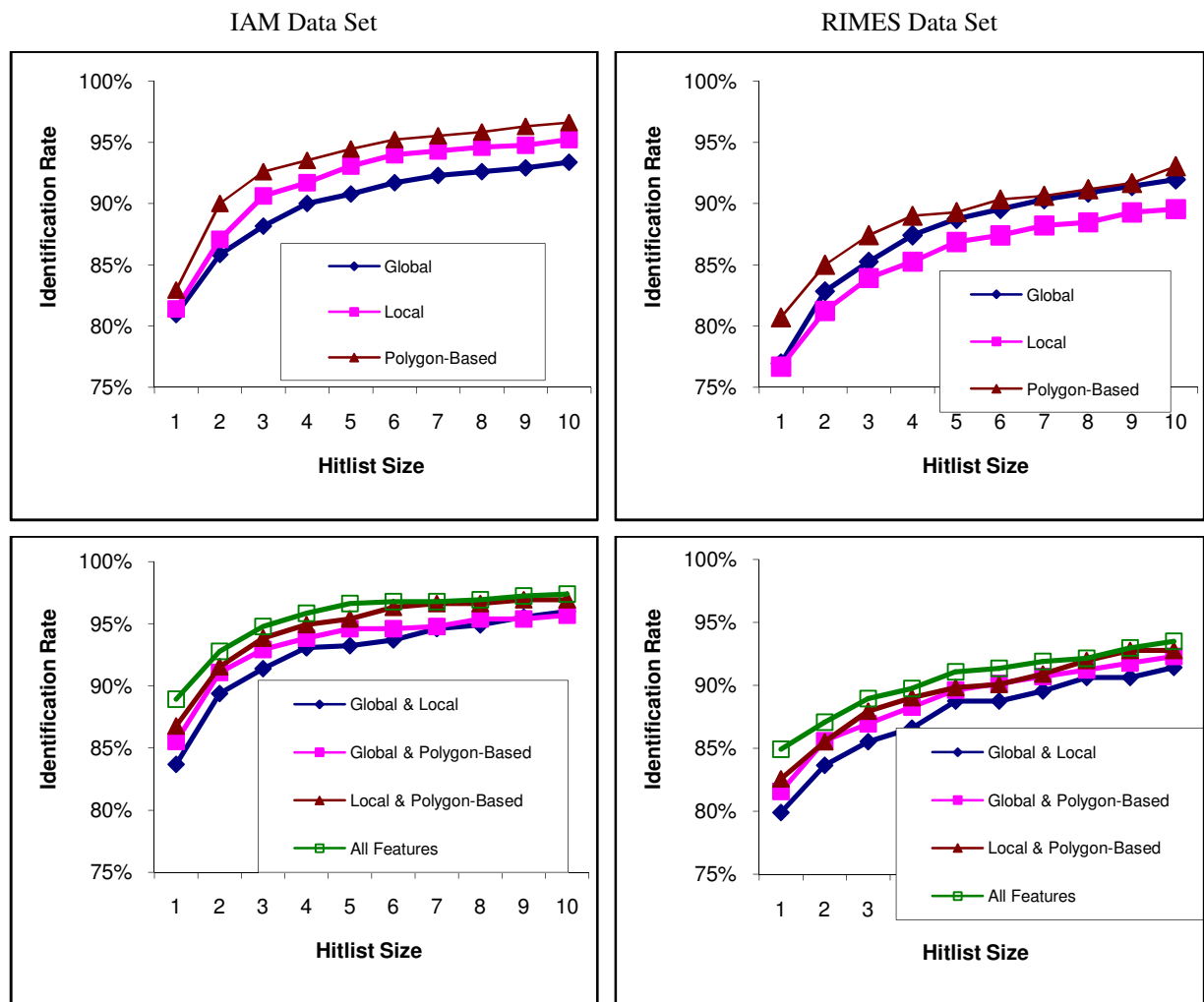


Figure 4.19 Performance of feature combinations on Writer Identification

As with the individual features, the performance of combined features is more or less consistent across the two data sets with the exception of local features. Contrary to IAM dataset, the global features outperform their local counterparts on the RIMES images (Figure 4.19). Although the Top1 identification rate on the RIMES writings for global and local features is the same (77%), the overall (Top1 – Top10) performance of global features is better than the local features. This is linked to the fact that the local features are calculated within small observation windows of fixed size therefore the wider variety in the writing size of the samples in the RIMES dataset results in a decreased performance.

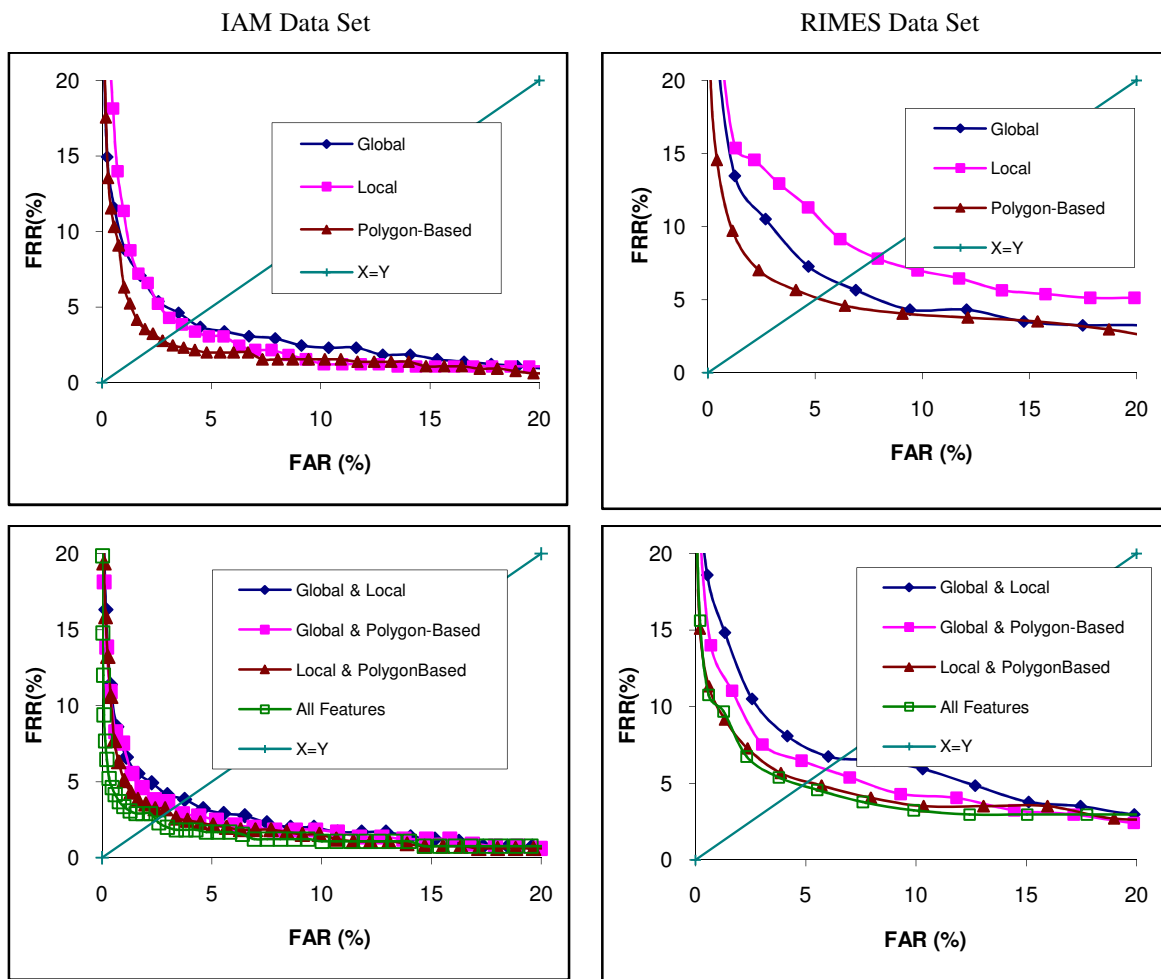


Figure 4.20 ROC Curves for feature combinations

The results that we presented are based on simple distance averaging where each feature contributes equally to the sum. We also evaluated a weighted combination of distances, the weights being chosen with respect to the performance of individual features. This however resulted only in marginal improvements in the over all performance of the system. Regarding the features that share some redundancy with others, we will discuss them towards the end of this chapter.

Keeping only the two best performing features f_5 and f_7 , we achieved an identification rate of approximately 83%. Figure 4.21 illustrates the effect of their weighted combination where the final distance between two writings is computed as: $(1 - \alpha)d_5 + \alpha d_7$ and it can be observed that the results exhibit only slight improvement in performance as compared to simple distance averaging ($\alpha = 0.5$).

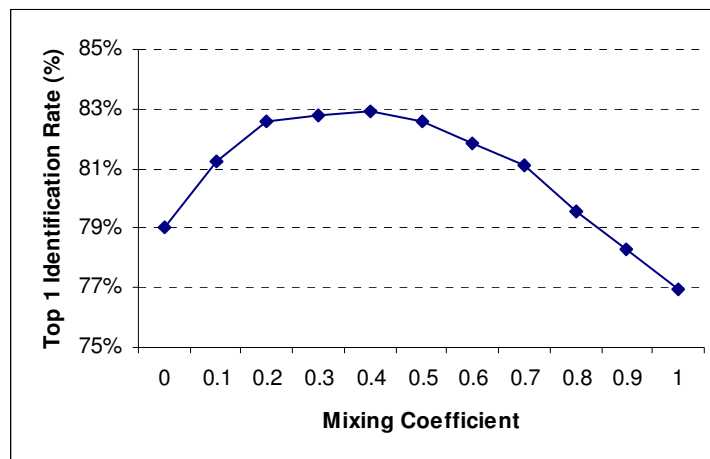


Figure 4.21 Writer identification performance for a weighted combination of $f5$ and $f7$

We also performed the same experimental evaluations by merging the two datasets into one large set of (650+375) 1025 writers. This combined set represents, in terms of number of writers, the largest data set used for text independent writer recognition until present. We will present the results of combined features only, summarized in Table 4-6 where we have achieved an overall identification rate of 86% and EER of as low as 3.30%.

Table 4-6 Performance of combined features on the combined dataset (1025 writers)

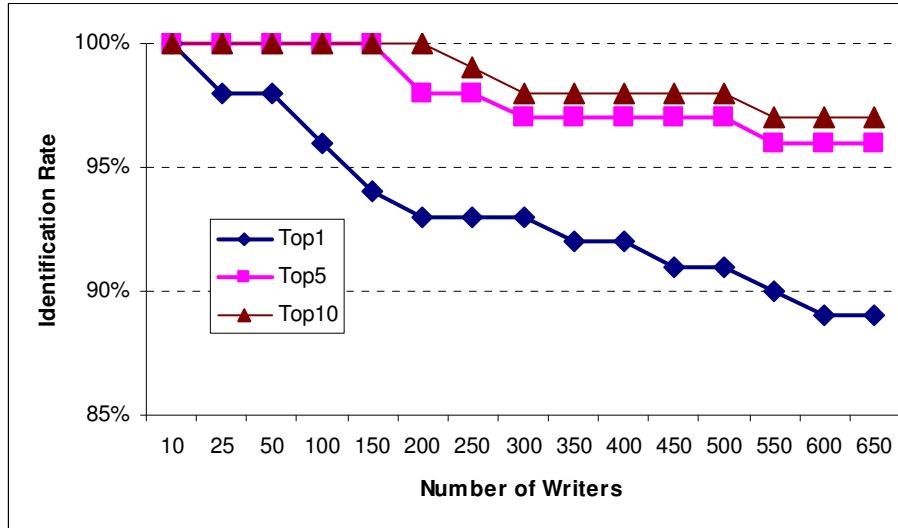
Data Set	IAM+RIMES 1025 Writers		
	Top1	Top10	EER
<i>Global</i>	78	92	4.39
<i>Local</i>	79	92	4.84
<i>Polygon Based</i>	80	93	3.92
<i>Global & Local</i>	82	93	4.34
<i>Global & Polygon Based</i>	83	94	3.87
<i>Local & Polygon Based</i>	84	95	3.76
<i>All Features</i>	86	95	3.30

Two important parameters that influence the performance of the system are the number of writers and the amount of text available for each writer. It would therefore be interesting to analyze how these parameters affect the results as presented in the following.

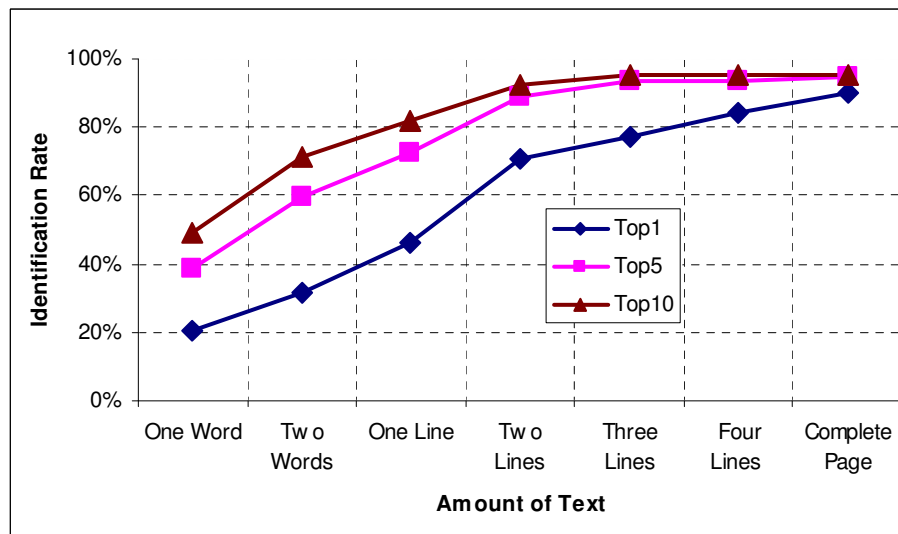
4.4 Stability of Features

We first present the influence of number of writers on the identification rate. These evaluations have been carried out on the IAM dataset by varying the number of writers from 10 to 650. It can be seen from Figure 4.22 (a) that there is a consistent but not dramatic decrease in the identification

rate as the number of writer increases starting with 100% for 10 writers and dropping to 89% for 650 writers. Naturally, the Top-5 and Top-10 performances are much more stable dropping to 96% and 97% respectively when the entire dataset is used.



(a) Identification rate as function of number of writers on the 650 writers of IAM dataset



(b) Identification rate as function of amount of text on 300 writers of IAM dataset

Figure 4.22 Writer Identification performance as function of (a) Number of writers (b) Amount of text

The influence of amount of available text for each writer is studied on the 300 writers (having contributed at least two samples) of the IAM base. Similar to section 3.9.3, the amount of text (both for training and testing) is varied from one word to complete page the identification rates being presented in Figure 4.22 (b). For a single word an identification rate of about 20% is achieved which is increased to 91% when the complete page (5 to 10 lines on the average) is used.

Comparing these results with the ones presented for the code book based features (Figure 3.21) indicates that contour-based features are relatively more stable when the amount of text is varied.

After having studied the contour based features, their combinations and stability, we would now be interested to see if we could go a step further and improve the writer recognition performance by combining these features with the information on redundant writing patterns of a writer.

4.5 Combining Contour-based Features with Redundant Writing Patterns

It would be interesting if we could combine the idea of redundant patterns in a writing presented in the previous chapter with the contour based features. We exploited this redundancy by generating a writer-specific as well as a universal code book, the questioned writing being compared to these code books, the later reporting better results on writer identification and verification. Since the frequent writing shapes of a writer are represented in the common code book space as a distribution (which will be termed as *f15* from here onwards), it could well be combined with the contour based distributions capturing the orientation and curvature information in a writing. We will report the performance of combined features on 650 writers of the IAM dataset (summarized in Table 4-7). Combining the contour based features and the code book distribution, the over all identification rate rises to 91% while the equal error rate drops to 2.23%.

Table 4-7 Performance of combining contour based features and the code book distribution

Data Set	IAM 650 Writers		
	Top1	Top10	EER
<i>f15 (Code book)</i>	84	96	4.49
<i>f1 – f14</i>	89	97	2.46
<i>f1 – f15</i>	91	97	2.23

As in section 3.9, we will present a comparative overview of the results of recent studies on writer identification task on the IAM data set. The comparison that we presented earlier was meant to give an idea of the performance of code book based methods only while the one in Table 4-8 provides a general comparison where the overall identification rates reported in different studies have been summarized. Although an identification rate of as high as 98% has been realized, it has been achieved on 100 writers with sufficient text from each of the writers (4 samples per writer in training and 1 in testing). Until present, [Bulacu & Schomaker, 2007] is regarded as achieving the best results with an identification rate of 89% on the entire IAM dataset. With the proposed features, we are also able to correctly identify the writer of a questioned document 89% of the times, on the same dataset and by using the same evaluation criterion as in [Bulacu & Schomaker,

2007] (leave-one-out approach on the entire dataset without distinguishing the training and test sets).

Table 4-8 Performance comparison of writer identification systems on IAM dataset

	Writers	Samples/writer	Performance
[Marti et al., 2001]	20	5	90.7%
[Bensefia et al., 2005b]	150	2	86%
[Schlapbach & Bunke, 2006a]	100	5/4	98.46%
[Bulacu & Schomaker, 2007]	650	2	89%*
Our method	650	2	91% / 89%*

As we discussed earlier, there exists a partial redundancy among certain features as well. This is one of the reasons why, for example, combining the chain code based global features did not result in a significant performance gain as compared to the results on individual features ($f5$ alone achieved an identification rate of 79% while combining the features $f1$ to $f6$ resulted in 81%). We therefore carry out a study in an attempt to discover, which subset, among the features $f1 - f15$, represents the relevant features for our problem of writer recognition. In a similar fashion, within each of the features, certain components (bins) might be more relevant than others. The selection of these features (as well as the components within a feature) is discussed in following section where we will very briefly introduce feature selection, and then demonstrate its application on our feature set.

4.6 Feature Selection

Feature selection is the process selecting relevant and informative features with the motivation of data/feature set reduction, performance improvement and data understanding [Guyon et al., 2006]. Our primary objective in carrying out a feature selection process would be to identify, among the proposed feature set, the features (or feature components) that are relevant in characterizing the writer of a handwritten text. Feature selection algorithms are mainly classified into two broad categories: filter and wrapper methods. *Filters* carry out feature selection independent of any learning algorithm and the features are selected as a pre-processing step. *Wrappers* on the other hand use the performance of a learning machine as a black box to score feature subsets. In addition to filters and wrappers, there are also *Embedded* methods which perform variable selection in the process of training and are usually specific to given learning machines [Guyon & Elisseeff, 2003]. Traditionally a feature selection method involves four key steps as illustrated in Figure 4.23.

* $k-m$ using a leave-one-out approach

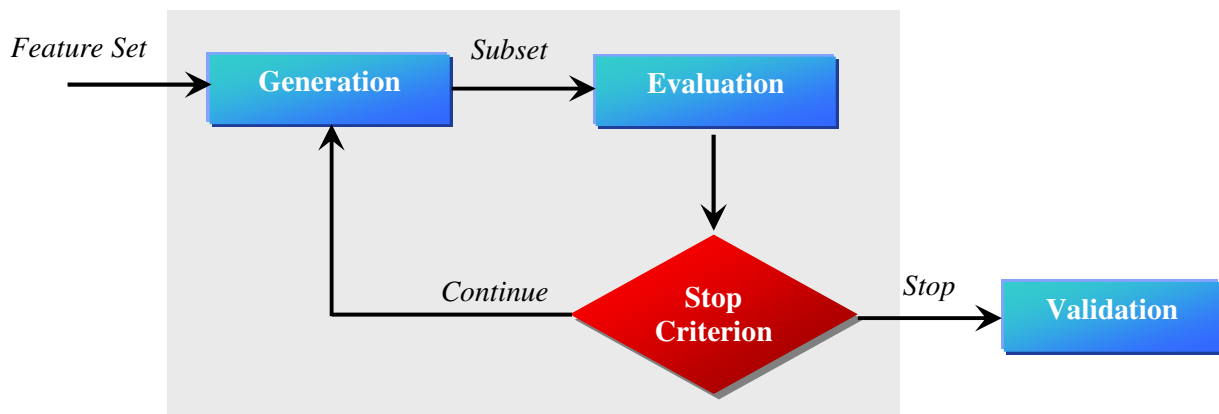


Figure 4.23 Key steps in Feature Selection

The generation process implements a search method that produces subset of features. It may start with an empty set (features are added successively: forward selection), with all the features (features are removed successively: backward elimination), at both ends (adding and removing features simultaneously: bidirectional) or a randomly selected subset. The goodness of generated subset is then evaluated by an evaluation criterion which may be independent (filter) or dependent (wrapper). A stopping criterion is tested every iteration to determine when the FS process should stop. Typical criteria involve achievement of optimal subset or bounds on number of features or iterations etc. The resulting subset of features can be validated once the stopping criterion has been satisfied.

For our problem, we have used a genetic algorithm (GA) to implement feature subset selection (generation step), the evaluation is carried out using a wrapper method while the stop criterion is based on the number of generations.

4.6.1 Genetic Algorithms

GAs belong to a group of methods, called evolutionary algorithms, that have been applied to feature selection with different degrees of success [Siedlecki & Sklansky, 1989]. In general, a GA begins with an initial set of random solutions called *population*. These solutions (individuals) are represented using a fixed-length binary string coding where each bit (gene) represents the elimination (0) or inclusion (1) of the respective feature. The individuals are evaluated using a *fitness function* and are assigned a fitness value. A *selection function* then decides which individuals to pick from the current population to create the next generation. The population is then updated using the *genetic operators* (crossover and mutation); an iteration of these operators being a generation. The process is repeated until a termination criterion (e.g. number of generations) is satisfied.

Genetic algorithms have the advantage of quickly scanning a large solution space however they suffer from drawbacks like finding the sub-optimal solution and over fitting. A number of solutions

have been proposed in the literature to improve the selection performance of genetic algorithms [Dos Santos et al., 2009] [Chen & Chen, 1997]. We have however limited our study to the application of basic genetic algorithms only.

Fitness function, that evaluates the goodness of each individual in a population, plays the most important role in genetic search. The simplest fitness function for our system could be the writer identification rate. For a query document Q we compute its distance to all the samples in the training set, the samples are then ordered in a sorted hit list with increasing distance to the questioned document. The identification rate is the percentage of finding the correct writer at rank 1 in the hit list. The objective of our system, however, is not only to get a good top 1 performance but also to improve the rank of the correct writer in the hit list, as high as possible. We thus propose the following fitness function where the identification rates are weighted with respect to the hit list size, the performance on a hit list of size 1 being given the maximum weight.

$$Fitness = \frac{\sum_{i=1}^N (N - i + 1) \times R_i}{N(N + 1)/2} \quad (4.7)$$

Where R_i is the writer identification rate on a hit list of size i , N is the total number of writers and the denominator represents the normalization factor.

4.6.2 Analysis of Feature Relevance

In order to analyze the usefulness of the features we divide them into three categories: indispensable, partially relevant, and irrelevant features. This division corresponds to the one proposed in [Pervouchine & Leedham, 2007] where the authors aim to study the usefulness of some of the features used by forensic document experts. A feature is attributed to one of these categories based on how often it is selected among several executions of the selection procedure. We have used the same definitions for the three categories as follows:

- Indispensable: Feature selected in each selected feature subset.
- Irrelevant: Feature not selected in any of the selected subsets.
- Partially Relevant: Feature selected in some of the subsets.

The feature selection in our case may operate at two different levels. We may either opt to select the features $f1$ to $f15$ or apply the selection mechanism on the individual components of each feature. We will analyze both of them one by one and then combine the two in a cascaded form as discussed in the following. It should be noted that we will be using the terms feature and feature selection for both levels (features and feature components).

4.6.2.1 Selection of Features

We will first analyze the relevance of the features $f1 - f15$. The GA is used to generate individuals of length 15 and the set bits are used to select the respective features. The feature selection mechanism is executed on the first 100 writers of the RIMES data set and the selected subset is evaluated on the 650 writers of the IAM data set. The GA is executed ten times with the following parameters (chosen experimentally):

- Population Size: 50,
- Crossover Rate: 0.6,
- Mutation Rate: 0.02,
- Selection Rule: Roulette wheel selection,
- Number of Generations: 50.

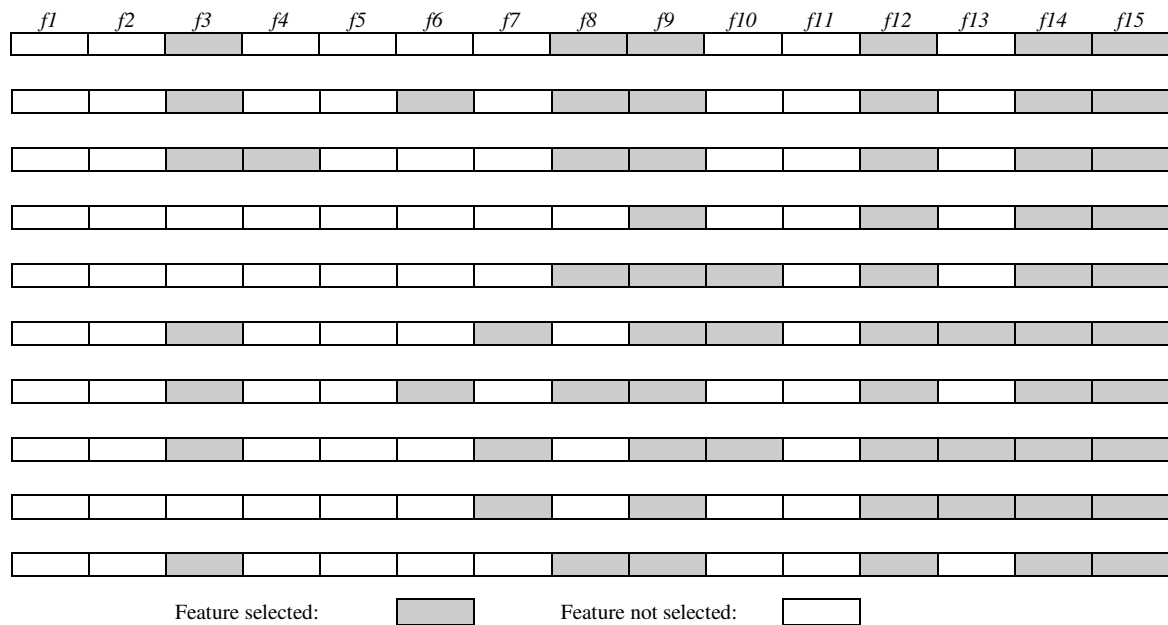


Figure 4.24 Features selected in ten runs of the GA

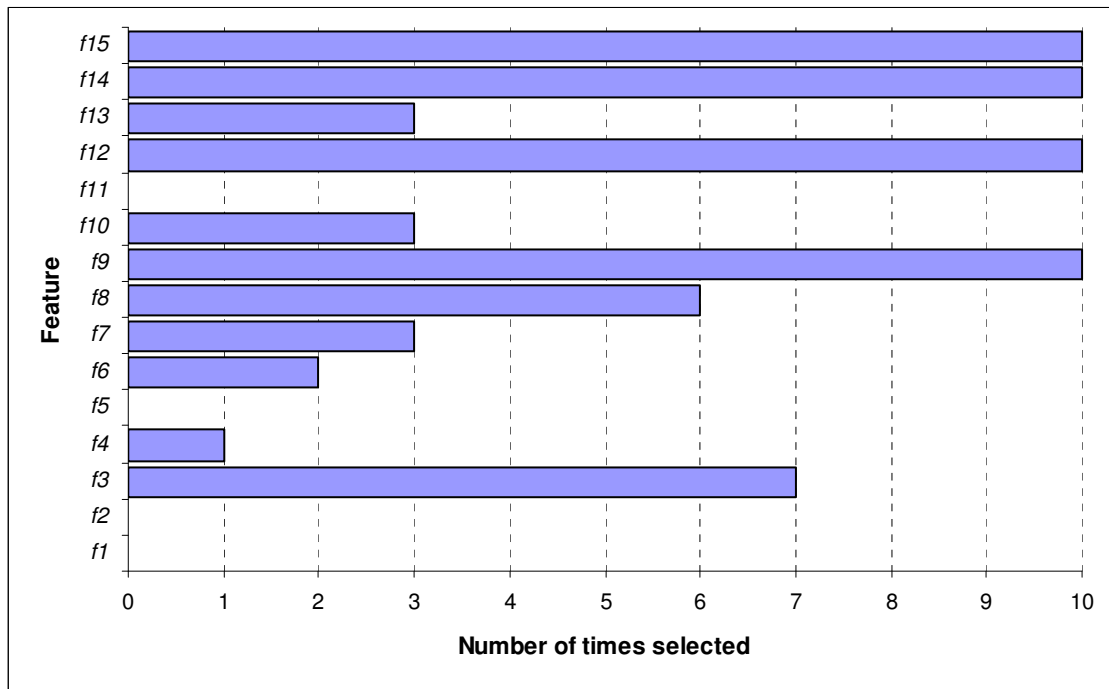


Figure 4.25 Frequencies of selected features

Table 4-9 Division of features according to relevance

Indispensable	Partially Relevant	Irrelevant
<i>f9,f12,f14,f15</i>	<i>f3,f4,f6,f7,f8,f10,f13</i>	<i>f1,f2,f5,f11</i>

Table 4-10 Performance of selected feature subsets on 650 writers of the IAM data set

Run	Features	Top 1	Top 10
1	<i>f3,f8,f9,f12,f14,f15</i>	90.61	97.54
2	<i>f3,f6,f8,f9,f12,f14,f15</i>	90.31	97.54
3	<i>f3,f4,f8,f9,f12,f14,f15</i>	90.61	97.54
4	<i>f9,f12,f14,f15</i>	91.08	98
5	<i>f8,f9,f10,f12,f14,f15</i>	89.69	97.54
6	<i>f3,f7,f9,f10,f12,f13,f14,f15</i>	91.08	97.69
7	<i>f3,f6,f8,f9,f12,f14,f15</i>	90.31	97.54
8	<i>f3,f7,f9,f10,f12,f13,f14,f15</i>	91.08	97.69
9	<i>f7,f9,f12,f13,f14,f15</i>	91.38	98
10	<i>f3,f8,f9,f12,f14,f15</i>	90.61	97.54
Average	-	90.68	97.66

The initial population is randomly generated. For each generation of the GA, each chromosome is evaluated using the fitness function, the fitness values of the current population being used to find the off springs of the next generation. The generational process ends when the termination criterion is satisfied which in our case is the number of generations. The selected features correspond to the best individual in the last generation. Figure 4.24 gives an overview of the features selected in the 10 runs of the GA while Figure 4.25 indicates the corresponding selection frequencies of these features. In our case, features f_9 , f_{12} , f_{14} and f_{15} are selected in each of the 10 runs and thus are considered indispensable for our problem whereas the features f_1 , f_2 , f_5 and f_{11} that are not selected in any of the runs are regarded as irrelevant, i.e. they do not result in improving the performance once used with other features in our feature set. The division of features according to their relevance has been summarized in Table 4-9 while the writer identification performance of selected feature subsets on the 650 writers of the IAM data set has been shown in

Table 4-10. An average identification rate of 90.68% (Top10: 97.66%) is realized by employing 6.5 features on the average. It is important to note that the features which are regarded as ‘irrelevant’ during the feature selection mechanism should not be considered as useless. They should be viewed as not contributing to performance enhancement in the presence of other features (in the combination). All features however might not be possible to compute in all situations. For example, the code book based feature (f_{15}) might not be available if the method is applied to a text written in a different script than the one for which we have an available code book. So in such cases, a feature that was regarded as irrelevant might play a useful role.

We will next study the feature selection mechanism on the individual components within each of the features as discussed in the following.

4.6.2.2 Selection of Feature Components

In our proposed feature set, the contour based features add up to a total of 586 dimensions (Table 4-2) while adding the distribution of frequent writing patterns (f_{15}) gives a dimensionality of 686. Thus in order to select a subset among these 686 feature components (bins), the GA thus generates individuals of length 686 and the set bits are used to select the respective bins. As earlier, the selection algorithm is executed on the RIMES data set and the selected subset is evaluated on the IAM data set with the following parameters for the GA:

- Population Size: 300,
- Crossover Rate: 0.6,
- Mutation Rate: 0.02,
- Selection Rule: Roulette wheel selection,
- Number of Generations: 50

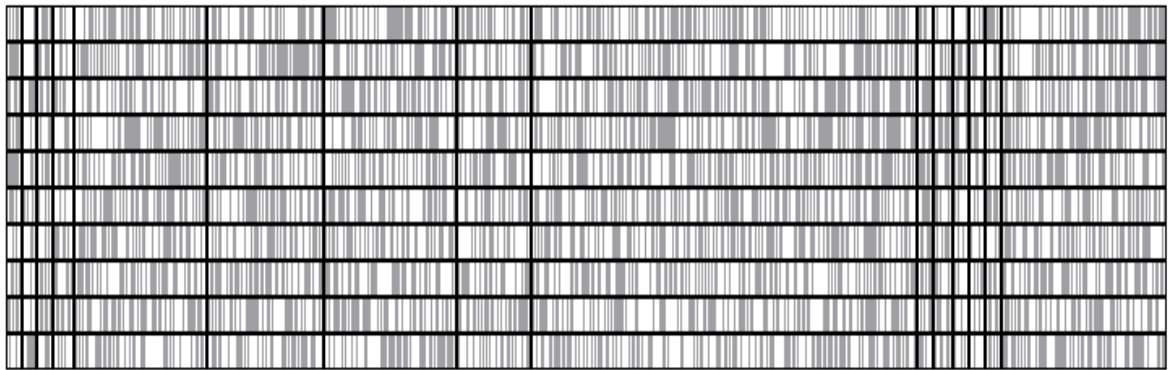


Figure 4.26 Feature components selected in 10 runs of the GA

Figure 4.26 indicates the feature components selected in the 10 runs of the genetic algorithm while Figure 4.27 shows the average (of 10 runs) number of components selected from each of the features $f1 - f15$. In general, the dimensionality of each of the features reduces to one half on the average. Since the total number of possible feature (component) subsets is very high (2^{686}) as opposed to applying the selection mechanism directly on the features $f1 - f15$ (2^{15}), it is quite rare to find the feature components that are selected in each (or none) of the ten optimal subsets. It is therefore difficult to classify the selected feature components into groups (indispensable, relevant and irrelevant) as we did for the features. After ten executions, an average identification rate of 89.82% (Top10: 97.65%) is achieved with approximately 50% overall reduction in the feature dimensions, summarized in Table 4-11.

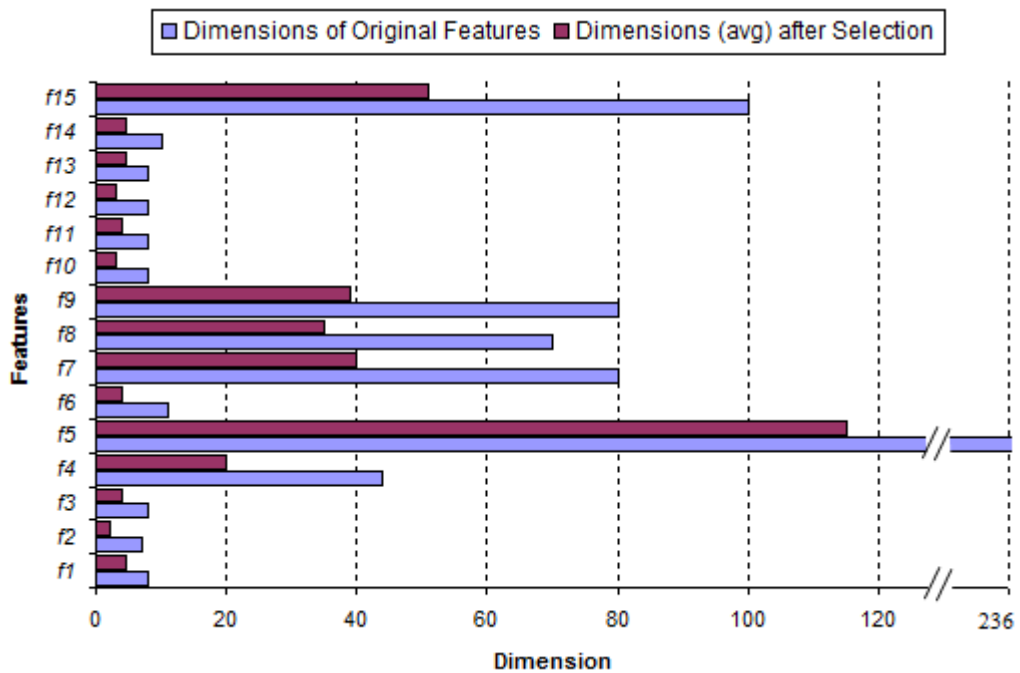


Figure 4.27 Average number of components selected for each of the features

We also applied the GA iteratively, each iteration beginning with the feature set selected in the previous iteration. The performance however begins to deteriorate so we have limited our discussion to the results of first iteration only.

Table 4-11 Performance of selected feature components on 650 writers of IAM data set

Run	Dimension	Top 1	Top 10
1	324	90.46	96.77
2	335	89.85	97.54
3	350	89.38	97.69
4	349	88.62	97.69
5	351	90.15	97.69
6	331	89.23	98.15
7	318	89.54	97.54
8	328	89.85	97.69
9	328	90.46	98
10	317	90.62	97.69
<i>Average</i>	333	89.82	97.65

After having applied the feature selection at the feature and component levels, it would now be interesting to combine the two by re-defining our feature set where each feature comprises only the frequently selected components as presented in the following.

4.6.2.3 Selection of Features (Reduced Dimensions)

Analyzing the selection frequencies of the feature components in the ten runs of the genetic algorithm (Figure 4.28), we define the feature set $f1' - f15'$ where f_i' comprises the components of f_i that are selected in at least 50% of the times. This results in reducing the overall dimension of our feature set from 686 to 399 as summarized in Table 4-12. We then execute the genetic algorithm using the same parameters as we used for the selection of features $f1 - f15$ (section 4.6.2.1).

The selection frequencies of the features $f1' - f15'$ have been shown in Figure 4.29 while the actual features selected in each of the runs as well as the respective identification rates have been summarized in Appendix D. Using 7.5 features on the average (average dimension: 315), we achieved an overall identification rate of 90.06% (Top10: 97.90%) as compared to 90.68% (Top10: 97.66%) when using an average of 6.5 features (average dimension: 281) from the feature set $f1 - f15$.

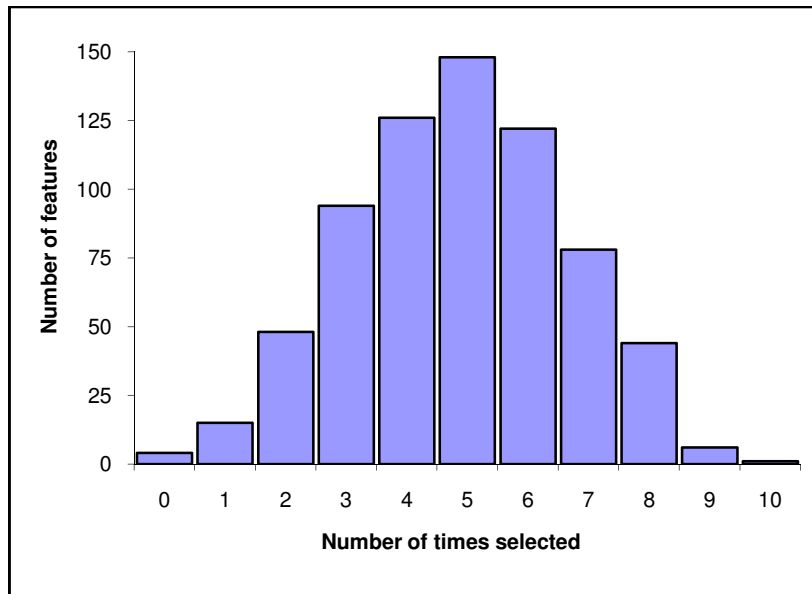


Figure 4.28 Feature selection distribution

Table 4-12 Features with reduced dimensions

Dimension	Features	<i>f1</i>	<i>f2</i>	<i>f3</i>	<i>f4</i>	<i>f5</i>	<i>f6</i>	<i>f7</i>	<i>f8</i>	<i>f9</i>	<i>f10</i>	<i>f11</i>	<i>f12</i>	<i>f13</i>	<i>f14</i>	<i>f15</i>	<i>Total</i>
	Original	8	7	8	44	236	11	80	70	80	8	8	8	8	10	100	686
	Reduced	4	2	5	22	140	3	51	40	47	1	5	2	6	4	67	399

Comparing the division of features (according to their relevance) across the two selection scenarios (Table 4-9 vs. Table 4-13) it can be noticed that *f9* and *f15* come out to be the common indispensable features in the two cases. On the contrary, the features *f14* and *f5* are assigned to the groups indispensable and irrelevant respectively whereas *f14'* and *f5'* are assigned to these groups in the reverse order (*f14'*: irrelevant and *f5'*: indispensable). Another interesting observation is that the selected feature subsets are more consistent on the set *f1' – f15'* than on the initial set *f1 – f15*. However, considering the overall identification rates and the dimensionality of the feature subsets, we can say that applying feature selection on the actual feature set *f1 – f15* is more useful than on the redefined set *f1' – f15'*.

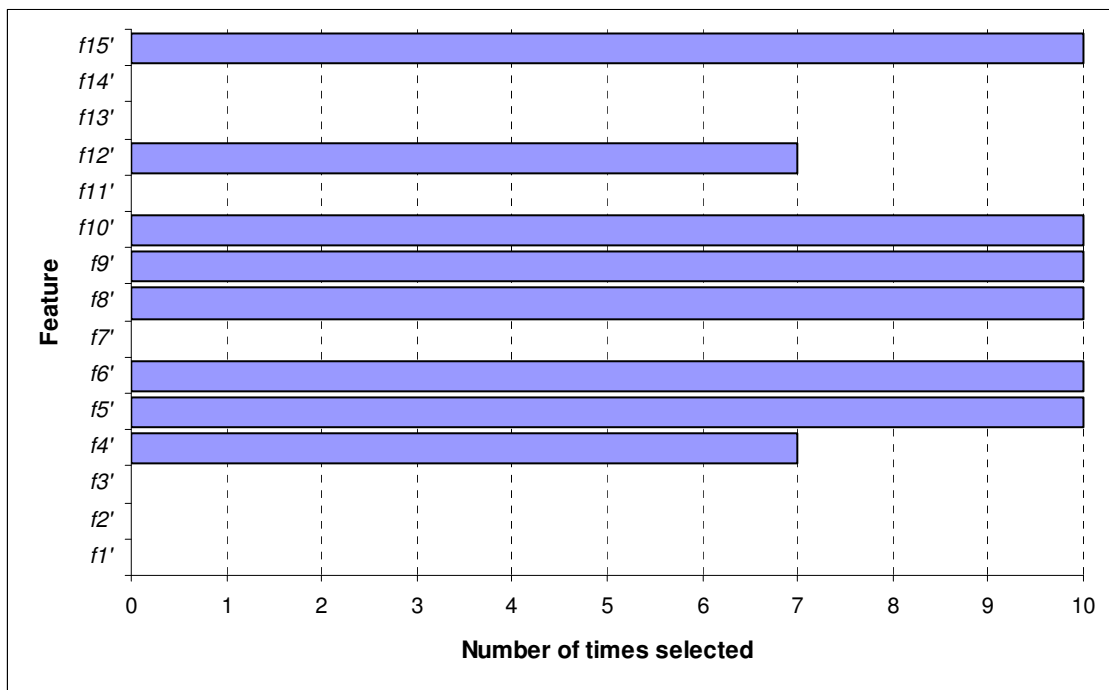


Figure 4.29 Frequencies of selected features (reduced dimension)

Table 4-13 Division of features (reduced dimension) according to relevance

Indispensable	Partially Relevant	Irrelevant
<i>f5', f6', f8', f9', f10', f15'</i>	<i>f4', f12'</i>	<i>f1', f2', f3', f7', f11', f13', f14'</i>

4.7 Conclusion

We presented in this chapter a set of features that were mainly designed to capture the orientation and curvature information in a writing. We first discussed the usefulness of extracting these features from contours rather than the skeleton and then presented the two representations for the extracted contours: a sequence of chain codes and a set of polygons. The effectiveness of these features in characterizing the writer of a handwritten document was demonstrated by evaluating their performance on writer identification and verification. We then combined the idea of redundant writing patterns with these features and the results realized were as good as the best results reported in the literature so far. A study on the relevance of the proposed features was also conducted using a feature selection mechanism. We also evaluated our system on the largest data set (in terms of number of writers) used so far in a study on text-independent writer recognition and the system was able to perform reasonably well without degrading the recognition rates too much.

Chapter 5

Applications

We have seen in the previous chapter how the proposed contour-based features are employed in characterizing the author of a document image. We then demonstrated the effectiveness of these features by applying them to identify and verify the writer of a questioned document. The proposed methodology is very generic which allows its application not only to writer recognition on non-Latin scripts but also enables to address a number of other problems like signature verification, classification and retrieval of ancient manuscripts and even recognition of characters.

In this chapter, we will present the application of our method to three of the potential applications namely the classification of medieval documents, writer recognition on Arabic script, and verification of signatures as presented in the following sections. We will limit our feature set to the contour-based features (introduced in the previous chapter) only, although the idea of redundant writing patterns can also be applied to each of these applications. In addition, we will not distinguish between the feature groups (indispensable, partially relevant and irrelevant) that we achieved as a result of the feature selection mechanism. Instead, for simplicity, we will show the application of the entire set of contour-based features, detailed in the following sections.

5.1 Classification of Ancient Manuscripts

Writing styles and forms are bound to evolve over time and the knowledge about these forms and the way they have evolved, enables the palaeographers identify the periods and the geographical location in which a manuscript was written. The quantity of these ancient manuscripts stored in archives, libraries and private collections is enormous and a system that could help the palaeographers in manuscripts dating, authentication and studying the links between writing styles and writers could be very useful. The analysis and classification of writing styles is similar in many respects to the recognition of writers, the latter requiring more precision in the decision to assign a script to a particular class.

Among the well known methods for handwriting classification [Crettez, 1995] proposes an analysis of the variability of handwritings with the objective of identifying the family of the handwriting style. The fractal analysis of a handwritten image reflects the writing style of its author and serves to classify writings according to their legibility [Boulétreau, 1997]. These methods have been validated on contemporary writings. Important contributions on medieval and humanistic manuscripts include [Aiolli et al., 1999] [Eglin et al., 2007] [Joutel et al., 2008] [Yosef et al., 2004] and [Moalla et al., 2006].

We will be interested in applying our proposed feature set for automatically classifying the bases of ancient manuscripts with particular focus on the digitized medieval handwritten texts. For that, we will be working on a sample of 310 medieval manuscripts selected from the collection of IRHT (Institut de Recherche en Histoire des Textes). In fact, this study has been carried out in the framework of the project ANR-GRAPHEM⁶, so from now onwards, we will use the term ‘*Graphem dataset*’ to refer these images.

Since we are dealing with ancient documents, we will first be discussing the extraction of text regions and binarization of extracted text before proceeding to feature extraction. We will next show how we generate a set of overlapping classes using these features and then employ the idea of relevance feed-back to improve the performance of document image retrieval when presented with a query. Finally, we will try to present our results in a quantified form.

5.1.1 Text Extraction

In addition to handwritten text, the documents that we consider contain plenty of drop caps, fancy borders and separators, drawings, images and marks of stamps etc (Figure 5.1). Since we characterize the document by writing and not by other objects (e.g. drop caps), we first need to extract text regions from the documents. The quality of the documents and the variety of non-text objects, however, does not allow an automatic segmentation of text areas. We therefore carry out a manual segmentation of text by cropping a part of image containing homogeneous text. All the subsequent steps are then applied to these segmented images as discussed in the following.



Figure 5.1 Examples of non-text regions in the document images

⁶ ANR GRAPHEM: ANR-07-MDCO-006-04.

5.1.2 Document Binarization

Our feature set is extracted from the contours of the writing and in order to extract the contours, the document images first need to be binarized. Contrary to the modern data sets that we considered earlier, the quality of historical manuscripts is generally quite poor as the documents degrade over time due to, for instance, storage conditions. Thus the foreground and background are difficult to separate and the classical thresholding methods fail when applied to these documents [Leedham et al., 2003]. We investigated a number of local thresholding methods where the threshold is computed for each pixel of the image as a function of its neighbourhood. The most classical example of such a method is the Niblack algorithm [Niblack, 1986] that calculates the threshold value for a pixel as a function of the mean and standard deviation of its neighbouring pixels. Over the years, a number of Niblack inspired have been proposed, in an attempt to improve the quality of binarization or to adapt to a certain type of documents. Notable contributions include [Sauvola et al., 1997] [Wolf & Jolion, 2003] [Feng & Tan, 2004]. Table 5-1 summarizes the calculation of threshold value in these methods.

Table 5-1 Summary of neighbourhood based thresholding methods

Method	Threshold	Parameters
Niblack	$T_{Niblack} = m + k \times s$	$m = \text{mean value of pixels in the window}; s = \text{standard deviation}; k = -0.2$
Sauvola	$T_{Sauvola} = m \times \left(1 - k \times \left(1 - \frac{s}{R}\right)\right)$	$k = 0.5; R = 128$
Wolf	$T_{Wolf} = (1 - k) \times m + k \times M + k \times \frac{s}{R} (m - M)$	$k = 0.5; M = \text{minimum grey-value in the image}; R = \text{maximum grey-value standard deviation over all the local neighbourhoods (windows)}$
Feng	$T_{Feng} = (1 - \alpha_1) \times m + \alpha_2 \times \left(\frac{s}{R_s}\right) \times (m - M) + \alpha_3 \times M$	$R_s = \text{dynamic range standard deviation calculated in a larger neighbourhood}; \alpha_2 = k_1 (s/R_s)^\gamma; \alpha_3 = k_2 (s/R_s)^\gamma; \gamma = 2; \alpha_1 = 0.1-0.2; k_1 = 0.15-0.25; k_2 = 0.01-0.05$

Since we could not carry out a quantified comparison of these methods, we evaluated them by visual inspection which revealed that none of the methods produced ‘*acceptably good*’ results on these images. We then evaluated the binarization scheme presented in [Bar-Yosef, 2005] which claims to work quite well on degraded images. First a global thresholding is carried out that enables separating the background from noise-free characters. A more sophisticated local method is then employed to binarize the noisy characters. Figure 5.2 shows an example of binarizing a noisy image while the algorithmic details of these steps could be found in [Bar-Yosef, 2005]. This method produced by far the (visually) best binarized images.



Figure 5.2 Binarization Steps

Once binarized, we extract the connected components in the image (using 8-connectivity) and for each of the components we find its contours and the two representations, the Freeman chain code and the set of polygons approximating the contours. We then proceed to the extraction of features.

5.1.3 Feature Extraction

Among the features presented in section 4.1 we have chosen to keep the global chain-code-based and the polygon-based feature sets as the wide variety in writing conditions, writing instrument and writing size etc. makes it difficult to apply the window-based local features on these images. Just to recall, the global feature set comprises the features $f1 - f6$ while the polygon-based feature-set includes $f10 - f14$.

5.1.4 Similarity based Incremental Grouping

We will now try and group similar writings into classes (clusters) which are then presented to the palaeographers for inspection and feedback. What makes the task difficult is that there are no ground truth classes so the ‘*goodness*’ of the produced classes is estimated by the feedback of the palaeographers. An image in the data set is selected as query q and its distance to all the $i \neq q$ images in the data set is computed using the features discussed earlier. The documents are then sorted with increasing distance to the query and the list is presented to the palaeographers who then identify the first ‘*false*’ image (f) in the retrieved list, i.e. the first image in the list which (according to their expert opinion) should not belong to the same writing class as that of the query document. All the samples in the list up to this wrongly retrieved image are then grouped into a

class and we start with f as the next query image. The process is repeated until all of the images have been attributed to one of the (potentially overlapping) classes. Evidently, the classes obtained are sensitive to the initial query so the entire procedure has to be carried out repeatedly each time starting with a different query image. So basically, it is more of an *Image Retrieval* than a classification. An example of a group of *similar* images retrieved on the Graphem dataset has been illustrated in Figure 5.3 where the first image represents the query document.

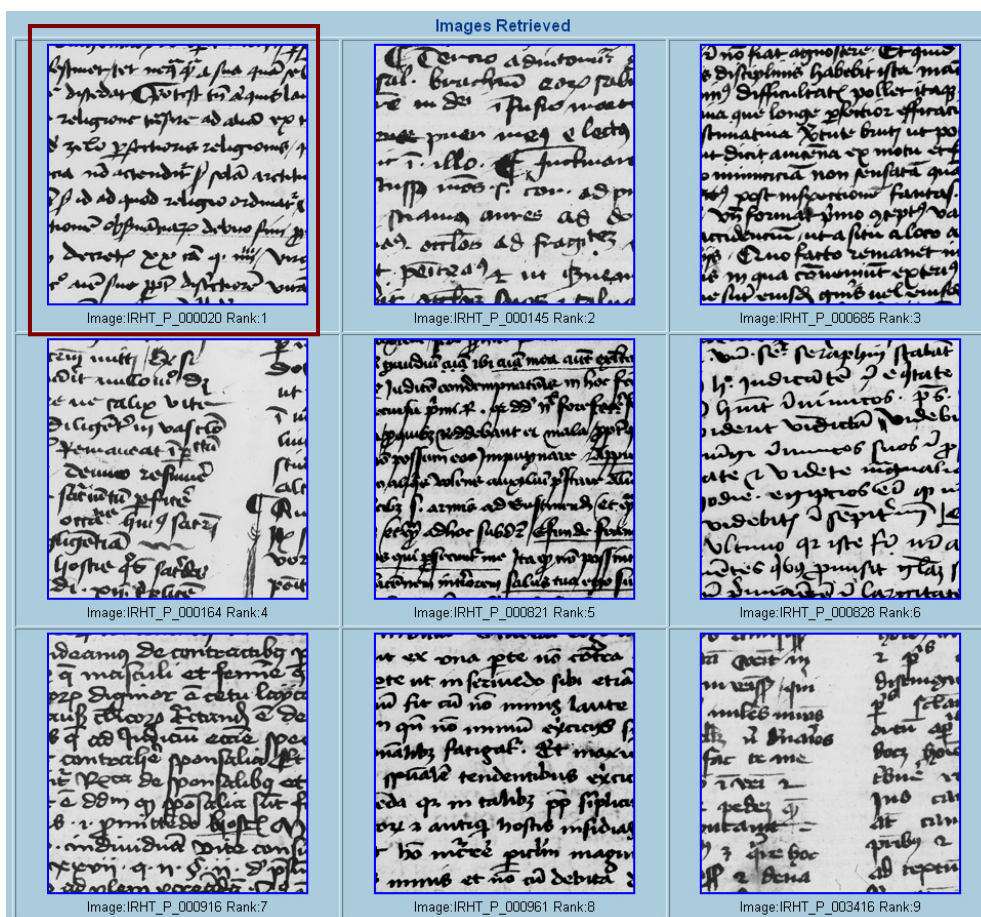


Figure 5.3 An example of images retrieved on the Graphem dataset

5.1.5 Relevance Feedback

Certain features might be of more interest for the palaeographers than the others and it might be desirable to assign more weight to a specific subset of features while calculating the similarity between two documents, according to the characteristics of the writing under study. This could be done by allowing the user (palaeographer in our case) to assign weights to the features [Niblack et al., 1993] [Bach et al., 1996]. This manual assignment of weights however, is not feasible solution as it imposes a big burden on the user, requiring the user to have a comprehensive understanding of the low level features used in the system, which generally is not the case [Lee & Street, 2002]. The most common solution to this problem is the relevance feedback method that

automatically adjusts the feature weights according to the preferences of the user. It is the process of automatically adjusting an existing query using information feed-back by the user about the relevance of previously retrieved documents [Rui et al., 1997]. Starting with a query, the initial results are presented to the user who marks them as relevant or irrelevant as per his requirements/perception. The weights associated with the features are then dynamically updated to better model the user's preferences. The working principle of a relevance feedback system has been illustrated in Figure 5.4

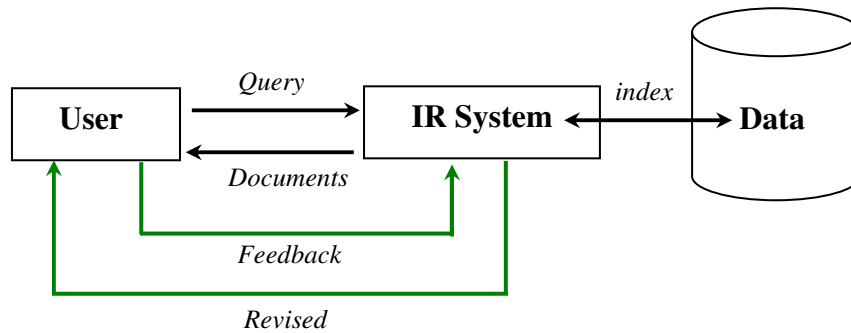


Figure 5.4 Principle of a relevance feedback system

We define the distance between a questioned document Q and a reference document D as:

$$D(Q, D) = \sum_{i=1}^{\text{No. of features}} w_i d_i(Q, D) \quad (5.1)$$

Where d_i represents the distance between the two documents computed for feature i and is given by:

$$d_i(Q, D) = \chi^2(f_i^Q, f_i^D) \quad (5.2)$$

The objective of the relevance feedback, in our particular case, is to update the weights w_i . We have chosen to use the vector space feedback model presented in [Rui et al., 1998] where the new query feature vector is generated as weighted linear combination of the original feature vector and the feature vectors of the images that were labelled as relevant or irrelevant by the user.

To start with, the weights w_i are initialized to w_0 , the non-bias weights. For a query document Q the N most similar documents are retrieved using equation 5.1 and presented to the user.

$$R = \{D_1, D_2, \dots, D_N\} \quad (5.3)$$

The user then assigns a relevance score S to each of the retrieved documents. [Rui et al., 1998] suggests the scores of +3, +1, 0, -1 and -3 to capture the semantic meaning of *highly relevant*, *relevant*, *no opinion*, *irrelevant* and *highly irrelevant* and we have kept the same scores for our system.

We then find the set of documents R^i similar to the query with respect to each of the individual features f_i .

$$R^i = \{D_1^i, D_2^i, \dots, D_N^i\} \quad (5.4)$$

The weight w_i for f_i is then calculated as follows:

$$w_i = \begin{cases} w_i + \text{Score}_j, & \text{if } R_j^i \text{ is in } R \\ w_i + 0, & \text{if } R_j^i \text{ is not in } R \end{cases} \quad (5.5)$$

$$j = 1, 2, \dots, N$$

This weight updating will ensure that larger the intersection of R^i and R , the greater is the weight assigned to f_i which is the very idea behind relevance feedback. Finally, the $w_i < 0$ are set to 0 and the weights are normalized ensuring that the total sum of the weights remains 1:

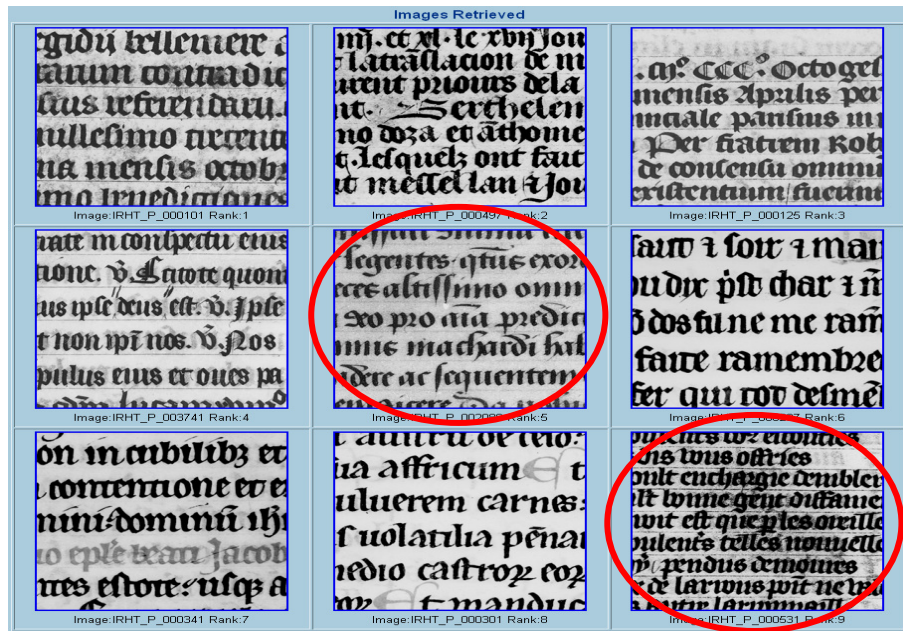
$$w_i \leftarrow \frac{w_i}{\sum_i w_i} \quad (5.6)$$

Figure 5.5 illustrates a retrieval session where given a query (the first image in the image set) the system first retrieves the most similar images with non-bias weights (Figure 5.5.a). The user (palaeographer) then remarks that two of the retrieved images (at ranks 5 and 9) do not belong to the same writing class as that of query and thus marks them as irrelevant. The system then recalculates the weights w_i incorporating user's preferences and as it can be seen from Figure 5.5.b, one of the irrelevant images has been eliminated from the list, the other has moved from rank 5 to rank 9 and a new relevant image has appeared as well (at rank 8). The user may perform multiple feedback iterations to improve the retrieval results.

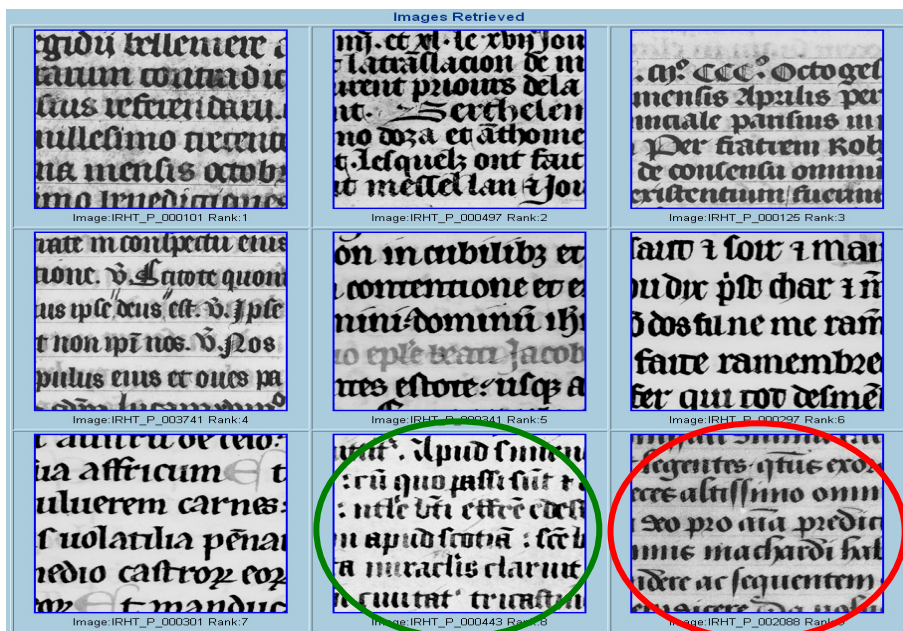
The retrieval and feed back methodology discussed above is appropriate in the framework of the project Graphem, however, in order to estimate the goodness of our feature set, we need to have some quantified results. For this purpose, we propose to distinguish these images into training and test sets and perform a classification task. We split the documents into two equal parts, the first half contributing to the training while the other to the evaluation set. The features are extracted from the training images and a *k-means* clustering algorithm is applied with two writings being compared using the χ^2 distance. Since the palaeographers currently search for a 'good' value of k , we have made k vary from 2 to 30. Applying a Principal Component Analysis (PCA) and reducing the writing representation to two dimensions, the writing classes obtained have been visualized (for $k=3$ & 5) in Figure 5.6.

The classification is then carried out by picking up each image in the test set and assigning it to one of the k classes. An image is said to be correctly classified if it is attributed to the same class as that of its counter part in the training set, i.e. the two halves (training & test) of an image should belong to the same class to be considered as correct classification. The classification is carried out

using the k -nearest neighbours (k -nn) with $k=3,5,7$. It should also be noted that a k -means with $k=1$ (i.e.; no clustering of the training set) and then a k -nn with $k=1$ changes the problem of classification to writer identification.



(a) Initial retrieval results



(b) Retrieval results after relevance feedback

Figure 5.5 Retrieval results with and without feedback

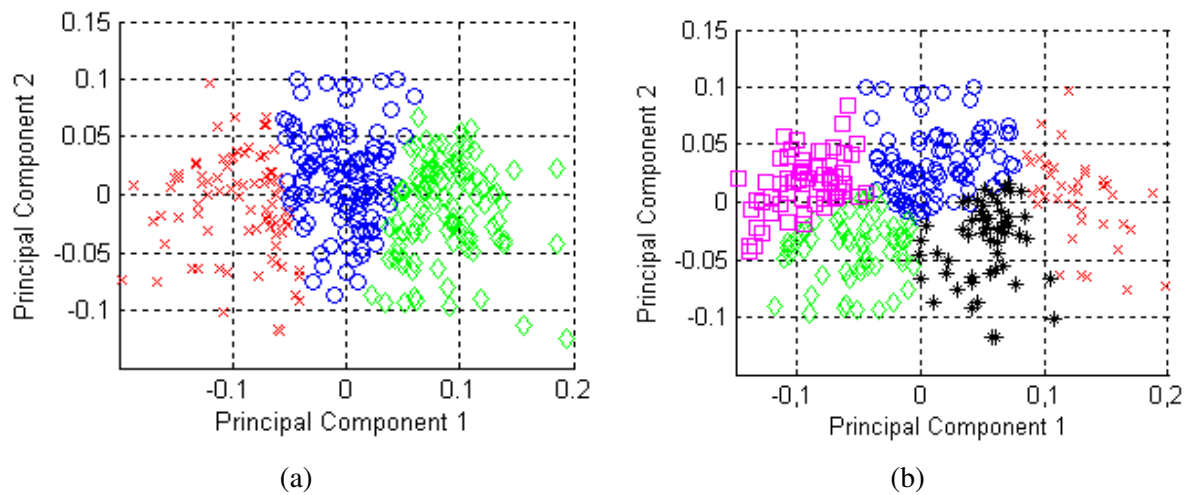


Figure 5.6 Writing classes obtained with k-means for (a) $k=3$ (b) $k=5$

Figure 5.7 illustrates the classification performance as a function of number of classes. Naturally the performance drops as the number of classes increases. For up to 10 classes, classification rates in excess of 80% are realized (k -nn with $k=3$). It should however be noted that the clusters of writings that we obtain by employing the proposed features have yet to be compared with the ones suggested by the palaeographers. The evaluation procedure that we followed (dividing the images into two disjoint sets) was meant to give a rough idea about the effectiveness of the system in a quantified form. Since the results are very promising, we expect that the suggested features would be of great help for the palaeographers.

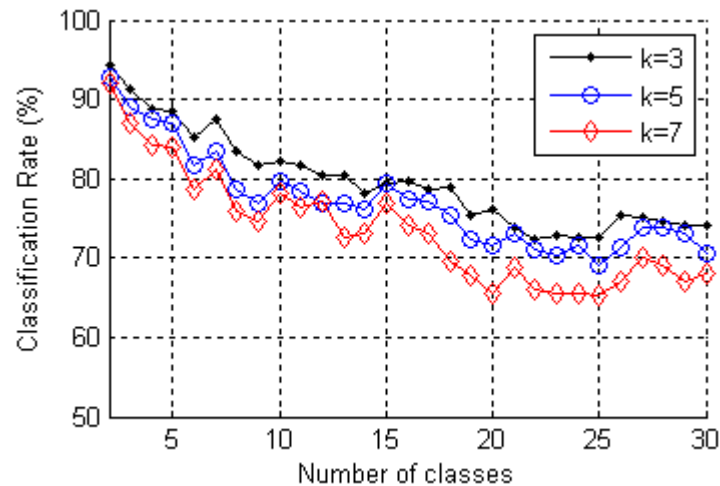


Figure 5.7 Classification performance (knn) on the Graphem dataset with $k=3,5$ and 7

5.2 Writer Recognition on Arabic Documents

After having applied the proposed features on ancient manuscripts, we will now see how stable their performance is once applied to a non-Latin script. For this purpose, we will perform the writer recognition task on Arabic handwritten documents from the IFN/ENIT database. The IFN/ENIT database comprises forms with handwritten Arabic town/village names (more than 26,000 words) collected from 411 different writers. This dataset was originally developed for the training and testing of Arabic handwriting recognition systems and was also used in the ICDAR 2005, 2007 and 2009 Arabic OCR competitions [Margner et al., 2005] [Margner & Abed, 2007] [Margner & Abed, 2009]. However, since each image also contains the identity of its writer, this data set can be employed for writer identification as well. The original dataset is divided into four disjoint sets and we will show our results on set ‘A’ only, containing word samples of 100 writers. Each writer contributed on the average 50 to 60 words. As always, we divide the available set of words for each writer into two roughly equal parts, one used in training while the other in testing.



Figure 5.8 Division of a word into sub-images and its polygon approximation

We will report the writer recognition performance only for the feature combinations in each feature type and the overall combination of all the features. As with Latin scripts, the combination of polygon-based features performs the best with an identification rate of 85% and an equal error rate of 3.22%. The combination of all the features produces an overall identification rate of 92% and an equal error rate of as low as 2.94%, indicating the generality of the features. The results have been summarized in Table 5-2 and illustrated in Figure 5.9.

Table 5-2 Writer recognition performance on the IFN/ENIT database

Data Set	IFN/ENIT 100 Writers		
	Top1	Top10	EER
<i>Global</i>	84	97	3.95
<i>Local</i>	82	98	4.97
<i>Polygon Based</i>	85	100	3.22
<i>All Features</i>	92	100	2.94

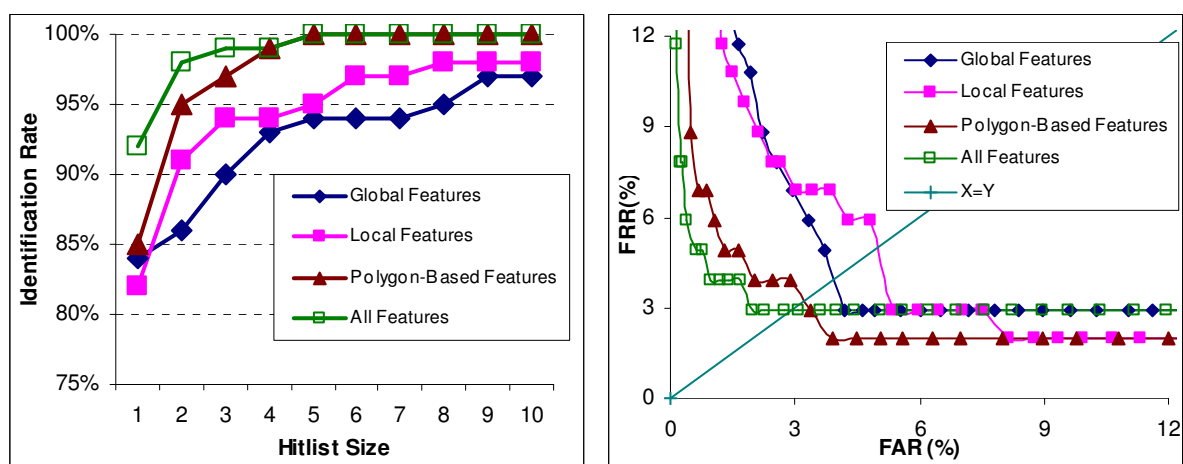


Figure 5.9 Identification rates and ROC curves on the IFN/ENIT dataset

5.3 Signature Verification

We will now present the application of our method not on the texts but on the signatures of different writers in an attempt to perform signature verification. Of course our objective here is not to propose a solution to the signature verification problem but to see how the proposed features would work on signatures. In comparison to writer recognition, signature verification enjoys the advantage of comparing same text (signature) for an individual. The disadvantage on the other hand is that a very limited amount of text per writer is available making it a challenging task. We have worked on a collection of signatures from 48 different individuals (students) with 15 signatures of each used for training whereas 15 for testing. We considered only the simple case where there are no forgeries and the data set comprises genuine signatures only. Figure 5.10 shows samples of three of the signatures used in our study.

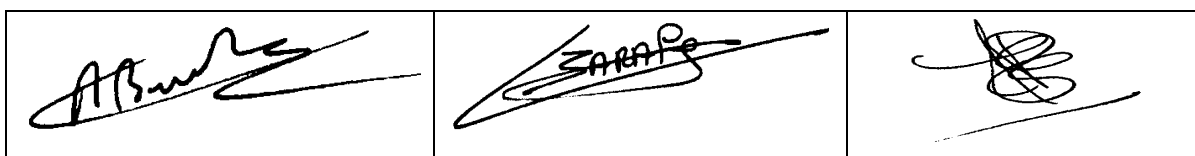


Figure 5.10 Examples of signatures

Before the extraction of features the signatures are first normalized. The normalization includes a rotation according to the inertial axis [Wirotius, 2005] (as indicated in Figure 5.11) and then a scaling that ensures that all the signatures are contained in a rectangle of fixed width (fixed to 300 pixels) preserving the proportions of the signature.

Once normalized, for each of the signatures we compute the contour-based features and represent an individual by the mean vector of the features computed from his/her 15 samples. As an example, Figure 5.12 illustrates the chain code distributions computed from 10 signatures of an individual along with the mean distribution. These mean values are computed for each of the

features for all the 48 individuals. Each individual can thus be viewed as a class comprising 15 elements (signatures) and that is represented by its mean value.

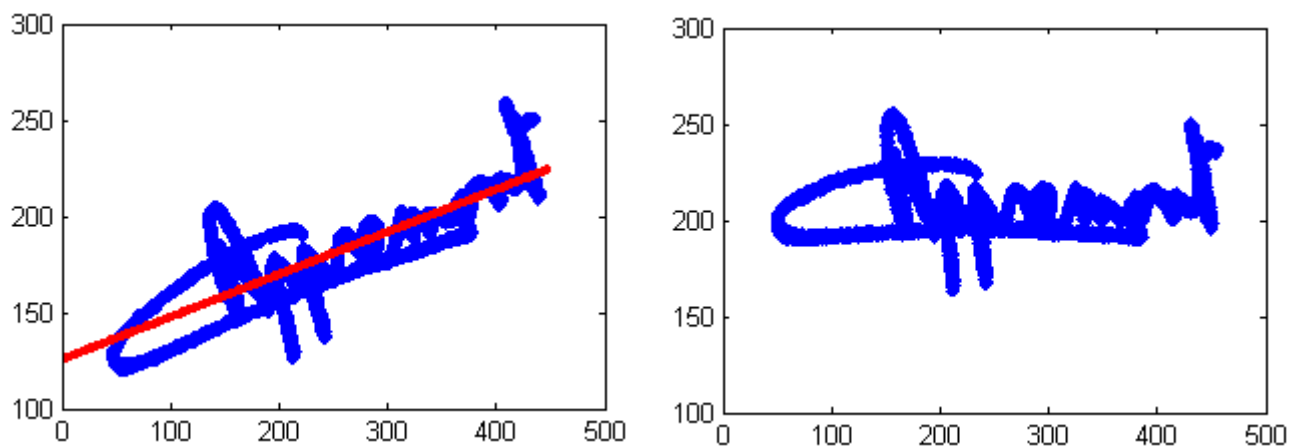


Figure 5.11 Inertial axis of the signature and the redressed signature

For evaluation we compute the distance of each signature in the test set ($15 \times 48 = 720$ signatures) to each of the 48 classes (individuals). 91% of the times, the signatures are assigned to the correct individual in the training set. The performance on signature verification is quantified, like in case of writer verification, by the equal error rate which comes out to be 5.88% on the tested set of signatures and is illustrated in Figure 5.13. The error of course is relatively higher in comparison to the state of the art methods in signature verification. But as we mentioned earlier, the objective of this study was to analyze the goodness of the features designed for writer recognition (with relatively larger amount of text per writer available) on signature verification. Considering this fact, the performance can be regarded as good enough.

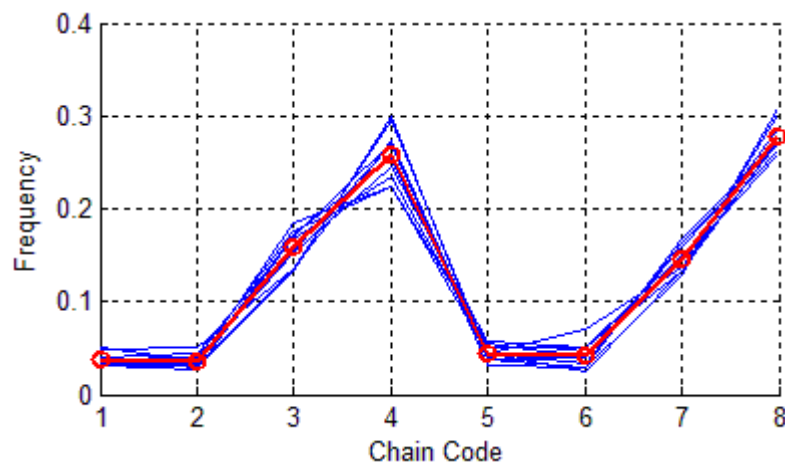


Figure 5.12 Distribution of chain codes for 10 signatures of an individual

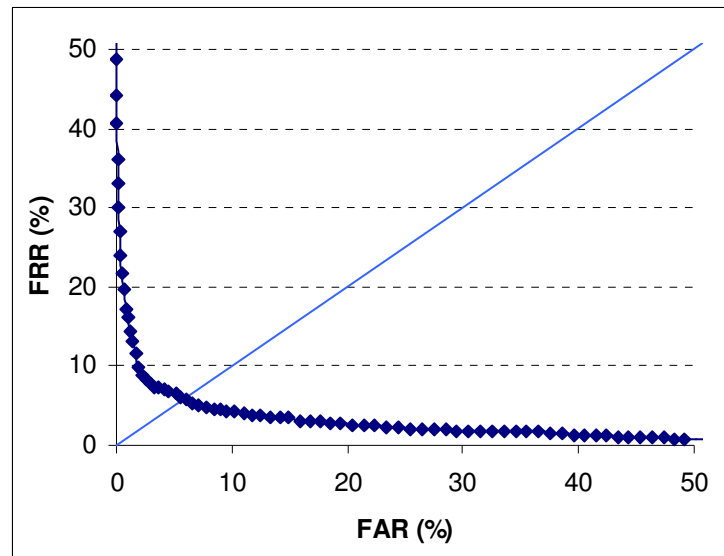


Figure 5.13 ROC curve on the signatures

5.4 Conclusion

We presented in this chapter the application of our method on the classification of ancient manuscripts, recognition of writer from Arabic handwritten documents and finally the verification of signatures. We mainly focused on the contour-based features that are simple and efficient to compute, nevertheless, the idea of frequent writing patterns could well be applied to these applications as well. As we mentioned earlier, the objective of this chapter was to study the effectiveness and generality of the proposed features and the encouraging results achieved on each of the considered applications validate that our proposed features are not only generic but quite effective as well.

Chapter 6

Conclusion and Perspectives

The objective of our research was to address the problem of automatic writer identification and verification from offline scanned images of handwriting, a problem that enjoys a renewed interest of the community due to its applications in forensic document analysis, indexing and retrieval of document bases and recognition of handwriting. This objective was met by developing an original method that exploits two different facets of handwriting: the existence of certain redundant patterns in writing and the visual attributes of orientation and curvature characterizing the writer of a handwritten text.

Contrary to the classical approaches which analyze the redundancy of writing shapes at the grapheme level, we exploited it at a much smaller scale of observation that correspond to small writing fragments and allows to capture the redundancy of writing gestures which might be common across different graphemes. These fragments are extracted by positioning windows over text and then grouping similar fragments into clusters which are determined either for each of the writers separately or for a group of writers generating a universal set of patterns. A comparison of the two revealed that employing a common code book to represent writing samples reports better results on writer identification and verification as compared to representing them in a writer-specific space reading an identification rate of 84% against 81% and EER of 4.49 % against 5.44% on a data set of 650 writers.

We next used the orientation and curvature information in writing which is extracted by computing a set of features from writing contours at different levels of observation. These features were extracted by representing the contours by the chain code sequence and then by a set of polygons eliminating the minute writing details. The effectiveness of these features in characterizing the writer of a handwritten sample was then demonstrated by evaluating their performance on two data sets (IAM & RIMES) realizing identification rates of 89% and 85% and equal error rates of 2.46% and 4.87% on the two data sets respectively.

Finally, we combined the idea of redundant writing patterns with the contour-based features and achieved very promising results on tasks of writer identification and verification which are comparable to the best results achieved so far in the domain. We also conducted a study on the relevance of the proposed features using a feature selection process where we were able to maintain very good identification rates by keeping approximately 50% of the features only.

Our method was also implied in the project ANR-Graphem where based on our proposed features, we developed a software for the indexing and retrieval of medieval manuscripts that could eventually serve the palaeographers for classification of writing styles. Finally, we showed the generality of our method by employing it for signature verification and applying it on Arabic handwritten documents for writer recognition.

To end, we will enumerate some interesting research directions laid down by our work. First of all, for the extraction of frequent writing shapes, it would be better if one could determine the window size (for division of text) automatically depending upon the writing details rather than using a fixed size for all writings. This would in turn require either a representation that would allow a comparison of writing fragments at different sizes or a normalization of these fragments prior to comparison. The idea of redundant writing patterns may be developed further to study whether an individual who writes more than one script (for examples someone who writes French & Arabic) share some basic patterns/shapes across the two scripts. This could be realized by comparing the writer-specific code books computed from the texts written in the two scripts. It would be very interesting to analyze if one could recognize the writer of a text written in one script from the samples of same writer written using a different script or, identify the script of a given text based on the basic writing shapes present in the text, comparing them to a universal code book of the respective alphabet.

For the contour-based features we tried to capture the direction and curvature information in writing by a set of distributions and it would be a good idea to extract this information at different image resolutions which in fact will correspond to different levels of observations. The features at different resolutions can then be combined and their effect on the performance could be studied to identify the observations scales which might be more effective for the problem of writer recognition.

In addition to the applications that we presented, after a normalization that would allow to eliminate writer-dependent variations in the text, the method could be adapted for applications like character recognition or word spotting.

Finally, the author hopes that the content of this thesis would be of valuable help to the researchers working on writer recognition or similar problems. Although the target performance requirements for forensic applications have yet to be met, this work would nevertheless be an important contribution to the domain.

Appendix A

Résumé en Français

Introduction

Malgré les prédictions d'un monde sans papier et le développement des documents électroniques, les documents manuscrits ont gardé leur importance et les problèmes de l'identification et de l'authentification des auteurs ont constitué un domaine de recherche actif au cours de ces dernières années. Comparé au texte électronique ou imprimé, le texte manuscrit diffuse des informations additionnelles sur la personnalité de la personne qui a écrit. Il existe un certain degré de stabilité dans le modèle d'écriture d'un individu, ce qui permet d'identifier l'auteur pour lequel on a déjà vu un texte écrit.

La nécessité d'authentifier un texte écrit est un problème récurrent, non seulement du point de vue de la biométrie comportementale [Plamondon & Lorette, 1989] [Bensefia et al., 2005b] [Seropian & Vincent, 2002] [Srihari et al., 2002], mais également dans le contexte de la reconnaissance d'écriture [Nosary et al., 1999] en exploitant le principe de l'adaptation du système au type du scripteur. On utilise également l'écriture pour l'analyse des documents anciens dans un but d'indexation et de recherche.

La recherche dans le domaine a beaucoup évolué dans les années récentes et une grande variété de techniques ont été proposées qui peuvent être divisées en deux approches principales : les méthodes dépendantes et les méthodes indépendantes du texte.

Dans les méthodes dépendantes du texte, les échantillons d'écriture à comparer doivent contenir le même texte. Ces méthodes utilisent normalement la comparaison entre des caractères ou des mots de transcription connue et nécessitent donc que le texte soit reconnu ou segmenté (manuellement ou automatiquement) en caractères ou en mots avant de faire l'identification de scripteur. Ces études sont principalement motivées par des applications légales et visent à concevoir des algorithmes qui permettent d'extraire les caractéristiques qui sont utilisées par les examinateurs légaux de document. On peut noter que ces méthodes sont très contraignantes, aussi la majeure partie de la recherche ces dernières années s'est concentrée sur des approches indépendantes du texte.

Évidemment, les méthodes indépendantes du texte sont plus utiles pour les applications pratiques où l'intervention humaine est minimisée. Ceux-ci utilisent des caractéristiques extraites de l'image entière d'un texte ou d'une région d'intérêt. On a proposé une grande variété de techniques qui

peuvent être classées soit comme globales [Boulétreau et al., 1998] [Said et al., 2000]: basées sur l'aspect soit global de l'écriture, soit local. [Marti et al., 2001] [Seropian et al., 2003] identifient l'auteur à partir de l'extraction des caractéristiques locales de l'écriture. L'identification de scripteur basée sur la reconnaissance d'écriture a également été proposée dans la littérature [Schlapbach & Bunke, 2004] [Schlapbach & Bunke, 2006a].

Depuis quelques années, la tendance des recherches sur la reconnaissance de scripteur, s'est focalisée vers les méthodes basées sur les codebooks où l'écriture est segmentée en graphèmes qui sont ensuite comparés avec des éléments d'un codebook soit propre au scripteur [Bensefia et al., 2002] soit un codebook universel [Bensefia et al., 2005b] [Schomaker & Bulacu, 2004] [Bulacu & Schomaker, 2005]. Ces méthodes ont été très développées au cours des dernières années et elles ont démontré une grande performance pour l'identification de scripteur. La combinaison de ces caractéristiques de codebook avec d'autres caractéristiques de niveaux différents est également connue pour améliorer le taux d'identification [Bulacu & Schomaker, 2007].

Le problème de la reconnaissance du scripteur comprend les tâches d'identification et de vérification. Pour l'identification de scripteur, étant donné un échantillon manuscrit S inconnu et une base avec des échantillons de N scripteurs connus, l'objectif est de trouver le scripteur (ou une liste probable de scripteurs) de S dans la base de données. Pour la vérification de scripteur, étant donné deux échantillons de manuscrits S_1 et S_2 l'objectif est de déterminer si les deux ont été écrits par la même personne ou pas. En traitant de grandes bases de données, l'identification de scripteur peut également être employée comme une étape de filtrage préalable à la vérification [Bensefia et al., 2005b]. L'étape d'identification pourrait extraire un sous-ensemble des candidats probables à partir de la base de données et l'écriture de chaque candidat peut alors être comparée à l'écriture en question, soit par le système de vérification soit par un expert humain.

Nous avons développé une méthode efficace pour la reconnaissance automatique de scripteur à partir des images de texte manuscrit offline. Notre méthode repose sur deux aspects différents de l'écriture, la présence des formes redondantes dans l'écriture et des attributs visuels de l'écriture. En nous basant sur l'hypothèse qu'un individu utilise certaines formes plus fréquemment que les autres quand il écrit, nous espérons extraire ces formes en analysant des petits fragments d'écriture et en regroupant les formes similaires dans des classes. Ces classes sont déterminées soit pour chacun des scripteurs séparément ou pour un groupe de scripteurs générant un ensemble universel de formes. L'écriture en question est ensuite comparée à ces classes de formes produites. Ensuite, nous exploitons les deux importants attributs visuels de l'écriture, l'orientation et la courbure, qui permettent de distinguer une écriture d'une autre. Ces attributs sont extraits par le calcul d'un ensemble de caractéristiques à différents niveaux d'observation. Deux écritures sont ensuite comparées en calculant les distances entre leurs caractéristiques respectives. Enfin, nous combinons les deux facettes de l'écriture pour caractériser le scripteur d'un échantillon manuscrit.

Formes redondantes de l'écriture

Pour les caractéristiques basées sur le codebook, on cherche à mettre en évidence des détails fréquents dans une écriture. Cette redondance des formes est aussi la base des études dans [Bensefia et al., 2002] [Bensefia et al., 2005b] [Bulacu & Schomaker, 2007]. En effet, dans toutes ces méthodes, l'approche est liée à la façon dont les lettres sont tracées et segmentées comme si le but était de lire le texte. Nous pensons que la reconnaissance du scripteur est indépendante de ce qui est écrit et est plutôt liée à la manière physique par laquelle des lignes ou des boucles sont produites. Ainsi l'échelle de l'observation peut être inférieure à celle d'une lettre. Aucune interprétation sémantique des portions de trait analysé n'est nécessaire. Les fragments que nous considérons sont des petites parties d'un texte manuscrit qui ne contiennent aucune information sémantique. Ces formes ont été extraites par un découpage adaptatif de l'écriture en imagettes, puis l'extraction de descripteurs de chacune des imagettes et enfin une classification groupant les imagettes similaires.

Le découpage de l'écriture est utilisé pour extraire des éléments inhérents au scripteur, donc c'est une partie importante du processus. Il doit être dépendant du tracé pour que les contenus puissent être comparables. On a choisi un découpage en carrés de taille $n \times n$ où la taille n a été choisie empiriquement égale à 13×13 [Siddiqi & Vincent, 2007]. En utilisant un algorithme adaptatif de positionnement, ces fenêtres sont placées sur le texte divisant ainsi l'écriture en un grand nombre de petites imagettes [Siddiqi & Vincent, 2008]. Les fenêtres sont consécutivement positionnées en suivant le squelette de l'écriture. Sans chercher à reconstituer l'ordre du tracé, nous réalisons un suivi du trait.

Une fois que le texte est divisé en imagettes, nous procédons à l'extraction de descripteurs de forme sur chacune. La position de la trace écrite dans la fenêtre est d'abord normalisée afin que les caractéristiques calculées soient invariantes à la *translation*. Il n'est cependant pas souhaitable dans notre cas d'avoir: i) l'invariance à l'*échelle*: puisque nous ne prévoyons pas que le scripteur modifie la taille de l'écriture dans un même échantillon d'écriture et ii) l'invariance à la *rotation*: comme une forme et sa version tournée ne sont pas produites par le même geste de la main.

L'ensemble des descripteurs que l'on calcule pour chaque imagette comprend les histogrammes horizontaux et verticaux, les profils supérieur et inférieur et une série de descripteurs de forme. Chaque imagette est alors représentée par un vecteur de dimension $d = 4n + 6$, où n est la taille de la fenêtre.

Représentant chaque imagette par un vecteur, on procède ensuite à leur regroupement de manière à réduire la quantité de données et à rendre le résultat indépendant de la quantité de texte étudié. L'objectif est de grouper les formes produites par le même geste de la main dans les mêmes classes. La méthodologie de classification dépendra du type de codebook selon qu'il soit propre au scripteur ou universel.

Pour un codebook propre au scripteur, nous avons évalué un certain nombre d'algorithmes qui ne demandent pas un choix préalable du nombre de classes car ce paramètre va varier d'une écriture à une autre. Après une série d'expériences sur l'ensemble de validation, nous avons choisi une classification hiérarchique pour extraire les formes redondantes de l'écriture et générer le codebook propre au scripteur. Pour chaque classe dans le codebook, on estime sa probabilité d'apparition et aussi on calcule le vecteur moyen représentant la classe. L'ensemble de ces probabilités peuvent être considérées comme une distribution h^D , où chaque cardinal de classe de h^D représenterait la probabilité d'émission de la forme respective par l'auteur du document D . Cette distribution est ensuite utilisée pour caractériser l'auteur d'un échantillon donné.

Nous prolongeons la même idée en générant un codebook universel. Puisque nous avons choisi de travailler sur un codebook qui n'est pas produit à partir de l'ensemble de données à l'étude, le codebook est généré à partir d'échantillons manuscrits de la base RIMES [Grosicki et al., 2008] pour évaluer les écritures du jeu de données IAM [Marti & Bunke, 2002] et vice versa. Un total de 50 échantillons d'écriture sont utilisés pour produire le codebook tandis que les fragments sont regroupés en utilisant l'algorithme *k-means* dans l'espace de caractéristiques avec k fixé à 100 après les évaluations sur l'ensemble de validation. Une fois le codebook généré, on trouve pour chaque scripteur, les fréquences de production des formes dans le codebook, la répartition étant caractéristique du scripteur.

Les attributs visuels de l'écriture

Nous analysons aussi deux importants attributs visuels de l'écriture qui permettront de capturer le style d'écriture de son auteur, l'orientation et la courbure des traits. L'information de l'orientation et de la courbure dans une écriture est capturée par un ensemble de caractéristiques. Ces caractéristiques sont calculées à partir des contours de l'écriture qui encapsulent le style d'écriture de l'auteur et permettent de préserver des variations (qui dépendent du scripteur) entre les formes de caractères. Nous avons choisi de représenter les contours de deux manières qui correspondent à deux échelles d'observation et deux niveaux de détails différents. Ces représentations incluent les chaînes de Freeman et un ensemble de polygones approximant les contours.

De la chaîne de Freeman associée aux contours, nous calculons un ensemble de caractéristiques, d'abord à un niveau global, puis à partir des fragments de contour dans des fenêtres d'observation de petite taille. Au niveau global, on utilise la distribution des codes de Freeman et de leurs différences, la distribution des indices de courbure et la distribution des paires et des triplets des codes de la chaîne. Au niveau local, on trouve comment les orientations de Freeman et leurs différences sont réparties au sein de petites fenêtres d'observation. Ces fenêtres sont positionnées sur l'image du contour de la même manière que celles utilisées pour la production de codebook.

On calcule également des caractéristiques similaires à un niveau d'observation différent, en estimant les contours par un ensemble de polygones. Cela ne correspond pas seulement à une

échelle d'observation lointaine, mais les caractéristiques calculées sont aussi plus robustes à des distorsions. Nous procédons d'abord à une approximation des contours par un ensemble de segments de ligne en employant un algorithme séquentiel de polygonisation [Wall & Danielsson, 1984]. Les caractéristiques sont ensuite calculées qui comprennent la distribution de pentes et de mesures des angles formés par les segments consécutifs, les mêmes distributions, pondérées par la longueur des segments (ayant une pente/angle donné) et la distribution des longueurs des segments dans une écriture.

On a extrait ainsi un ensemble de quatorze distributions (normalisées) qui permettent de représenter une image du document dans un espace de dimension totale 586 (Tableau A-1).

Tableau A-1 Caractéristiques et leurs dimensionalités

Carac.	Description	Dimension
<i>f1</i>	Distribution des codes de Freeman	8
<i>f2</i>	Distribution des différences (1 ^{er} ordre) de codes de Freeman	7
<i>f3</i>	Distribution des différences (2 ^e ordre) de codes de Freeman	8
<i>f4</i>	Distribution de paires de codes de Freeman	44
<i>f5</i>	Distribution de triplets de codes de Freeman	236
<i>f6</i>	Distribution des indices de courbure	11
<i>f7</i>	Distribution locale des directions de trait	80
<i>f8</i>	<i>f2</i> calculé localement	70
<i>f9</i>	<i>f3</i> calculé localement	80
<i>f10</i>	Distribution des pentes de segments	8
<i>f11</i>	Distribution pondérée des pentes de segments	8
<i>f12</i>	Distribution des courbures	8
<i>f13</i>	Distribution pondérée des courbures	8
<i>f14</i>	Distribution des longueurs de segments	10
<i>Total:</i>		586

Reconnaissance de scripteur

Pour la reconnaissance de scripteur, après avoir évalué un certain nombre de mesures de distance, nous avons gardé la distance du χ^2 pour calculer les distances entre les valeurs des caractéristiques (histogrammes) de deux échantillons d'écriture.

$$\chi^2(p, q) = \sum_{i=1}^{dim} \frac{(p_i - q_i)^2}{p_i + q_i}$$

où p et q représentent les deux histogrammes à comparer, p_i et q_i sont les fréquences associées aux classes i et j des histogrammes et dim représente le nombre total de classes dans l'histogramme.

Pour les caractéristiques basées sur les contours, le calcul de (dis)similarité entre deux documents D et Q est relativement simple et est donné par la distance entre les caractéristiques f_i^D et f_i^Q . Pour les caractéristiques basées sur le codebook, il est tout d'abord nécessaire de diviser le texte dans l'image requête Q en petites imagerie.

Lorsqu'on utilise un codebook propre au scripteur, chacun des fragments de l'image requête est attribué à l'une des classes dans le document de référence. Pour comparer le document en question Q avec un document de référence D , pour chaque forme dans le document de test, on retrouve la plus proche classe dans le document D . Nous trouvons par conséquent comment les entrées dans le codebook du scripteur de D sont distribuées dans le document Q . Ainsi, en fait, le document en question est représenté dans l'espace du document de référence et les deux écritures sont comparées en calculant la distance entre les distributions respectives h^D et h^{DQ} .

Pour le codebook universel, on compare les imagerie extraites du document en question aux formes contenues dans le codebook et donc on trouve les fréquences d'occurrence des formes du codebook pour un scripteur particulier. Deux écritures sont ensuite comparées en calculant la distance entre les distributions de probabilité respectives h^D et h^Q de produire les formes du codebook.

L'identification de scripteur est effectuée en calculant la distance entre l'image requête Q et toutes les images dans la base d'apprentissage en utilisant une caractéristique sélectionnée, l'auteur de Q étant identifié comme l'auteur du document qui donne la distance minimale. Cela correspond à la classification du plus proche voisin (knn avec $k = 1$).

Pour la vérification de scripteur, on calcule la distance entre deux échantillons donnés et on considère qu'ils sont écrits par la même personne si la distance est inférieure à un seuil prédéfini. Au-delà de la valeur du seuil, on considère que les échantillons sont écrits par des scripteurs différents. En faisant varier le seuil d'acceptation, des courbes ROC sont calculées et la performance de la vérification est quantifiée par le « equal error rate » (EER), le point de la courbe où le taux de fausses acceptations (FAR) est égal au taux de faux rejets (FRR).

Résultats expérimentaux

Concernant les résultats expérimentaux, nous avons d'abord effectué une évaluation comparative des codebooks universel et propre au scripteur sur les 650 scripteurs dans la base IAM. Une analyse comparative des performances des deux révèle que représenter les écritures dans un espace commun conduit à de meilleurs résultats sur l'identification et la vérification par rapport à une représentation dans un espace spécifique au scripteur (un taux d'identification de 84% contre 81% et EER de 4,49 % contre 5,44% comme indiqué dans le Tableau A-2).

Tableau A-2 Résultats (en pourcentages) des caractéristiques basées sur des codebooks

Data Set	IAM 650 Writers		
	Top1	Top10	EER
<i>Writer-specific</i>	81	94	5,44
<i>Universal</i>	84	96	4,49

Pour les caractéristiques basées sur le contour, nous avons réalisé une série détaillée d'expériences pour évaluer la performance des caractéristiques individuelles, ainsi que leurs diverses combinaisons. En plus de la base IAM, nous avons également testé ces caractéristiques sur 375 scripteurs de la base de données RIMES. Nous avons réalisé un taux d'identification de 89% (Top10: 97%) sur les écritures dans la base IAM et de 85% (Top10: 93%) sur les images de RIMES tandis que les EER correspondant étaient de 2,46% et 4,87% respectivement. Les résultats de la combinaison de différentes caractéristiques sont résumés dans le Tableau A-3. Nous avons également étudié l'effet de combiner les caractéristiques de contour avec celles de codebook. En utilisant ces caractéristiques, on obtient des taux d'identification qui sont comparables aux meilleurs résultats rapportés à ce jour pour l'identification de scripteur hors ligne (Tableau A-4).

Tableau A-3 Résultats (en pourcentages) des caractéristiques basées sur des contours

Data Set	IAM 650 Writers			RIMES 375 Writers		
	Top1	Top10	EER	Top1	Top10	EER
<i>Global</i>	81	93	4,08	77	92	6,18
<i>Local</i>	81	95	3,76	77	89	7,85
<i>Polygon Based</i>	83	97	2,77	81	93	5,16
<i>Global & Local</i>	83	96	3,81	80	91	6,65
<i>Global & Polygon Based</i>	85	96	3,32	82	92	5,92
<i>Local & Polygon Based</i>	87	97	3,03	83	93	5,11
<i>All Features</i>	89	97	2,46	85	93	4,87

Nous avons également mené une étude sur la pertinence des caractéristiques proposées en utilisant un processus de sélection de caractéristiques. En considérant les fréquences de sélection, nous les avons regroupées en trois catégories, les caractéristiques indispensables, partiellement pertinentes et non pertinentes [Pervouchine & Leedham, 2007]. Nous avons été capable de maintenir de très bons taux d'identification en gardant environ 50% des caractéristiques uniquement.

Enfin, nous avons démontré la généralité de notre méthode en l'employant pour la classification de manuscrits médiévaux, pour la vérification des signatures et la reconnaissance de scripteur sur

les documents manuscrits en Arabe. Les résultats sur chacune de ces applications sont très prometteurs, validant ainsi les caractéristiques proposées.

Tableau A-4 Comparaison de performance de l'identification de scripteur sur la base IAM

	Scripteurs	Echantillon/ scripteur	Performance
[Marti et al., 2001]	20	5	90.7%
[Bensefia et al., 2005b]	150	2	86%
[Schlapbach & Bunke, 2006a]	100	5/4	98,46%
[Bulacu & Schomaker, 2007]	650	2	89%*
Méthode propose	650	2	91% / 89%*

Il y a aussi quelques pistes de recherche intéressantes prévues par notre travail. Tout d'abord, en ayant la capacité d'ajuster automatiquement la taille des fenêtres locales (dépendant des détails de l'écriture) pendant la phase de découpage de l'écriture, on peut avoir un système plus robuste. L'idée des formes redondantes d'écriture pourrait être développée afin d'étudier si une personne qui écrit plusieurs langues ou utilise différents alphabets partage certaines caractéristiques de base/formes entre les deux cas. Il serait très intéressant d'analyser si l'on pouvait reconnaître le scripteur d'un texte écrit dans un alphabet à partir d'échantillons du même scripteur dans un alphabet différent. La capacité d'analyser le texte à différentes résolutions et avec différentes épaisseurs des traits peut également être inclus dans le système.

Pour finir, l'auteur espère que le contenu de cette thèse sera d'une aide précieuse aux chercheurs qui travaillent sur la reconnaissance de scripteur ou des problèmes similaires. Bien que l'objectif de performance exigé pour les applications « forensic » ne soit pas encore rempli, ce travail est néanmoins une contribution importante au domaine.

* *k-nn* using a leave-one-out approach

Appendix B

Results: Codebook Features

Table B-1 Writer identification rates (on 150 writers of IAM dataset) on different clustering methods for writer-specific and universal codebooks

Codebook	Writer Specific		Universal	
	Top1	Top10	Top1	Top10
Sequential clustering	83	96	91	97
Iterative sequential clustering	80	92	86	95
Minimum spanning tree clustering	77	89	80	92
Hierarchical clustering	86	97	92	97

Table B-2 Writer identification performance (on 300 writers of IAM dataset) as a function of codebook size

Codebook Size	50	100	250	300	350	400	500	750
Top1	86	90	90	89	88	85	84	86
Top10	94	95	95	95	95	95	95	94

Table B-3 Writer identification performance (on 300 writers of IAM dataset) as a function of the number of samples (writers) used to generate the codebook

Number of samples	1	5	10	25	50	100
Top1	84	89	89	89	90	91
Top10	94	95	95	95	95	96

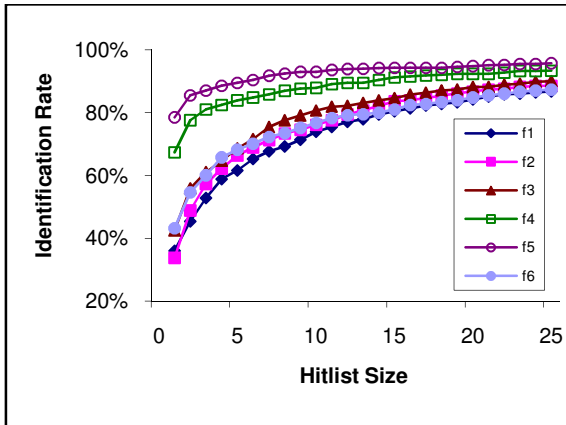
Table B-4 Writer identification rates (on 300 writers of IAM dataset) as function of amount of text

Codebook	Writer Specific		Universal	
	Top1	Top10	Top1	Top10
One word	07	22	10	34
Two words	17	41	18	50
One line	21	51	29	66
Two lines	43	78	48	88
Three lines	58	85	68	94
Four lines	71	92	78	94
Complete page	85	94	90	95

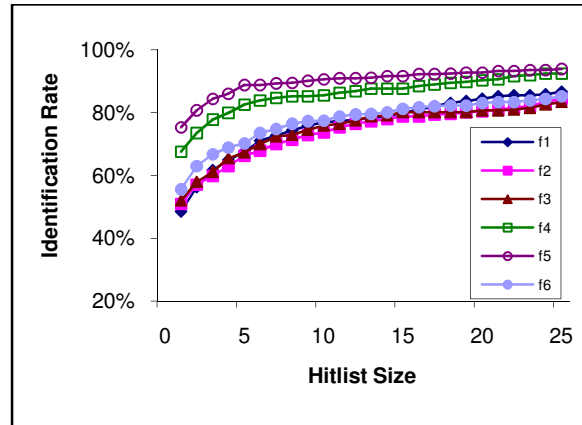
Appendix C

Results: Contour Based Features

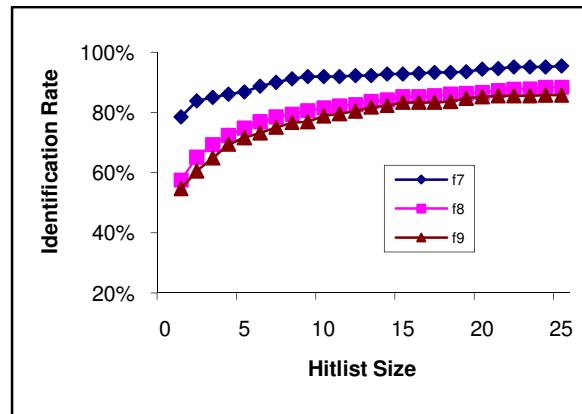
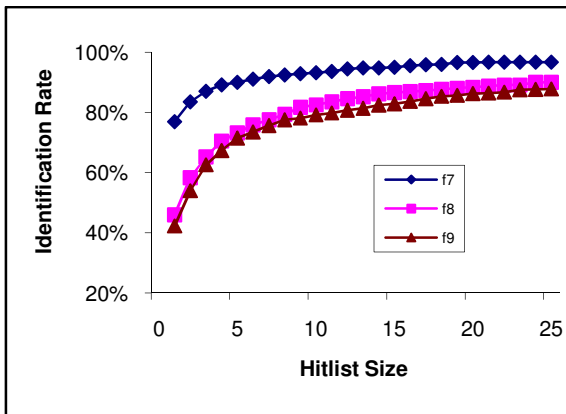
IAM Data Set



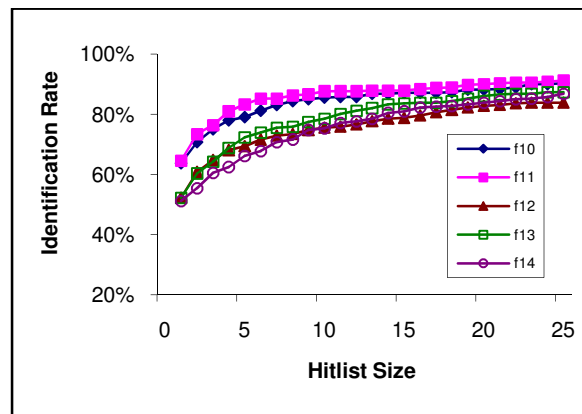
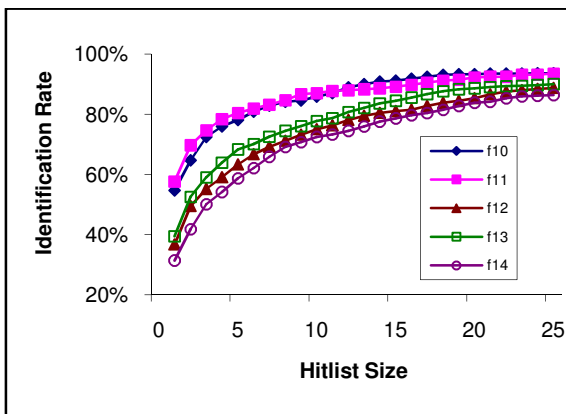
RIMES Data Set



Global Chain Code Features



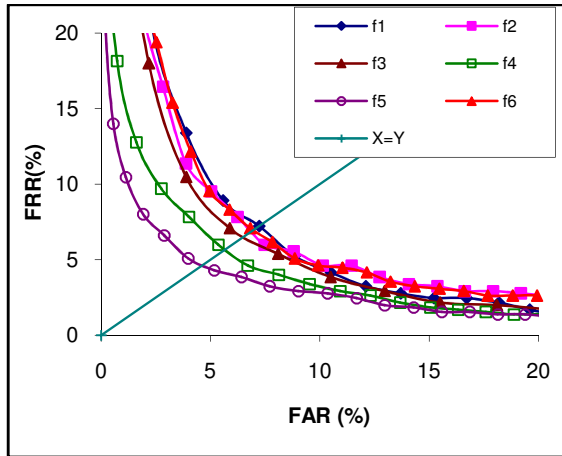
Local (Window-based) Chain Code Features



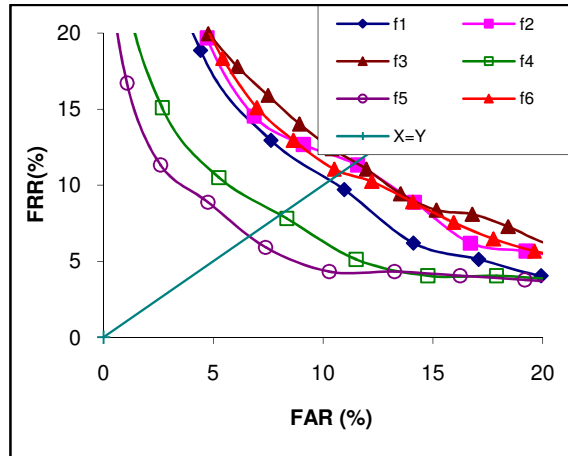
Polygon-based Features

Performance of individual features on Writer Identification

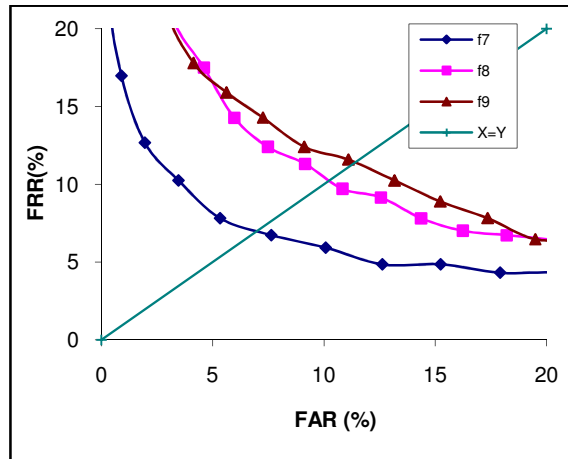
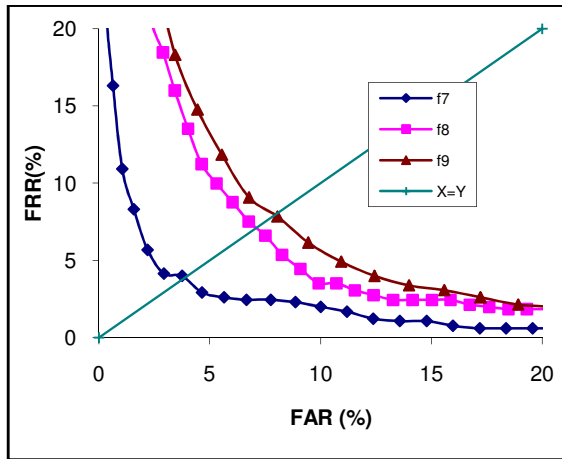
IAM Data Set



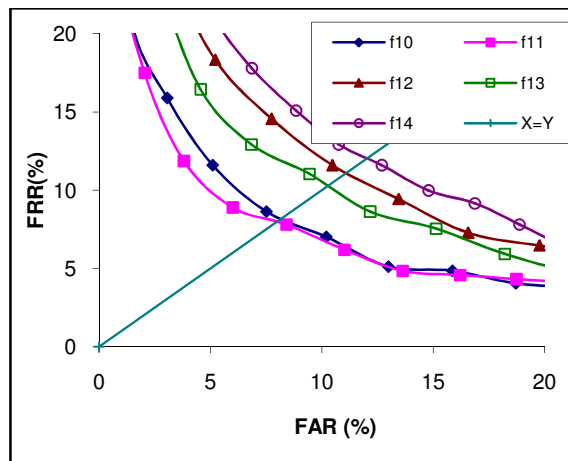
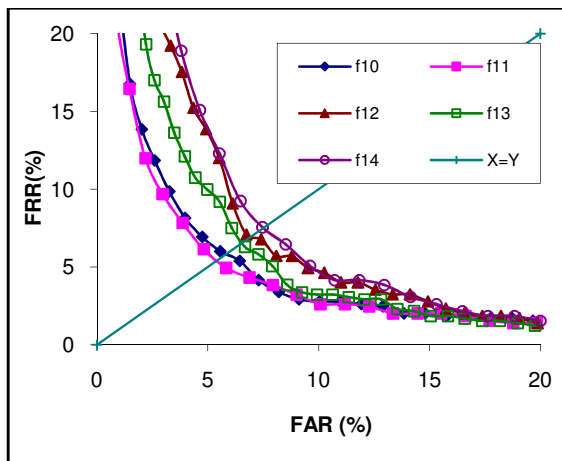
RIMES Data Set



Global Chain Code Features



Local (Window-based) Chain Code Features



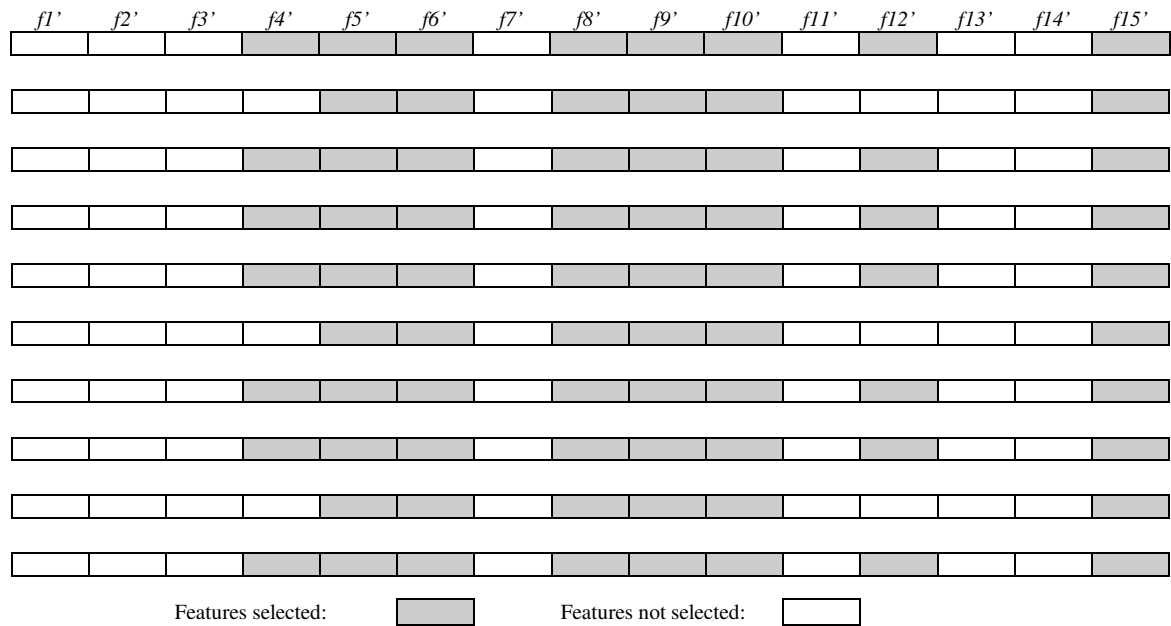
Polygon-based Features

ROC curves for individual features

Appendix D

Feature Selection Results

This appendix presents the features $f1' - f15'$ selected in ten runs of the genetic algorithm where f_i' represents the feature f_i with a reduced dimension (as discussed in section 4.6.2.3).



Features (reduced dimension) selected in ten runs of the GA

Run	Features	Top 1	Top 10
1	$f4', f5', f6', f8', f9', f10', f12', f15'$	90.15	98
2	$f5', f6', f8', f9', f10', f15'$	89.85	97.69
3	$f4', f5', f6', f8', f9', f10', f12', f15'$	90.15	98
4	$f4', f5', f6', f8', f9', f10', f12', f15'$	90.15	98
5	$f4', f5', f6', f8', f9', f10', f12', f15'$	90.15	98
6	$f5', f6', f8', f9', f10', f15'$	89.85	97.69
7	$f4', f5', f6', f8', f9', f10', f12', f15'$	90.15	98
8	$f4', f5', f6', f8', f9', f10', f12', f15'$	90.15	98
9	$f5', f6', f8', f9', f10', f15'$	89.85	97.69
10	$f4', f5', f6', f8', f9', f10', f12', f15'$	90.15	98
Average	-	90.06	97.90

Performance of selected feature subsets on 650 writers of the IAM data set

Appendix E

Sample Images

Sentence Database

A06-008

Mr. Harold Wilson, Shadow Chancellor, jumped up to offer the Government an easy passage for such legislation. "Why don't you make proposals to legislate in the autumn?" Mr. Wilson asked. "We wouldn't call it an Autumn Budget. You can call it a Taxation Management Bill, if you like." While Mr. Wilson was speaking, the Prime Minister and Mr. Lloyd had a whispered conversation.

Mr. Harold Wilson, Shadow Chancellor, jumped up to offer the Government an easy passage for such legislation. "Why don't you make proposals to legislate in the autumn?" Mr. Wilson asked. "We wouldn't call it an Autumn Budget. You can call it a Taxation Management Bill, if you like." While Mr. Wilson was speaking, the Prime Minister and Mr. Lloyd had a whispered conversation.

Name:

Dataset : IAM

Noémie Renou
9 rue du doyen J. Parisot
54630 Flavigny / Moselle
03.37.75.80.04
Références client = HZETU71

à Les 3 Luifs
Bourg Dammarie / Leing
45230
Bourg Dammarie / Leing

Flavigny, le 27/07/06

Madame, monsieur,

Suite à mon emménagement avec mon conjoint,
je vous informe du changement de mes coordonnées
téléphoniques (précisées ci-dessus) afin que vous
puissiez mettre à jour mon dossier.

Avec mes sincères remerciements,

M^{me} Renou

~~NR~~

D qui die nob̄ extulisti in tui
 martiris māmetis ueneratio
 ne salutiferū. ccede p̄pitiu
 ut qui ei nobile celebram̄ triūphū
 p̄sidiū sentiam̄ in p̄petuū. p̄ sc̄r
S rata tibi dñe p̄ tui martiris mam
 metis que tue offerim̄ dignitati sin

dicatis: non in contentione et emulati
 one. Sed induimini dominū ih̄m xpm.
secunda quarta lectio epl̄e beati Jacobi ap̄t̄i.
Patientes estote: usq; ad adue
 tum domini. Ecce agricola expectat
 pretiosum fructum terre patient̄ ferens?
 donec accipiat remanentem et serotinu.

CODE ↓	PLACE ↓	
5154	المحارزة 18	المحارزة 18 5154
2116	زنوش	زنوش 2116
8041	قربص	قربص 8041
3097	ربايح سيدي ظاهر	ربايح سيدي ظاهر 3097
5032	مزدور	مزدور 5032
9040	تبرسق	تبرسق 9040
4016	بني كلثوم	بني كلثوم 4016
4111	أم التمر	أم التمر 4111
8132	سوق الجمعة	سوق الجمعة 8132
3134	حيط الواد	حيط الواد 3134
4134	شماخ	شماخ 4134
7063	أوتيك الجديدة	أوتيك الجديدة 7063

Age: < 20
 21 - 30
 31 - 40
 > 40

Profession : Étudiant/élève
 Enseignant
 Administratif
 Autre

Nom:

Touhami Wala

Ville:

Tunis

Dataset : IFN/ENIT

Sample Signatures

Appendix F

Graphem Retrieval Software: User Interface

Weights for each of the features

Retrieved list of similar images: Ranks, Image Names & Distances

The screenshot displays the Graphem Retrieval Software interface, divided into several sections:

- Query:** A text box contains the filename "IRHT_P_000821.jpg_yosef.jpg" with an "Open" button.
- Parameters:** A grid of 14 feature weight sliders (F1-F14), each set to 1. A red circle highlights this section.
- Results:** A list of search results with columns for rank, image name, and distance. A red circle highlights this section.
- Images:** A section showing the "Query Image" and a "Result Image". Both are handwritten text documents. The "Result Image" is selected, and its distance is shown as 0.039139.
- View:** Buttons for "100%", "Full", and "40%" (or "48%") to zoom the images.
- Classification:** Buttons for "Select as Query" and "Stop".

Rank	Image Name	Distance
1	IRHT_P_000821.jpg_...	0
2	IRHT_P_000164.jpg_...	0.0248344
3	IRHT_P_003713.jpg_...	0.039139
4	IRHT_P_000910.jpg_...	0.0409129
5	IRHT_P_000196.jpg_...	0.0418935
6	IRHT_P_000145.jpg_...	0.04246
7	IRHT_P_000121.jpg_...	0.0430841
8	IRHT_P_000685.jpg_...	0.0432004
9	IRHT_P_000872.jpg_...	0.0440552
10	IRHT_P_000019.jpg_...	0.0452475
25	IRHT_P_003750.jpg_...	0.06313
50	IRHT_P_003859.jpg_...	0.0779638
100	IRHT_P_002131.jpg_...	0.108886
150	IRHT_P_000089.jpg_...	0.143952
200	IRHT_P_002100.jpg_...	0.199134
250	IRHT_P_000351.jpg_...	0.252744
300	IRHT_P_000166.jpg_...	0.52087

Query Image

Image selected from the retrieved list

Bibliography

- [Ahmad et al., 2003] Ahmad, M. B., Park, J.-A., Chang, M. H., Shim, Y.-S., & Choi, T.-S. (2003). Shape registration based on modified chain codes. In *Advanced Parallel Processing Technologies* (pp. 600–607).: Springer Berlin / Heidelberg.
- [Aiolli et al., 1999] Aiolli, F., Simi, M., Sona, D., Sperduti, A., Starita, A., & Zaccagnini, G. (1999). Spi: a system for palaeographic inspections. *AIIA Notizie*, 4, 34–38.
- [Arazi, 1977] Arazi, B. (1977). Handwriting identification by means of run-length measurements. *IEEE Trans. Syst., Man and Cybernetics*, 7(12), 878–881.
- [Arazi, 1983] Arazi, B. (1983). Automatic handwriting identification based on the external properties of the samples. *IEEE Trans. Syst., Man and Cybernetics*, 13(4), 635–642.
- [Bach et al., 1996] Bach, J. R., Fuller, C., Gupta, A., Hampapur, A., Horowitz, B., Humphrey, R., Jain, R. C., & Shu, C.-F. (1996). Virage image search engine: an open framework for image management. In *SPIE Conf. on Storage and Retrieval for Still Image and Video Databases IV*, volume 2076.
- [Baggett, 2004] Baggett, B. (2004). *Handwriting Analysis 101-The Basic Traits*. Empressé Publishing.
- [Bandera et al., 1999] Bandera, A., Urdiales, C., Arrebola, F., & Sandoval, F. (1999). 2d object recognition based on curvature functions obtained from local histograms of the contour chain code. *Pattern Recognition Letters*, 20, 49 – 55.
- [Bar-Yosef, 2005] Bar-Yosef, I. (2005). Input sensitive thresholding for ancient hebrew manuscript. *Pattern Recognition Letters*, 26, 1168 – 1173.
- [Belongie et al., 2002] Belongie, S., Malik, J., & Puzicha, J. (2002). Shape matching and object recognition using shape contexts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(4), 509–522.
- [Bensefia et al., 2002] Bensefia, A., Nosary, A., Paquet, T., & Heutte, L. (2002). Writer identification by writer's invariants. In *IWFHR '02: Proceedings of the Eighth International Workshop on Frontiers in Handwriting Recognition* (pp. 274–279). Washington, DC, USA: IEEE Computer Society.
- [Bensefia et al., 2003] Bensefia, A., Paquet, T., & Heutte, L. (2003). Information retrieval based writer identification. In *IGS'03: In Proceedings of the 11th Conference of the International Graphonomics Society*.

- [Bensefia et al., 2005a] Bensefia, A., Paquet, T., & Heutte, L. (2005a). Handwritten document analysis for automatic writer recognition. *Electronic Letters on Computer Vision and Image Analysis, ELCVIA*, 5(2), 72–86.
- [Bensefia et al., 2005b] Bensefia, A., Paquet, T., & Heutte, L. (2005b). A writer identification and verification system. *Pattern Recognition Letters*, 26(13), 2080–2092.
- [Boulétreau, 1997] Boulétreau, V. (1997). *Vers un classement de l'écrit par des méthodes fractales*. PhD thesis.
- [Boulétreau et al., 1995] Boulétreau, V., Vincent, N., & Emptoz, H. (1995). A writing qualification invariant towards line thickness and resolution changings. In *ACCV'95: In Proceedings of the Asian Conference on Computer Vision* (pp. 325–329).
- [Boulétreau et al., 1998] Boulétreau, V., Vincent, N., Sabourin, R., & Emptoz, H. (1998). Handwriting and signature: One or two personality identifiers? In *ICPR '98: Proceedings of the 14th International Conference on Pattern Recognition-Volume 2* (pp. 1758–1760). Washington, DC, USA: IEEE Computer Society.
- [Bulacu & Schomaker, 2003] Bulacu, M. & Schomaker, L. (2003). Writer style from oriented edge fragments. In *Proc. of the 10th Int. Conference on Computer Analysis of Images and Patterns* (pp. 460–469).: Springer.
- [Bulacu & Schomaker, 2005] Bulacu, M. & Schomaker, L. (2005). A comparison of clustering methods for writer identification and verification. In *ICDAR '05: Proceedings of the Eighth International Conference on Document Analysis and Recognition* (pp. 1275–1279). Washington, DC, USA: IEEE Computer Society.
- [Bulacu & Schomaker, 2006] Bulacu, M. & Schomaker, L. (2006). Combining multiple features for text-independent writer identification and verification. In *IWFHR'06: 10th International Workshop on Frontiers in Handwriting Recognition*.
- [Bulacu & Schomaker, 2007] Bulacu, M. & Schomaker, L. (2007). Text-independent writer identification and verification using textural and allographic features. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(4), 701–717. Student Member-Bulacu,, Marius and Member-Schomaker,, Lambert.
- [Bulacu et al., 2003] Bulacu, M., Schomaker, L., & Vuurpijl, L. (2003). Writer identification using edge-based directional features. In *ICDAR '03: Proceedings of the Seventh International Conference on Document Analysis and Recognition* (pp. 937–941). Washington, DC, USA: IEEE Computer Society.

- [Chen & Chen, 1997] Chen, T.-Y. & Chen, C.-J. (1997). Improvements of simple genetic algorithm in structural design. *International Journal for Numerical Methods in Engineering*, 40, 1323–1334.
- [Chiu & Tseng, 1997] Chiu, H.-P. & Tseng, D.-C. (1997). A feature-preserved thinning algorithm for handwritten chinese characters. *Signal Processing*, 58(2), 203–214.
- [Chung & Wong, 1998] Chung, Y. Y. & Wong, M. T. (1998). High accuracy handwritten character recognition system using contour sequence moments. In *ICSP'98: Proceedings of the fourth International Conference on Signal Processing* (pp. 1249 – 1252).
- [Costa & Jr, 2001] Costa, L. D. F. & Jr, R. M. C. (2001). *Shape Analysis and Classification: Theory and Practice*. CRC Press.
- [Crettez, 1995] Crettez, J.-P. (1995). A set of handwriting families: style recognition. In *ICDAR '95: Proceedings of the Third International Conference on Document Analysis and Recognition* (pp. 489–494).
- [Dos Santos et al., 2009] Dos Santos, E. M., Sabourin, R., & Mauping, P. (2009). Overfitting cautious selection of classifier ensembles with genetic algorithms. *International Journal of Information Fusion*, 10, 150–162.
- [Duda & Hart, 2000] Duda, R. O. & Hart, P. E. (2000). *Pattern Classification and Scene Analysis, 2nd Edition*. John Wiley & Sons Inc.
- [Eaton, 1938] Eaton, H. D. (1938). Handwriting a neurological study. *California and Western Medicine*, 48(6), 430–435.
- [Eglin et al., 2007] Eglin, V., Bres, S., & Carlos, R. (2007). Hermite and gabor transforms for noise reduction and handwriting classification in ancient manuscripts. *International Journal on Document Analysis and Recognition*, 9(2), 101–122.
- [Eldridge et al., 1984] Eldridge, M., Nimmo-Smith, I., Wing, A., & Totty, R. (1984). The variability of selected features in cursive handwriting: Categorical measures. *Journal of the Forensic Science Society*, 24, 179–219.
- [Feng & Tan, 2004] Feng, M.-L. & Tan, Y.-P. (2004). Contrast adaptive binarization of low quality document images. *IEICE Electronics Express*, 1(16), 501–506.
- [Fisher, 1995] Fisher, Y. (1995). *Fractal Image Compression : Theory and Application*. Springer-Verlag, New York.
- [Freeman, 1974] Freeman, H. (1974). Computer processing of line-drawing images. *Computing Surveys*, 6(1), 57–97.

- [Friedman & Kandel, 1999] Friedman, M. & Kandel, A. (1999). *Introduction to Pattern Recognition : Statistical, Structural, Neural and Fuzzy Logic Approaches*. World Scientific Publishing Company.
- [Gilloux, 1994] Gilloux, M. (1994). Writer adaptation for handwritten word recognition using hidden markov models. In *ICPR'94: In Proceedings of the 12th International Conference on Pattern Recognition* (pp. 135–139).
- [Greening et al., 1995] Greening, C., Sagar, V., & Leedham, C. (1995). Automatic feature extraction for forensic purposes. In *Proceedings of the Fifth International Conference on Image Processing and its Applications* (pp. 409 – 414).
- [Grosicki et al., 2008] Grosicki, E., Carré, M., Brodin, J.-M., & Geoffrois, E. (2008). Rimes evaluation campaign for handwritten mail processing. In *ICFHR'08: In Proceedings of 11th Int'l Conference on Frontiers in Handwriting Recognition*.
- [Guyon & Elisseef, 2003] Guyon, I. & Elisseef, A. (2003). An introduction to variable and feature selection. *Journal of Machine Learning Research*, 3, 1157–1182.
- [Guyon et al., 2006] Guyon, I., Gunn, S., Nikravesh, M., & Zadeh, L. A., Eds. (2006). *Feature Extraction: Foundations and Applications*, volume 207 of *Studies in Fuzziness and Soft Computing*. Springer.
- [Guyon et al., 1994] Guyon, I., Schomaker, L., Plamondon, R., Liberman, M., & Janet, S. (1994). Unipen project of on-line data exchange and recognizer benchmarks. In *ICPR'94: In Proceedings of the 12th International Conference on Pattern Recognition* (pp. 29–33).
- [Heutte et al., 1998] Heutte, L., Paquet, T., Moreau, J. V., Lecourtier, Y., & Olivier, C. (1998). A structural/statistical feature based vector for handwritten character recognition. *Pattern Recognition Letters*, 19, 629–641.
- [Huber & Headrick, 1999] Huber, R. A. & Headrick, A. (1999). *Handwriting Identification: Facts and Fundamentals*. CRC Press.
- [Hull, 1994] Hull, J. (1994). A database for handwritten text recognition research. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16(5), 550–554.
- [Jain & Dubes, 1988] Jain, A. K. & Dubes, R. C. (1988). *Algorithms for Clustering Data*. Prentice-Hall, Inc.
- [Jain et al., 1999] Jain, A. K., Murty, M. N., & Flynn, P. J. (1999). Data clustering: a review. *ACM Computing Surveys*, 31(3), 264–323.

- [Joutel et al., 2007a] Joutel, G., Eglin, V., Bres, S., & Emptoz, H. (2007a). Curvelets based feature extraction of handwritten shapes for ancient manuscripts classification. In SPIE (Ed.), *Document Recognition and Retrieval XIV* (pp. 0D1–0D12).
- [Joutel et al., 2007b] Joutel, G., Eglin, V., Bres, S., & Emptoz, H. (2007b). Curvelets based queries for cbir application in handwriting collections. In *ICDAR '07: Proceedings of the Ninth International Conference on Document Analysis and Recognition (ICDAR 2007) Vol 2* (pp. 649–653). Washington, DC, USA: IEEE Computer Society.
- [Joutel et al., 2008] Joutel, G., Eglin, V., & Emptoz, H. (2008). A complete pyramidal geometrical scheme for text based image description and retrieval. In *ICISP '08: Proceedings of the 3rd international conference on Image and Signal Processing* (pp. 471–480).
- [Ke Liu & Suen, 1999] Ke Liu, Y. S. H. & Suen, C. Y. (1999). Identification of fork points on the skeletons of handwritten chinese characters. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(10), 1095–1100.
- [Kim & Govindaraju, 1997] Kim, G. & Govindaraju, V. (1997). A lexicon driven approach to handwritten word recognition for real-time applications. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(4), 366–379.
- [Kohonen, 1988] Kohonen, T. (1988). *Self-Organization and Associative Memory*. Springer; 2nd edition.
- [Koppenhaver, 2007] Koppenhaver, K. M. (2007). *Forensic Document Examination: Principles and Practice*. Humana Press.
- [Lee & Street, 2002] Lee, K.-M. & Street, W. N. (2002). Incremental feature weight learning and its application to a shape-based query system. *Pattern Recognition Letters*, 23(7), 865 – 874.
- [Lee, 1999] Lee, S.-W. (1999). *Advances in Handwriting Recognition*. World Scientific Publishing Company.
- [Leedham & Chachra, 2003] Leedham, G. & Chachra, S. (2003). Writer identification using innovative binarised features of handwritten numerals. In *ICDAR '03: Proceedings of the Seventh International Conference on Document Analysis and Recognition* (pp. 413).
- [Leedham et al., 2003] Leedham, G., Yan, C., Takru, K., Tan, J. H. N., & Mian, L. (2003). Comparison of some thresholding algorithms for text/background segmentation in difficult document images. In *ICDAR '03: Proceedings of the Seventh International Conference on Document Analysis and Recognition* (pp. 859).

- [Legault & Suen, 1992] Legault, R. & Suen, C. (1992). A comparison of methods of extracting curvature features. In *Proceedings of the 11th IAPR International Conference on Pattern Recognition*.
- [Maarse & Thomassen, 1983] Maarse, F. J. & Thomassen, A. J. W. M. (1983). Produced and perceived writing slant: differences between up and down strokes. *Acta Psychologica*, 54(1-3), 131–147.
- [Madhvanath & Govindaraju, 1997] Madhvanath, S. & Govindaraju, V. (1997). Contour-based image preprocessing for holistic handwritten word recognition. In *ICDAR '97: Proceedings of the 4th International Conference on Document Analysis and Recognition* (pp. 536–539).
- [Mandelbrot, 1975] Mandelbrot, B. (1975). *Les objets fractals*. Flammarion.
- [Margner & Abed, 2009] Margner, V. & Abed, H. E. (2009). Icdar 2009 arabic handwriting recognition competition. In *ICDAR '09: Proceedings of the 10th International Conference on Document Analysis and Recognition* (pp. 1383–1387).
- [Margner & Abed, 2007] Margner, V. & Abed, H. E. (2007). Arabic handwriting recognition competition. In *ICDAR '07: Proceedings of the Ninth International Conference on Document Analysis and Recognition* (pp. 1274–1278).
- [Margner et al., 2005] Margner, V., Pechwitz, M., & Abed, H. (2005). Icdar 2005 arabic handwriting recognition competition. In *ICDAR '05: Proceedings of the Eighth International Conference on Document Analysis and Recognition* (pp. 70–74).
- [Marti & Bunke, 2002] Marti, U.-V. & Bunke, H. (2002). The iam-database: an english sentence database for offline handwriting recognition. *International Journal on Document Analysis and Recognition*, 5, 1433–2825.
- [Marti et al., 2001] Marti, U.-V., Messerli, R., & Bunke, H. (2001). Writer identification using text line based features. In *ICDAR '01: Proceedings of the Sixth International Conference on Document Analysis and Recognition* (pp. 101–105). Washington, DC, USA: IEEE Computer Society.
- [Mergl et al., 2004] Mergl, R., Juckel, G., Rihl, J., Henkel, V., Karner, M., Tigges, P., Schröter, A., & Hegerl, U. (2004). Kinematical analysis of handwriting movements in depressed patients. *Acta Psychiatrica Scandinavica*, 109(5), 383–391.
- [Miura et al., 1997] Miura, K. T., Sato, R., & Mori, S. (1997). A method of extracting curvature features and its application to handwritten character recognition. In *ICDAR '97: Proceedings of the 4th International Conference on Document Analysis and Recognition* (pp. 450–454).

- [Moalla et al., 2006] Moalla, I., LeBourgeois, F., Emptoz, H., , & Alimi, A. M. (2006). Contribution to the discrimination of the medieval manuscript texts : Application in the palaeography. In *DAS'06: In Proceedings of the 7th international workshop on Document Analysis Systems* (pp. 25–37).
- [Nadler & Smith, 1993] Nadler, M. & Smith, E. P. (1993). *Pattern Recognition Engineering*. Wiley-Interscience.
- [Naske, 1982] Naske, R. (1982). Writer recognition by prototype related deformation of handprinted characters. In *ICPR'82:In Proceedings of the 6th International Conference on Pattern Recognition* (pp. 819–822).
- [Niblack et al., 1993] Niblack, C. W., Barber, R., Equitz, W., Flickner, M. D., Glasman, E. H., Petkovic, D., Yanker, P., Faloutsos, C., & Taubin, G. (1993). The qbic project: querying images by content, using color, texture, and shape. In *Proc. SPIE Conf. on Storage and Retrieval for Image and Video Databases V*, volume 1908 (pp. 173–187).
- [Niblack, 1986] Niblack, W. (1986). *An Introduction to Digital Image Processing*. Prentice-Hall, Inc.
- [Nickell, 2007] Nickell, J. (2007). *Detecting Forgery Forensic Investigation of Documents*. University Press of Kentucky.
- [Nickell & Fischer, 1999] Nickell, P. J. & Fischer, J. F. (1999). *Crime Science: Methods of Forensic Detection*. The University Press of Kentucky.
- [Ünlü et al., 2006] Ünlü, A., Brause, R., & Krakow2, K. (2006). Handwriting analysis for diagnosis and prognosis of parkinson's disease. In *Proc. of the Int. Symp. Biological and Medical Data Analysis*, volume LNCS Vol 4345 (pp. 441–450).
- [Nosary et al., 1998] Nosary, A., Heutte, L., Paquet, T., & Lecourtier, Y. (1998). A step towards the use of writer's properties for text recognition.
- [Nosary et al., 1999] Nosary, A., Heutte, L., Paquet, T., & Lecourtier, Y. (1999). Defining writer's invariants to adapt the recognition task. In *ICDAR '99: Proceedings of the Fifth International Conference on Document Analysis and Recognition* (pp. 765–768). Washington, DC, USA: IEEE Computer Society.
- [Nosary et al., 2002] Nosary, A., Paquet, T., Heutte, L., & Bensefia, A. (2002). Handwritten text recognition through writer adaptation. In *IWFHR '02: Proceedings of the Eighth International Workshop on Frontiers in Handwriting Recognition*.
- [Olyanova, 1960] Olyanova, N. (1960). *The Psychology of Handwriting*. Sterling Pub Co Inc, NY.

- [Otsu, 1979] Otsu, N. (1979). A threshold selection method from grey-level histograms. *IEEE Trans. Syst., Man., Cybern.*, SMC-9, 62–66.
- [Päivinen, 2005] Päivinen, N. (2005). Clustering with a minimum spanning tree of scale-free-like structure. *Pattern Recogn. Lett.*, 26(7), 921–930.
- [Pareti & Vincent, 2006] Pareti, R. & Vincent, N. (2006). Global method based on pattern occurrences for writer identification. In *IWFHR'06: Proceedings of the 10th International Workshop on Frontiers in Handwriting Recognition*.
- [Peake & Tan, 1997] Peake, G. S. & Tan, T. N. (1997). Script and language identification from document images. In *BMVC'97: In Proceedings of British Machine Vision Conference*, volume 2 (pp. 610–619).
- [Pervouchine & Leedham, 2006] Pervouchine, V. & Leedham, G. (2006). Extraction and analysis of document examiner features from vector skeletons of grapheme 'th'. In *DAS'06: In Proceedings of the 7th international workshop on Document analysis systems* (pp. 197–207).
- [Pervouchine & Leedham, 2007] Pervouchine, V. & Leedham, G. (2007). Extraction and analysis of forensic document examiner features used for writer identification. *Pattern Recogn.*, 40(3), 1004–1013.
- [Peura & Iivarinen, 1997] Peura, M. & Iivarinen, J. (1997). Efficiency of simple shape descriptors. In *In Aspects of Visual Form* (pp. 443–451).
- [Plamondon & Lorette, 1989] Plamondon, R. & Lorette, G. (1989). Automatic signature verification and writer identification – the state of the art. *Pattern Recognition*, 22(2), 107–131.
- [Plamondon & Srihari, 2000] Plamondon, R. & Srihari, S. N. (2000). On-line and off-line handwriting recognition: A comprehensive survey. *IEEE Trans. Pattern Anal. Mach. Intell.*, 22(1), 63–84.
- [Prim, 1957] Prim, R. (1957). Shortest connection networks and some generalizations. *Bell Systems Technical Journal*, (pp. 1389–1401).
- [Racine et al., 2008] Racine, M. B., Majnemer, A., Shevell, M., & Snider, L. (2008). Handwriting performance in children with attention deficit hyperactivity disorder (adhd). *Journal of Child Neurology*, 23(4), 399–406.
- [Rath & Manmatha, 2007] Rath, T. M. & Manmatha, R. (2007). Word spotting for historical documents. *International Journal on Document Analysis and Recognition*, 9, 139–152.
- [Rui et al., 1997] Rui, Y., Huang, T. S., & Mehrotra, S. (1997). Content-based image retrieval with relevance feedback in mars. In *ICIP'97: In Proceedings of the IEEE International Conference on Image Processing* (pp. 815–818).

- [Rui et al., 1998] Rui, Y., Huang, T. S., Ortega, M., & Mehrotra, S. (1998). Relevance feedback: A power tool for interactive content-based image retrieval. *IEEE Transactions on Circuits and Systems for Video Technology*, 8(5), 644 – 655.
- [Sage & Duvoisin, 2001] Sage, J. I. & Duvoisin, R. (2001). *Parkinson's Disease: A Guide for Patient and Family*. Lippincott Williams and Wilkins.
- [Said et al., 2000] Said, H. E. S., Tan, T. N., & Baker, K. D. (2000). Personal identification based on handwriting. *Pattern Recognition*, 33, 149–160.
- [Sarfraz & Ridha, 2007] Sarfraz, M. & Ridha, A. (2007). Content-based image retrieval using multiple shape descriptors. In *IEEE/ACS International Conference on Computer Systems and Applications* (pp. 730 – 737).
- [Sauvola et al., 1997] Sauvola, J. J., Seppänen, T., Haapakoski, S., & Pietikäinen, M. (1997). Adaptive document binarization. In *ICDAR '97: Proceedings of the 4th International Conference on Document Analysis and Recognition* (pp. 147–152).
- [Schlapbach, 2007] Schlapbach, A. (2007). *Writer Identification and Verification*.
- [Schlapbach & Bunke, 2004] Schlapbach, A. & Bunke, H. (2004). Using hmm based recognizers for writer identification and verification. In *IWFHR '04: Proceedings of the Ninth International Workshop on Frontiers in Handwriting Recognition* (pp. 167–172). Washington, DC, USA: IEEE Computer Society.
- [Schlapbach & Bunke, 2006a] Schlapbach, A. & Bunke, H. (2006a). Off-line writer identification using gaussian mixture models. In *ICPR '06: Proceedings of the 18th International Conference on Pattern Recognition* (pp. 992–995). Washington, DC, USA: IEEE Computer Society.
- [Schlapbach & Bunke, 2006b] Schlapbach, A. & Bunke, H. (2006b). Off-line writer verification: A comparison of a hidden markov model (hmm) and a gaussian mixture model (gmm) based system.
- [Schomaker & Bulacu, 2004] Schomaker, L. & Bulacu, M. (2004). Automatic writer identification using connected-component contours and edge-based features of uppercase western script. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(6), 787–798.
- [Schomaker et al., 2004] Schomaker, L., Bulacu, M., & Franke, K. (2004). Automatic writer identification using fragmented connected-component contours.
- [Schomaker, 1998] Schomaker, L. S. (1998). From handwriting analysis to pen-computer applications. *IEE Electronics Communication Engineering Journal*, 10, 93–102.

- [Seropian, 2003] Seropian, A. (2003). *Analyse de Document et Identification de Scripteurs*. PhD thesis, University Toulon.
- [Seropian et al., 2003] Seropian, A., Grimaldi, M., & Vincent, N. (2003). Writer identification based on the fractal construction of a reference base. In *Seventh International Conference on Document Analysis and Recognition, ICDAR 2003*.
- [Seropian & Vincent, 2002] Seropian, A. & Vincent, N. (2002). Writers authentication and fractal compression. In *IWFHR'02: In Proceedings of the 8th International Workshop on Frontiers in Handwriting Recognition* (pp. 434–439).
- [Siddiqi & Vincent, 2007] Siddiqi, I. & Vincent, N. (2007). Writer identification in handwritten documents. In *ICDAR '07: Proceedings of the Ninth International Conference on Document Analysis and Recognition (ICDAR 2007)*, volume 1 (pp. 108–112).: IEEE Computer Society.
- [Siddiqi & Vincent, 2008] Siddiqi, I. & Vincent, N. (2008). Combining global and local features for writer identification. In *ICFHR '08: Proceedings of the Eleventh International Conference on Frontiers in Handwriting Recognition*.
- [Siedlecki & Sklansky, 1989] Siedlecki, W. & Sklansky, J. (1989). A note on genetic algorithms for large-scale feature selection. *Pattern Recognition Letters*, 10(5), 335–347.
- [Srihari et al., 2002] Srihari, S. N., Cha, S.-H., Arora, H., & Lee, S. (2002). Individuality of handwriting. *Journal of Forensic Sciences*, 47(4).
- [Srihari et al., 2003] Srihari, S. N., Tomai, C. I., Zhang, B., & Lee, S. (2003). Individuality of numerals. In *ICDAR '03: Proceedings of the Seventh International Conference on Document Analysis and Recognition* (pp. 1096–1100).
- [Suen et al., 1992] Suen, C., Nadal, C., Legault, R., Mai, T., & Lam, L. (1992). Computer recognition of unconstrained handwritten numerals. *Special Issue of Proc IEEE*, 80(7), 1162–1180.
- [Sutanto et al., 2003] Sutanto, P. J., Leedham, G., & Pervouchine, V. (2003). Study of the consistency of some discriminatory features used by document examiners in the analysis of handwritten letter a. In *ICDAR '03: Proceedings of the Seventh International Conference on Document Analysis and Recognition* (pp. 1091).
- [Tan et al., 2009] Tan, G. X., Viard-Gaudin, C., & Kot, A. C. (2009). Impact of alphabet knowledge on online writer identification. In *ICDAR'09: In Proceedings of the 10th International Conference on Document Analysis and Recognition*.

- [Tan, 1996] Tan, T. (1996). Written language recognition based on texture analysis. In *ICIP'96: In Proceedings of International Conference on Image Processing*, volume 2 (pp. 185–188).
- [Tomai et al., 2004] Tomai, C. I., Zhang, B., & Srihari, S. N. (2004). Discriminatory power of handwritten words for writer recognition. In *ICPR '04: Proceedings of the 17th International Conference on Pattern Recognition* (pp. 638–641).
- [Viard-Gaudin et al., 1999] Viard-Gaudin, C., Lallican, P. M., Binter, P., & Knerr, S. (1999). The ireste on/off (ironoff) dual handwriting database. In *ICDAR '99: Proceedings of the Fifth International Conference on Document Analysis and Recognition* (pp. 455).
- [Vincent & Emptoz, 1995] Vincent, N. & Emptoz, H. (1995). A classification of writings based on fractals. *Fractal Reviews in the Natural and Applied Sciences*, (pp. 320–331).
- [Vincent et al., 2005] Vincent, N., Seropian, A., & Stamon, G. (2005). Synthesis for handwriting analysis. *Pattern Recogn. Lett.*, 26(3), 267–275.
- [Vinciarelli, 2002] Vinciarelli, A. (2002). A survey on offline cursive word recognition. *Pattern Recognition*, 35, 1433–1446.
- [Wall & Danielsson, 1984] Wall, K. & Danielsson, P.-E. (1984). A fast sequential method for polygonal approximation of digitized curves. *Computer Vision, Graphics, and Image Processing*, 28(3), 220–227.
- [Wilkinson et al., 1992] Wilkinson, R., Geist, J., Janet, S., Grother, P., C.J.C.Burges, Creecy, R., Hammond, B., Hull, J., Larsen, N., Vogl, T., & Wilson, C. (1992). The first census optical character recognition systems conference. In *NISTIR 4912, National Institute of Standards and Technology, Gaithersburg, Md., USA*.
- [Wirocius, 2005] Wirocius, M. (2005). *Authetification par signature manuscrite sur support nomade*. PhD thesis, Université de Tours.
- [Wolf & Jolion, 2003] Wolf, C. & Jolion, J.-M. (2003). Extraction and recognition of artificial text in multimedia documents. *Pattern Anal. Appl.*, 6(4), 309–326.
- [Xu & Uberbacher, 1997] Xu, Y. & Uberbacher, E. C. (1997). 2d image segmentation using minimum spanning trees. *Image and Vision Computing*, 15, 47–57.
- [Yang et al., 2005] Yang, L., Suen, C. Y., Bui, T. D., & Zhang, P. (2005). Discrimination of similar handwritten numerals based on invariant curvature features. *Pattern recognition*, 38(7), 947–963.
- [Yosef et al., 2004] Yosef, I. B., Kedem, K., Dinstein, I., Beit-Arie, M., & Engel, E. (2004). Classification of hebrew calligraphic handwriting styles: Preliminary results. In *DIAL '04*:

Proceedings of the First International Workshop on Document Image Analysis for Libraries (pp. 299).

[Zahn, 1971] Zahn, C. (1971). Graph-theoretical methods for detecting and describing gestalt clusters. *IEEE Transactions on Computers*, C-20, 68–86.

[Zhang & Srihari, 2003] Zhang, B. & Srihari, S. N. (2003). Analysis of handwriting individuality using word features. In 1142-1146 (Ed.), *ICDAR '03: Proceedings of the Seventh International Conference on Document Analysis and Recognition*.

[Zhang et al., 2003] Zhang, B., Srihari, S. N., & Lee, S. (2003). Individuality of handwritten characters. In *ICDAR '03: Proceedings of the Seventh International Conference on Document Analysis and Recognition* (pp. 1086–1090). Washington, DC, USA: IEEE Computer Society.

[Zipf, 1949] Zipf, G. (1949). *Human Behavior and the Principle of Least Effort*. Addison-Wesley.

[Zois & Anastassopoulos, 2000] Zois, E. N. & Anastassopoulos, V. (2000). Morphological waveform coding for writer identification. *Pattern Recognition*, 33(3), 385–398.

[Zramdini & Ingold, 1993] Zramdini, A. W. & Ingold, R. (1993). Optical font recognition from projection profiles. *Electronic Publishing*, 6, 249–260.

Author's Publications

2009 **Siddiqi, I.**, Cloppet, F. and Vincent, N.: “Contour Based Features for the Classification of Ancient Manuscripts”, In *IGS'09: Proceedings of the 14th Conference of the International Graphonomics Society*, 13th – 16th September 2009, Dijon, France.

Siddiqi, I. and Vincent, N.: “Combining Contour Based Orientation and Curvature Features for Writer Recognition”. In *CAIP'09: Proceedings of the Thirteenth International Conference on Computer Analysis of Images and Patterns*, 2nd – 4th September 2009, Münster, Germany.

Siddiqi, I. and Vincent, N.: “A Set of Chain Code Based Features for Writer Recognition”. In *ICDAR '09: Proceedings of the Tenth International Conference on Document Analysis and Recognition*, 26th – 29th July 2009, Barcelona, Spain.

Khurshid, K., **Siddiqi, I.**, Faure, C. and Vincent, N.: “Comparison of Niblack Inspired Binarization Methods for Ancient Documents”, In *DRR'09: Proceedings of the 16th Document Recognition and Retrieval Conference*, 21st – 22nd January 2009, San Jose, CA, USA.

2008 **Siddiqi, I.** and Vincent, N.: « Descripteurs Locaux de Forme pour la Reconnaissance de Scripteur », In *CIFED'08: X^e Colloque International Francophone sur l'Écrit et le Document*, 28th – 31st October 2008, Rouen, France.

Siddiqi, I. and Vincent, N.: “Stroke Width Independent Feature for Writer Identification and Handwriting Classification”. In *ICFHR '08: Proceedings of the Eleventh International Conference on Frontiers in Handwriting Recognition*, 19th – 21st August 2008, Montreal, Canada.

Siddiqi, I. and Vincent, N.: “Combining Global and Local Features for Writer Identification”. In *ICFHR '08: Proceedings of the Eleventh International Conference on Frontiers in Handwriting Recognition*, 19th – 21st August 2008, Montreal, Canada.

Siddiqi, I. and Vincent, N.: “How to Define Local Shape Descriptors for Writer Identification and Verification”. In *PRIS'08: 8th Int'l workshop on Pattern Recognition in Information Systems*, 12th – 13th June 2008, Barcelona, Spain.

2007

Siddiqi, I. and Vincent, N.: “Writer Identification in Handwritten Documents”. In *ICDAR '07: Proceedings of the Ninth International Conference on Document Analysis and Recognition*, 23rd – 26th September 2007, Curitiba, Brazil.