CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

1-NN	Nearest neighbor classifier
ASCII	American standard code for information interchange
AWD	Arabic word descriptor
BOW	Bag-of-words model
CC	Connected component
CTC	Connectionist temporal classification
DAG	Directed acyclic graph
ÉTS	École de technologie supérieure
IFN/ENIT	Institut für Nachrichtentechnik (IfN) / Ecole Nationale d'Ingénieurs de Tunis (ENIT)
HMM	Hidden Markov model
LR	Lexicon reduction
LSTM	Long short-term memory
MATLAB	Matrix laboratory
NN	Neural network
RNN	Recurrent neural network
SD	Structural descriptor
SIFT	Scale-invariant feature transform
SOM	Self-organizing map
SRV	Square root velocity representation

TSV	Topological signature vector
WRS	Word recognition system
W-TSV	Weighted topological signature vector

INTRODUCTION

0.1 Handwriting recognition for paper-based documents

Since its invention, paper has been used by man as a medium to convey information or messages. Since antiquity, messengers have carried paper-based messages over long distances to deliver the message. Then, with the progress of printing, billboards were introduced in cities, gradually replacing town criers who were in charge of making public announcements in the streets. Nowadays paper-based documents are ubiquitous in our society. In personal life, we receive mails, advertisement and news on paper. In companies, paper is the most prominent way to share and transfer information, for example through mails or forms. The processing of all the information requires many resources. In order to improve efficiency and reduce cost, handwriting recognition systems have been designed for specific applications such as postal address reading on mails and legal amount (amount written in words) and courtesy amount (amount written in numeral) reading on bank checks.

A second purpose of paper-based document has been to archive information. For a long time, the easiest way to save a piece of information has been to write it on paper. Libraries contain a large amount of knowledge in the form of books that people can consult. Most of the historical books are handwritten, while modern books are printed. In order to provide large access to their collection, libraries use powerful cataloging tools. Each document is described by a metadata file, which contains a set of fields as a global description of the document. However, most of the ancient documents are not indexed, so there is no metadata file and they are not reachable through search. Ideally, each document should be carefully annotated in order to allow an efficient browsing. Since manual annotation is very costly due to the length of the task and the large amount of documents to be processed, handwriting recognition systems are needed in an archiving context to provide access to historical documents.

Nowadays, with the popularization of computers, we observe a switch from paper-based documents to electronic documents. As electronic documents are *born-digital*, they have the obvious advantage to contain textual information in digital format, which can be easily processed

by computers. Paper is still widely used due to its convenience and our society's past habits. Regardless of this, the importance of handwriting recognition systems will decrease in many traditional areas, due to electronic payments or electronic forms. One promising area is the document *dematerialization*, where the paper-based document will be transformed into a complete, digital document, in which the textual content is embedded similarly to born-digital documents. This implies a complete analysis of the textual content of the document, and again, handwriting recognition is at the heart of the process. Document materialization is particularly suited for ancient documents, because their physical support suffers from aging, unlike digital documents, and because searching by keyword would render the document available.

In this thesis, we focus on the task of handwriting recognition. It is at the heart of all the applications aforementioned, namely automated document processing, automated document indexing and document dematerialization. Handwriting recognition systems involve several steps (Figure 0.1). First, the paper document is scanned, after which a copy of the document is obtained as a digital image. Then, a layout analysis is performed in order to identify the text portions of the document. Finally, the handwriting recognition engine is applied on the text portions and the recognized text is obtained as a sequence of characters.

0.2 Document scripts

We focus in this thesis on Latin and Arabic script based documents, because both scripts possess a large amount of historical books that would benefit from dematerialization, but also because academic research is particularly active for these two scripts. We briefly introduce here the Arabic script. Unlike Latin script, it is written from right to left, and the alphabet is composed of 28 letters instead of 26 (Figure 0.2). The shape of the letters is dependent on their position in the word, and is usually different if they are at the beginning, middle, or end of a word. Six letters ('ﺩ', 'D', 'D̲', 'R', 'Z', and 'W') can be connected only if they appear in a final position; if they appear in initial or medial position, a space is inserted after them and the word is broken into *subwords*. Several letters share the same base shape and are only distinguishable by diacritics in the form of one, two, or three dots appearing above or below the shape. The features of Arabic words are illustrated in Figure 0.3.

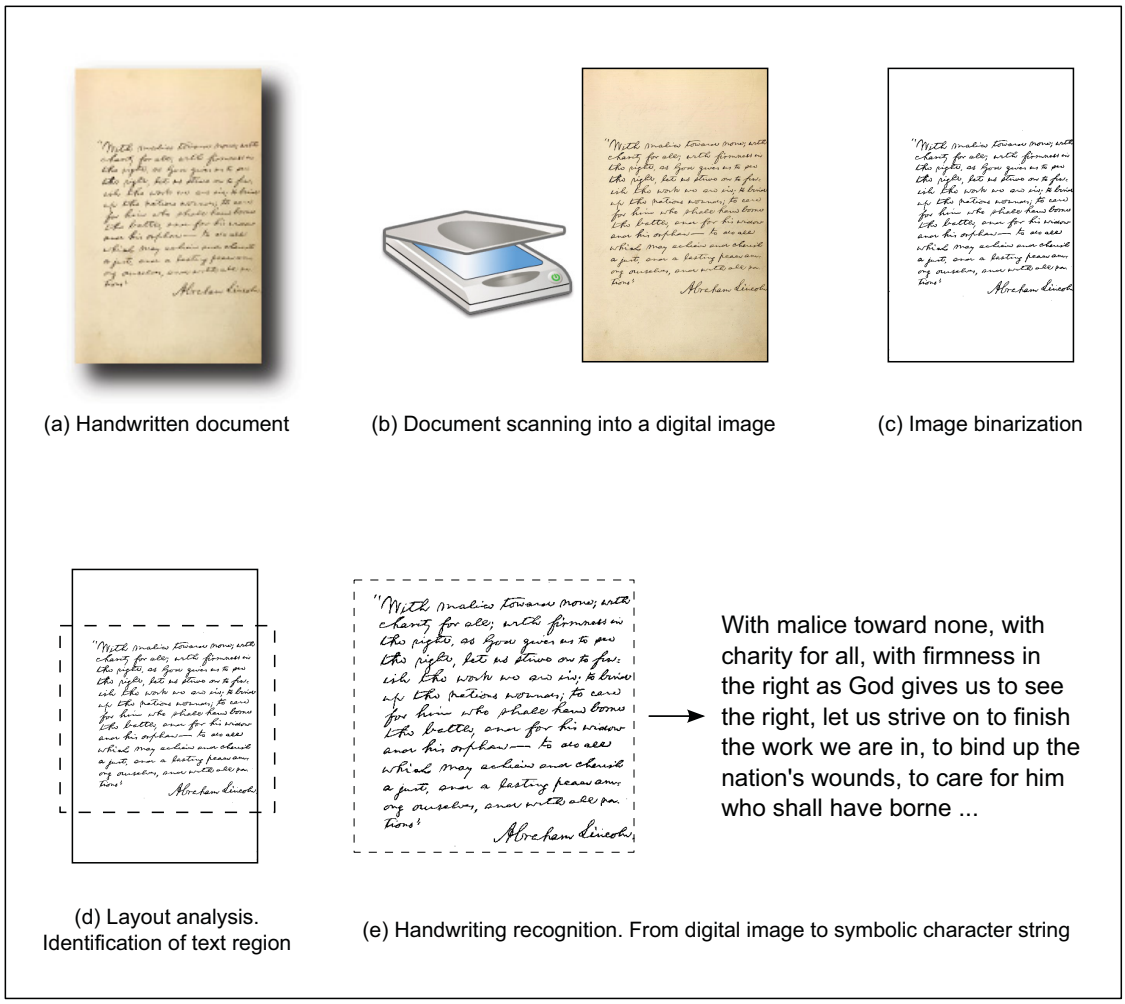


Figure 0.1 Overview of the handwriting recognition process. From original document to recognized text.[1]

Automatic word recognition is a complex process, as it involves several aspects and challenges. One particular aspect of interest is the visual features. Visual features are extracted from the document image and fed to the recognition system. Such an approach provides a better performance than directly using the image raw pixels. Therefore, they constitute an important component of recognition systems and have a great impact of their performance. In order to build the best recognition system, it is necessary to use the best features. This opens up the main question of this thesis, **what are the relevant features for handwriting recognition sys-**

[1]Sample document image reproduced from http://www.loc.gov/loc/lcib/0903/detail/legacy04.html

tems? To motivate the importance of this question and for a better understanding of the overall recognition process, we detail in the next section some of the main challenges of handwriting recognition, with a focus on visual appearance.

ص	ش	س	ز	ر	ذ	د	خ	ح	ج	ث	ت	ب	ا
Ṣ	Š	S	Z	R	Ḏ	D	Ḫ	Ḥ	Ǧ	Ṯ	T	B	ʾ
ي	و	ه	ن	م	ل	ك	ق	ف	غ	ع	ظ	ط	ض
Y	W	H	N	M	L	K	Q	F	Ġ	ʿ	Ẓ	Ṭ	Ḍ

Figure 0.2 Arabic letters with their ISO 233 transliteration.

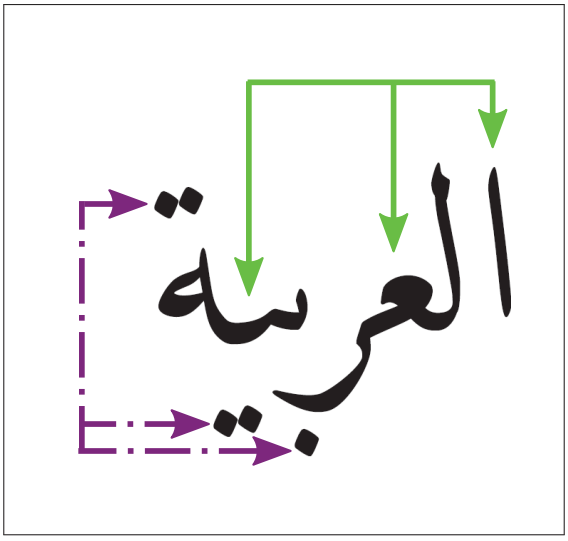


Figure 0.3 An Arabic word with its subwords (solid lines) and diacritics (dashed lines).

0.3 Problem statement

Automated handwriting recognition is a very challenging problem, which is yet to be solved. The difficulties are related to two distinct sources. The first is related to the visual appearance of the document text. It is influenced directly by the handwriting of the writer, as well as the paper

appearance. This class of problem makes handwriting recognition challenging for automated systems as well as for humans. The second is related to the simulation of the complex process occurring in our brains during handwriting recognition. This process is not yet well understood; therefore, it is very difficult to simulate it.

0.3.1 Text visual appearance

a. Handwriting variability: The handwritten word recognition problem is very challenging because of the high variability of the handwriting process. This problem can be summarized as follows, no two people have similar handwriting, and a single person cannot write twice exactly the same way. It means that the variability in the writing of a word is high between different individuals, but also non-negligible for a single individual. This variability has many origins. People may write using different scaling (small or big writing), or with slants; such variations can be categorized as affine transforms (Schomaker, 1998). Moreover, people may write using different cursive styles, where they use alternative shapes to write characters; this is also known as allographic variation (Schomaker, 1998). The variability also depends on the psychological state of the writer. Some illustrative cases are shown in Figure 0.4.

b. Document degradation: Documents suffer from various forms of degradation during their lifetime, most often due to age and bad maintenance. Over a period of time, the color of the paper changes, and the ink diffuses in the paper. Documents also suffer from physical damage, such as loss of page fragment, shears and stain. Physical degradation is particularly noticeable in ancient documents (Figure 0.5). Finally, during the digitalization of the document, some new degradation appears in the images, as the bleed-through effect (the verso of the document is visible) and deformations when a hardcover document page cannot properly be flattened.

c. Document background: Often writing papers or forms come with printed ruling lines to help the writer keep his baseline straight and to guide his writing into given portions of the page. It is common practice to remove such lines from the document image before the recognition, in order to prevent any interference with the text strokes. Certain types of documents, such as forms or bank checks, can be personalized with logos or various patterns or images as

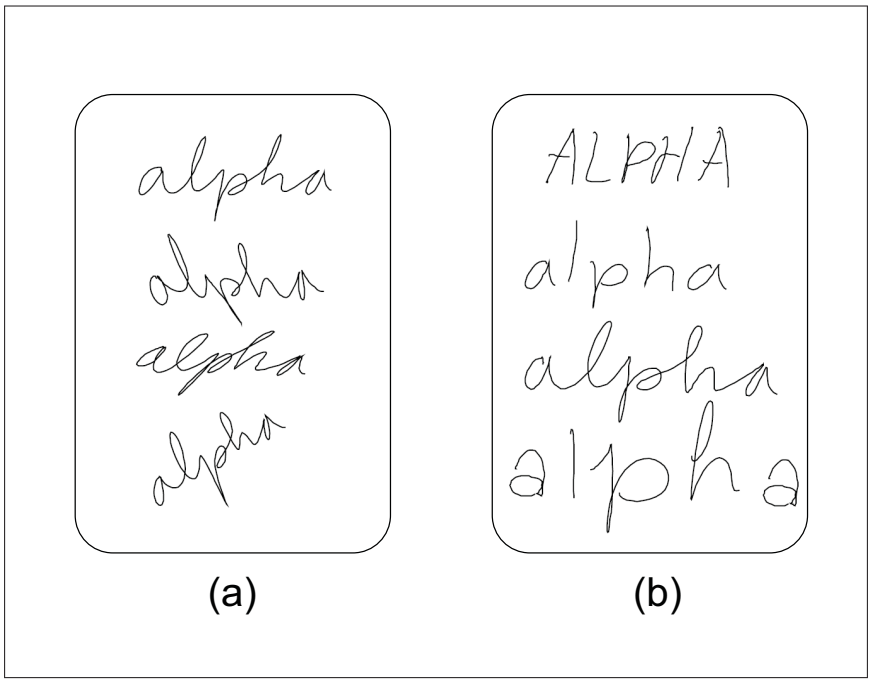


Figure 0.4 Source of variations in handwriting. (a) Affine transforms. (b) Allographic variation.
Images reproduced from Schomaker (1998), © 1998 IEEE.

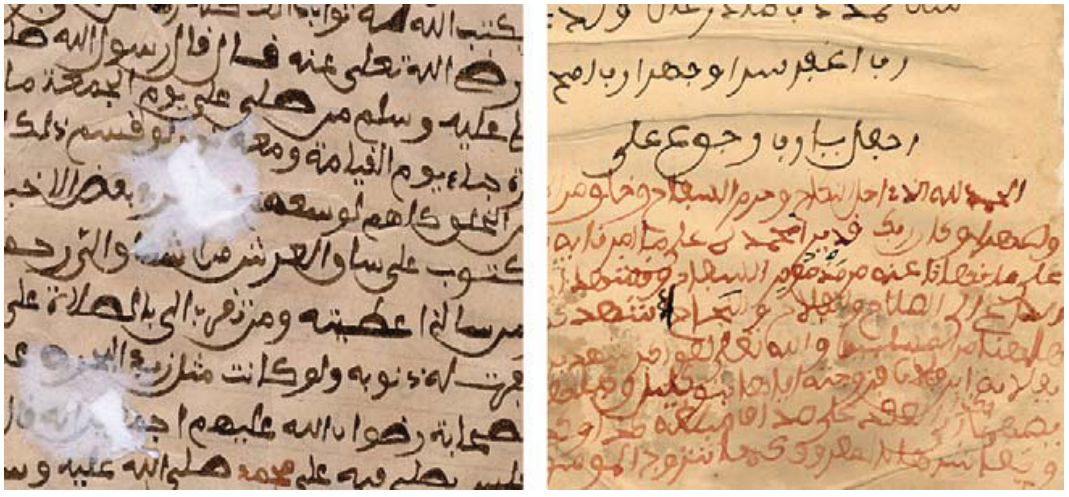


Figure 0.5 Historical documents with physical degradations.
Reprinted from Leydier *et al.* (2009), with permission from Elsevier.

background. In such cases, the background is more complex than simple vertical and horizontal lines and its removal requires specific algorithms. Examples of such documents are shown in Figure 0.6.

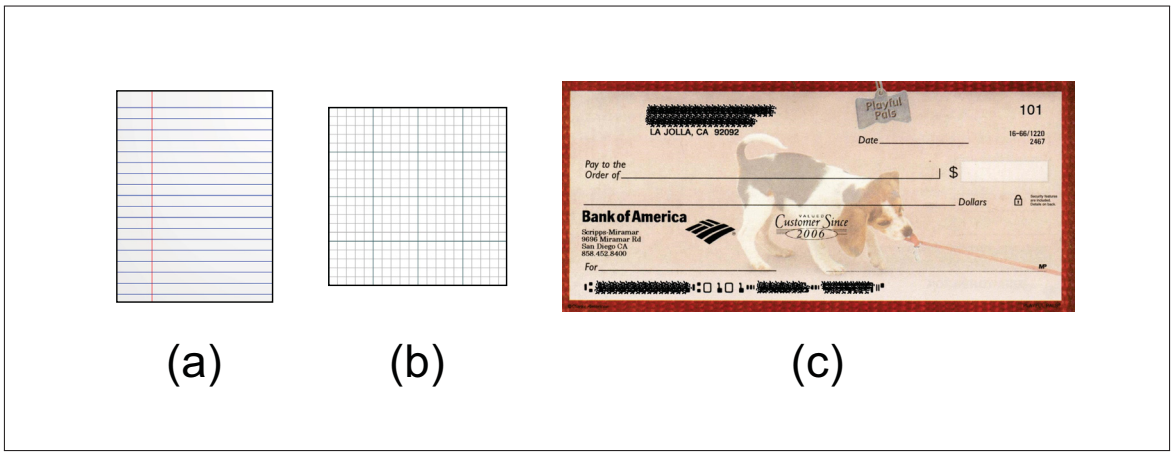


Figure 0.6 Documents with background. (a) Ruled paper. (b) Graph paper. (c) Check with complex background.[2]

0.3.2 Simulating the human reading process

a. Reading process: Words are spelled and written as a sequence of characters. Reading a written word therefore requires implicitly to recognize a sequence of characters. This is a very challenging task, specifically in cursive handwriting where the characters of a given word are connected to each other. To recognize a word, each character must be first isolated and recognized before recognizing the whole word. Nevertheless, accurate character isolation and recognition are possible only if the word is already known. This phenomenon is known as Sayre's paradox (Sayre, 1973). The task of reading therefore implies joint recognition at the word and character level. The set of word candidates considered during the recognition is called a *lexicon*. It is often context dependent, which makes the recognition easier. For example, if we are reading a medical prescription, we will be implicitly looking for medical words, and if we are reading a mail address, our lexicon will be made of street and city names. Finally, when the document to read is a full text, the language grammar, which includes the syntax and semantics,

[2]Check image reproduced from http://eaptips.wikidot.com/using-checks

also guides the reading process; for example it allows us to detect mistakes if any. Therefore, many levels of formal knowledge are involved for successful handwriting recognition, namely the alphabet, the lexicon, and the language grammar (Figure 0.7). However, the actual human reading process is not yet understood; that is we do not know exactly how these sources of knowledge are combined. Several reading models exist in the literature, but they are still debated and under progress (Côté *et al.*, 1998). Therefore, simulating the task of reading with a computer is very challenging.

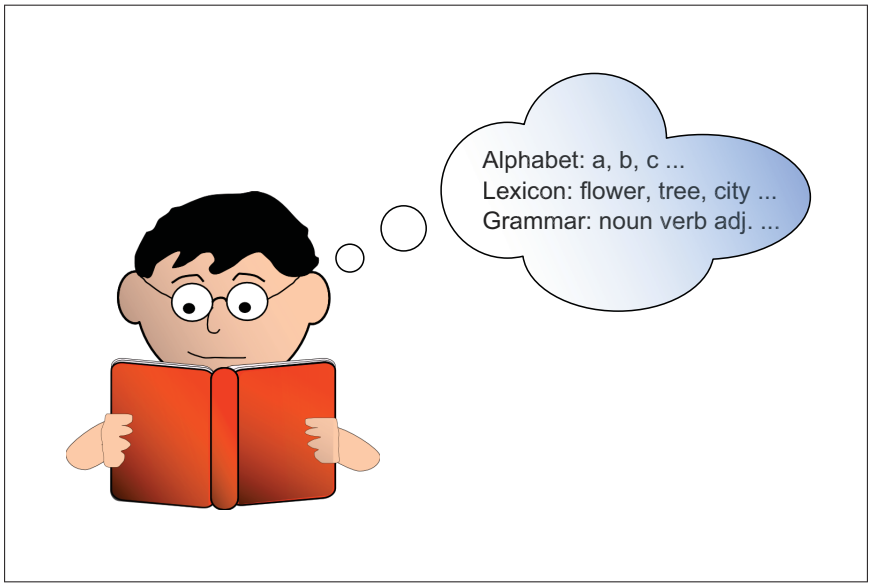


Figure 0.7 Formal knowledge involved during human reading.

b. Visual features for recognition: Visual features are extracted from the document image in order to improve the robustness to the handwriting variability such as geometrical deformation of the character shape and to binarization artifacts. They also provide a layer of abstraction from the image pixels toward the word symbolic character string. Nevertheless, the design of such features is difficult because of the lack of explicit knowledge. People are taught to recognize characters from a limited number of well-defined samples. Then, they generalize their knowledge to real world handwriting independently, based on their personal experience. This implicit knowledge must be explicitly defined as features. Features can be classified into low-level and high-level features, depending on their level of abstraction. Both have their

advantages and disadvantages. Low-level features are often based on local geometry and local gradient orientation, while high-level features are based on symbolic features, such as the presence of ascenders, descenders, dots or loop (Cheriet and Suen, 1993). Unlike low-level features, high-level features are easily explainable with characters, for example the presence of a descender can be related to the characters ‘g’, ‘p’, or ‘q’ among others. However, they are harder to extract than low-level features and therefore less robust. For example, a writer could leave the loop of the character ‘o’ open, leaving the application of the strict definition of a loop useless for the extraction purpose. Such cases are illustrated on sample handwritten digits from the MNIST database in Figure 0.8. Another issue with features is their evaluation. Because the definition of relevant features is based on implicit knowledge, it is not possible to directly compare features. Therefore, the relevance of features is evaluated indirectly, through the empirical performance of the recognition system. Finally, the choice of visual feature for handwriting recognition still remains an open question since consensus on which feature to use has not yet been reached. This is demonstrated by the large number of features proposed in the literature.

0.4 Contributions

Past research on handwriting recognition has established a framework for automatic recognition. A critical aspect is the description of the visual shape of handwritten word. As seen in the previous section, the design of efficient visual feature is challenging. A large body of features are available in the literature and yet no consensus is found. The search for better and more relevant features is still an active field of research. Therefore, **the purpose of this thesis is to improve our understanding of relevant features for word recognition systems (WRS)**. It will be done through the design of novel visual descriptors along with the evaluation of existing ones. Our research focuses on two complementary aspects of features in handwriting recognition.

First, the research will focus on the lexicon reduction (LR) problem, which quickly selects a set of candidate word hypotheses given a word image. One key aspect of LR is to efficiently extract and represent the word shape. Contribution will be made on this particular topic for

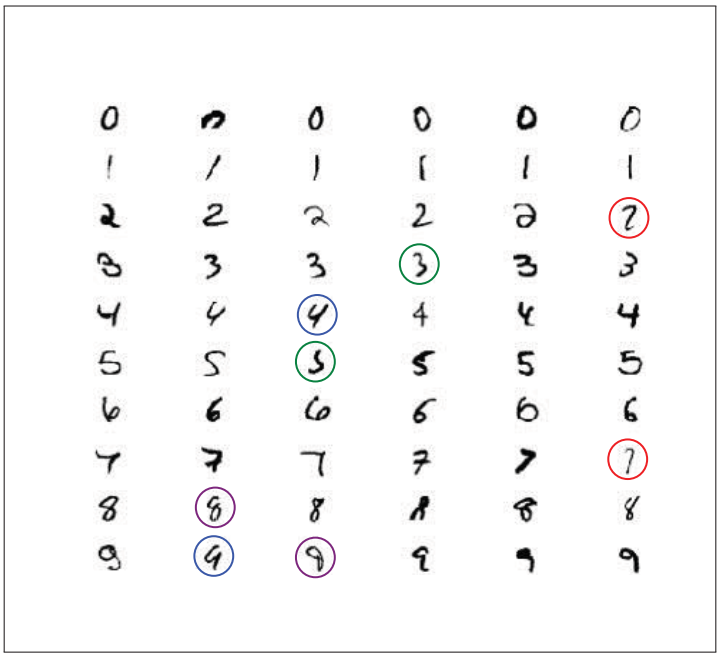


Figure 0.8 Variability in handwritten digits from the MNIST database. Similar digits from different classes are highlighted with same color. It demonstrates the difficulties of feature design.
Reprinted from Niu and Suen (2012), with permission from Elsevier.

Arabic script. Indeed, current methods focus only on symbolic information (diacritics and the number of subwords) and totally ignore the subwords shape. Two novel visual descriptors are proposed, based on the subword shapes and symbolic information. It will be shown in the experimental section that there is a significant improvement of the LR performance with a simple and efficient implementation.

Secondly, the research will focus on the actual WRS. A large body of features already exist, but no tool is available to compare them, except the recognition rate. Contribution will be made by proposing a framework for feature evaluation, which assesses the strength as well as the complementarity between features. It is done by assigning a score to each feature which has a simple interpretation. The results provide interesting insights on popular features of the literature.

0.5 Context of the thesis

This section details the scientific context of this thesis. Features for handwriting recognition systems emerge from several fields of computer science, and in particular computer vision. During our research, we were interested in features used for object matching, where a query object is matched against an object database, for the purpose of recognition or retrieval. This problem is very challenging because the appearance of a 3D object in a 2D image depends on its pose. Different viewpoints can modify an object scale and orientation in the image, but also its silhouette, for example through self-occlusion. To tackle this problem, several representations have been proposed (with ad hoc matching strategies), which are invariant to scale, rotation, and translation, but also tolerant to object deformations to handle the case of non-rigid object and occlusion.

A parallel can be drawn between objects and handwritten words. Indeed, words can be considered as entities, similarly to objects. Also, handwriting variability, which is one of the main challenges of word recognition, can be related to object deformation. Therefore, object representations are of great interest for handwriting recognition. However, differences exist. Given a document image, the scale and orientation are important cues to discriminate between words, and invariance to these criteria is not suited. For example, the scale helps to differentiate between words with few and many characters, while the document orientation allows to distinguish between the characters ‘d’ and ‘p’, which are just rotated versions of the same shape.

We focused our attention on a particular object matching framework based on shape analysis, called the shock graph (Siddiqi *et al.*, 1999). A skeletal graph, interpreted as a DAG, is formed based on a shape medial axis and its radius function. This shock graph is then used for shape matching, but the procedure is computationally expensive when matching against a large database of exemplar shapes, due to the high complexity of the matching algorithm. To alleviate this problem, a shape indexing strategy has been proposed. The DAG is modeled by a vector, based solely on its topology, called the topological signature vector, or TSV (Shok-

oufandeh *et al.*, 2005). The database is then dynamically reduced given a query shape, by selecting the most similar database entries based on the TSV index.

This powerful combination, of a rich graphical representation with an efficient indexing scheme have a great potential for Arabic subword lexicon reduction, and it has inspired the Chapter 3 of this thesis. The original application of this framework and that of this thesis are different (object recognition vs. word recognition), therefore, two main adaptations are needed. First, the shock graph model is not efficient for Arabic subword shapes because they have a fixed thickness defined by the strokes width, leading to a constant radius function. Therefore, a new weighted DAG model has been proposed, specifically tailored for the Arabic subword structure, and encoding the shape scale in its edges. Second, the indexing framework has been extended to consider the case of weighted DAG. This new signature vector, combined with the proposed weighted DAG model, is able to discriminate subword shapes based on topology, but also based on scale. A formal analysis of the robustness to perturbation of this new signature vector compared to the original one has been proposed. Also, a fast signature computation method has been proposed, solely based on the weights of the graph and ignoring its topology.

From this starting point, the lexicon reduction methodology has been extended to the whole Arabic word, using this time the bag-of-word model. This model has been extensively used in computer vision for object retrieval, generally associated with popular local image descriptors. Finally, we investigated the question of relevant features for the actual task of handwriting recognition, considering features proposed from various fields such as pattern recognition and machine learning.

0.6 Outline of the thesis

This introductory chapter gave the general context of the thesis. Chapter 1 reviews the literature on features for lexicon reduction and handwriting recognition. The limitations of the literature are highlighted. Chapter 2 introduces the general methodology, including the objectives of the thesis based on the limitations of the state of the art. The resulting general approach for lexicon reduction in Arabic script and feature evaluation is then described. The next three chapters

present the methods and results developed in this thesis. Chapter 3 presents the first journal article. A method for lexicon reduction in Arabic script using subword shape is developed. Chapter 4 presents the second journal article. A word shape descriptor is designed for lexicon reduction in Arabic script, incorporating subword shapes and symbolic information. Chapter 5 presents the third journal article. A framework for the evaluation of features for handwriting recognition system is developed. It has been applied on both Latin and Arabic scripts. Then, Chapter 6 provides the general discussion that highlights the strengths and weaknesses of the proposed methods. Finally, the general conclusion summarizes the work accomplished and presented in this thesis and provides our recommendations and perspectives.

CHAPTER 1

LITERATURE REVIEW

In this chapter we review the relevant literature related to features used in handwritten WRS. We first provide formal descriptions of statistical word recognition systems by detailing two state-of-the-art models. It will allow us to better understand the interaction of features with WRS. Features are indeed involved for lexicon reduction if any, and at the input of the WRS. Therefore, we also review state-of-the-art methods for lexicon reduction followed by a review of the most commonly used features for handwriting recognition. Finally, we discuss the limitations of the literature.

1.1 Statistical word recognition system

State-of-the-art word recognition systems are based on a statistical framework. They model the sequential behavior of the handwriting process. More precisely, the input image is decomposed into vertical frames, then fed sequentially to the WRS. Visual features are extracted from each frame (more detail in Section 1.3). Then, the WRS selects the most probable word from the given lexicon. An overview of WRS is shown in Figure 1.1. More formally, the goal of the WRS is to find the word $\hat{w}$ which maximize $P(w|\mathbf{O})$, i.e. the probability of the word w given the input sequence of features $\mathbf{O} = \{\mathbf{o}_1, \mathbf{o}_2, \ldots, \mathbf{o}_T\}$. This probability can be written in multiple forms using Bayes's theorem (Eq. 1.1). In the last line, $P(w)$ represents the prior probabilities of a given word w based on the language model. In this work, all words belonging to the lexicon are given the same probability because we are limiting ourselves to single word recognition. Optionally, a lexicon reduction module can be added, to dynamically select specific word hypotheses based on the query word image, in order to improve the recognition rate and/or the processing speed. In the following, we will detail the two most competitive models for handwriting recognition, namely the hidden Markov models (HMM), and the recurrent neural networks (RNN).

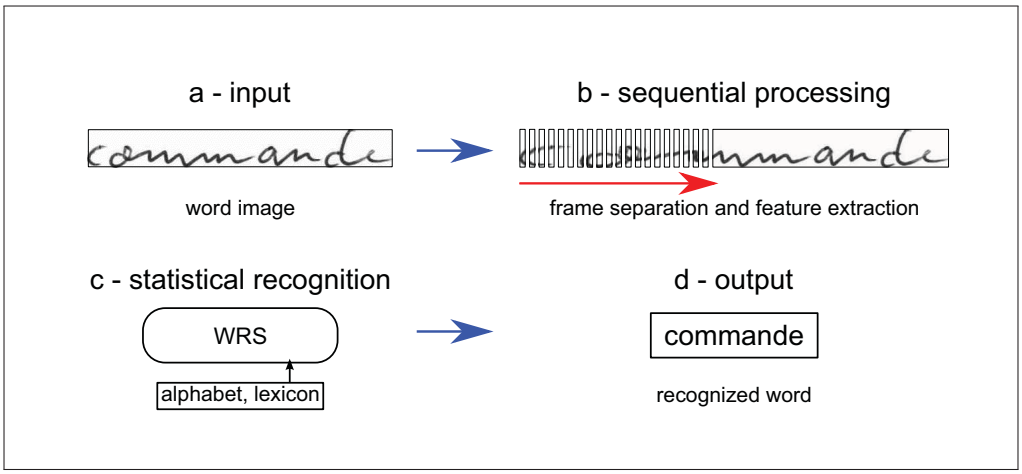


Figure 1.1 Overview of a statistical word recognition system.

$$\begin{aligned} \hat{w} &= \arg\max_{w} P\left(w|\mathbf{O}\right) \\ &= \arg\max_{w} \frac{P\left(w\right) P\left(\mathbf{O}|w\right)}{P\left(\mathbf{O}\right)} \\ &= \arg\max_{w} P\left(w\right) P\left(\mathbf{O}|w\right) \end{aligned} \tag{1.1}$$

1.1.1 Hidden Markov models

Hidden Markov model (HMM) is a statistical model used for sequential data (Fink, 2008). HMM has the ability to both replicate the generation process of the data and to segment it into some meaningful unit. They describe a double stochastic process. The first stage is discrete, a random variable models the state of a system through time and takes on values from a finite number of states. The probability of the future state only depends on its immediate predecessor, therefore, there is only a first order dependency. In the second stage, an emission is generated for every time step, whose probability distribution is dependent only on the current state. The model is named 'hidden' Markov model because the states are not directly observable. An overview is shown in Figure 1.2.

More formally, the parameters α of an HMM are defined by the following elements:

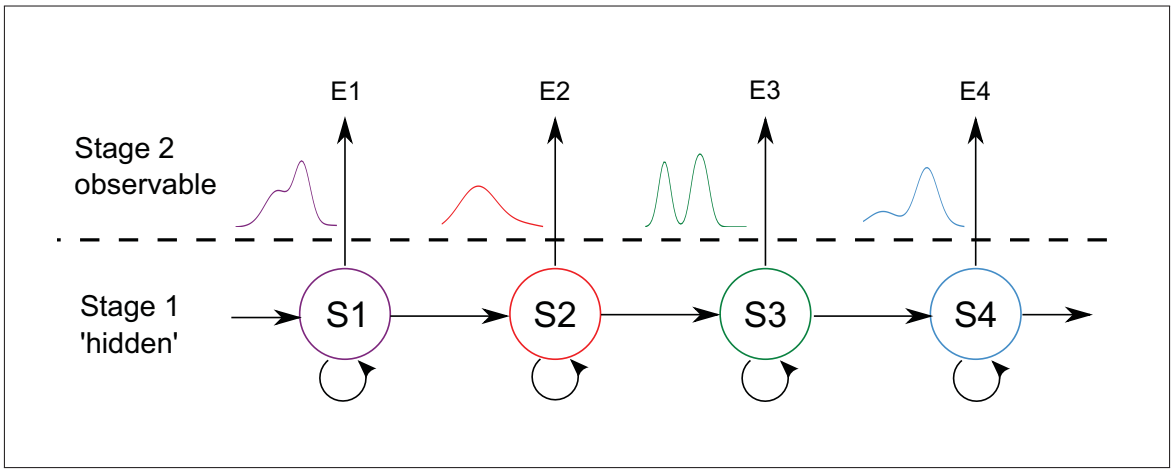


Figure 1.2 HMM with a linear topology. Stage 1: State transition to the next state or itself, not observable. Stage 2: Observable emissions with their state dependent probability distribution.

- The number K of states of the model : $\mathbf{S} = \{S_1, S_2, \ldots, S_K\}$
- The state transition probability matrix $\mathbf{A}$ of dimension $K \times K$, where a_{ij} is the transition probability from state S_i to state S_j.
- Probability of the observation $\mathbf{o} \in \mathbb{R}^{\mathbf{n}}$ knowing the state: $b_j(\mathbf{o}) = P(\mathbf{o}|q_t = S_j)$, also known as the emission probability. Because of the continuous nature of the probability density function, we obtain the *continuous* HMM. Instead, if $\mathbf{o}$ would take on values from a discrete set, we would obtain the *discrete* HMM.
- The initial state distribution π, where π_i is the initial probability of the system to be in state S_i

For the estimation of the HMM parameters, no algorithm able to find a global optimum for any criteria is known. Therefore, a particular case of the expectation-maximization (EM) technique is used to find a local optimum, namely the Baum-Welch algorithm (Baum *et al.*, 1970). This algorithm updates the parameters λ into $\hat{\lambda}$ such that the generation probability of the data given the model is improved (Eq. 1.2). The algorithm is iteratively repeated until a local optimum is found.

$$P\left(\mathbf{O}|\hat{\lambda}\right) \geq P\left(\mathbf{O}|\lambda\right) \tag{1.2}$$

Once the probability distribution parameters are estimated, the HMM can be used to solve the decoding problem, in which given an observation sequence $\mathbf{O} = \{\mathbf{o}_1, \mathbf{o}_2, \ldots, \mathbf{o}_T\}$, we want to retrieve the most probable state sequence $\mathbf{q} = \{q_1, q_2, \ldots, q_T\}$:

$$\mathbf{q}^* = \arg\max_{\mathbf{q}} P\left(\mathbf{O}, \mathbf{q}|\lambda\right) \tag{1.3}$$

The Viterbi algorithm is used for this purpose, based on the following recursion relation:

$$\begin{aligned} V_{1,k} &= P\left(\mathbf{o}_1|q_1 = S_k\right).\pi_k \\ V_{t,k} &= P\left(\mathbf{o}_t|q_t = S_k\right). \max_{i\in\{1..K\}} a_{ik}.V_{t-1,i} \end{aligned} \tag{1.4}$$

Where $V_{t,k}$ is the probability of the most probable state sequence responsible for the first t observations. The Viterbi path can be retrieved by saving back-pointers which remember which state i was used in the second equation.

HMMs are applied in two ways to handwriting recognition. The first approach is *holistic*, where an HMM model exists for each word of the lexicon. Due to the potentially large number of models and training data required, the holistic approach is limited to specific applications such as reading the amount on bank checks. The other approach is *analytic*, an HMM model is trained for each character of the alphabet, and compound word level HMMs are formed by concatenating character level HMMs. This approach is the most popular, as it solves the limitations of the holistic approach; HMM models for unseen words can be created, and sufficient data usually exists for the training of the character HMMs.

1.1.2 Recurrent neural networks

Recurrent neural network (RNN) is a class of neural network (NN) where connections between neurons form a directed cycle. This allows the model to keep a 'memory' of its previous state and therefore to make use of past context. This ability of the model is important for the task of handwriting recognition, where the context plays an important role. Also, as most of the neural networks, this model is discriminative, unlike standard HMMs which are generative. It therefore outperforms HMMs in many recognition applications. The current state-of-the-art method in most handwriting recognition task is based on the combination of long short-term memory (LTSM) layer (Gers *et al.*, 2003) and the connectionist temporal classification (CTC) layer (Graves *et al.*, 2009).

The LTSM layer is made of nodes with specific architecture called memory block, able to preserve contextual information over a long range of time. Each memory block contains a memory cell, and its interaction with the rest of the network is controlled by three multiplicative gates, namely: an input gate, an output gate and a forget gate. For example, if the input gate is closed, the block input has no influence on the memory cell. Similarly, the output gate has to be opened so the rest of the network can access the cell activation. The forget gate scales the recurrent connection of the cell. The gates behavior is controlled by the rest of the network (Figure 1.3).

For the specific task of handwriting recognition, the 'past' and 'future' context is necessary for better performance. Therefore, the bidirectional LSTM (BLSTM) layer is used, where one LSTM layer processes the sequence in the forward direction, while another layer processes it in the backward direction.

Then, the connectionist temporal classification (CTC) layer is plugged at the output of the BLSTM layer. The CTC layer has been designed for sequence labeling task. It is trained to predict the probability $P(w|\mathbf{O})$ of an output character sequence, i.e., a word w, given an input sequence $\mathbf{O}$, making the training discriminative. Its activation function provides the probability

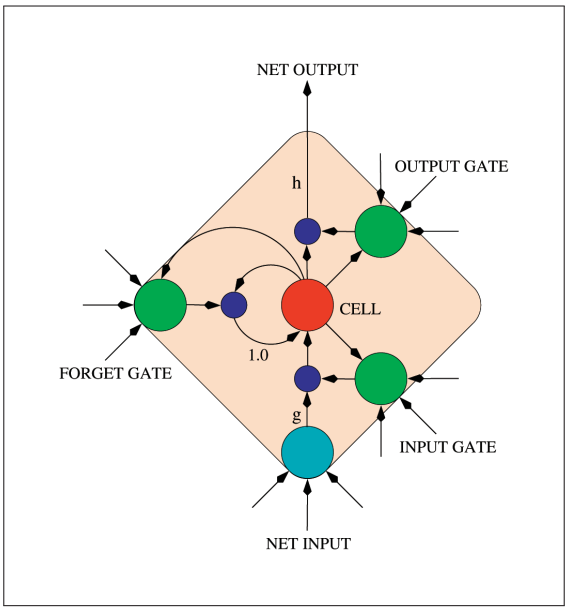


Figure 1.3 LSTM memory block, with the input, output and forget gates controlling its interaction with the rest of the network.
Figure reproduced from Graves *et al.* (2009), © 2009 IEEE.

to observe each character for each sequence time. One of the features of CTC is its ability to be trained with unsegmented data similarly to HMMs.

1.2 Features and strategies for lexicon reduction

Lexicon reduction is a high-level task, where word hypotheses are pruned from the lexicon. As it is used as a pre-processing step before the actual recognition, it must have a low computational overhead. Therefore, most of the methods rely on high-level features to take fast decisions. In the following, LR approaches are detailed for Latin and Arabic scripts, as well as for specific document types.

1.2.1 Latin script

Lexicon reduction can be performed by comparing the optical shapes of the lexicon words to improve recognition speed. When the word's optical shape is used, the simplest criterion for lexicon reduction, but still efficient, is word length, as this makes it easy to discriminate between long words and short words. More refined knowledge about the word's shape can

also be used. Zimmermann and Mao (1999) propose the concept of key characters, which are characters that can be accurately identified without a full contextual analysis. Character class specific geometrical properties are used, such as the average number of horizontal transitions, normalized vertical position and the normalized height. Lexicon reduction is performed by considering only the lexicon entries that match the regular expression generated by the key characters. They also estimate the letter count in a word using a neural network for further reduction. A similar approach is proposed by Palla *et al.* (2004), where regular expressions are built from the detection of ascenders and descenders in the query word image (Figure 1.4).

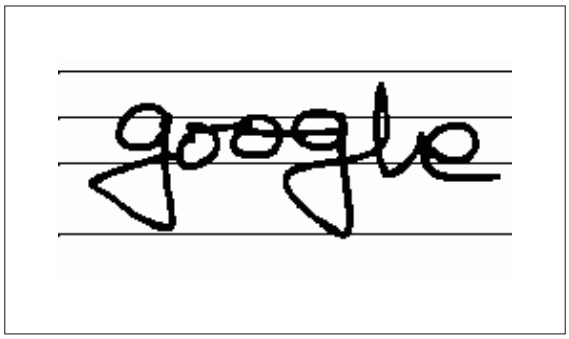


Figure 1.4 Ascenders and descenders features.
Image reproduced from Palla *et al.* (2004), © 2004 IEEE.

Bertolami *et al.* (2008) propose mapping each character of a word to a shape code. There are fewer shape codes than characters, as they only discriminate between characters based on their ascenders/descenders and basic geometry and topology. The mapping is performed by a hidden Markov model (HMM), which outputs the n best shape-code sequences for a query word. The lexicon is reduced by considering only the words that correspond to one of the shape-code sequences. Kaufmann *et al.* (1997) propose a holistic approach, using the quantified feature vectors extracted sequentially from the word image. These vectors are used by the HMM recognizer, so there is no overhead for the extraction of these features. A model is created for each class of the lexicon, and the word hypotheses are ranked according to the distance between their models and the features of the query word. Several other holistic approaches for lexicon reduction extract a string-based descriptor for each shape, which is further matched using dynamic programming, the lexicon entries with the smallest edit distances being consid-

ered part of the reduced lexicon. Madhvanath *et al.* (2001) holistic approach is based on using downward pen-strokes descriptors. These pen strokes are extracted from the word shape using a set of heuristic rules, and categorized according to their positions relative to the baseline. Then, lexicon reduction is performed by matching the word descriptors to the ideal descriptors extracted from the lexicon's ASCII string. Carbonnel and Anquetil (2004) compared two lexicon-reduction strategies, one based on lexicon indexing and the other on lexicon clustering. Using ascender/descender-based shape descriptors, the indexing approach showed better performance.

1.2.2 Arabic script

Arabic word shapes have a rich structure, due to their loops, branches, and diacritics (Lorigo and Govindaraju, 2006; Abuhaiba *et al.*, 1994; Zidouri, 2004). Mozaffari *et al.* (2008a) propose a two-stage reduction of an Arabic lexicon. In the first stage, the lexicon is reduced based on the number of subwords (defined in Section 0.2) of the query word (Figure 1.5). In the second stage, the word's diacritical mark types and positions are encoded into a string, and the lexicon is reduced based on the string edit distance. Mozaffari *et al.* (2008b) extended the previous approach to Farsi handwritten words, which contain more letters than the Arabic alphabet. Wshah *et al.* (2010) propose a similar algorithm, in which the diacritic detection stage is improved by the use of a convolutional neural network.

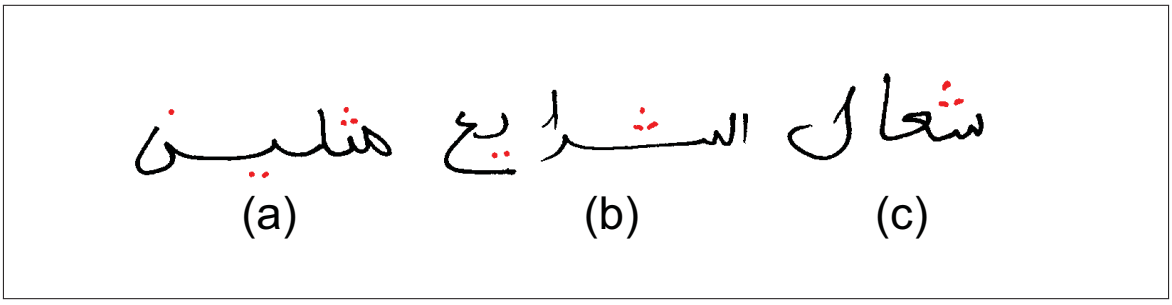


Figure 1.5 Arabic words with different number of subwords. (a) One subword. (b) Four subwords. (c) Two subwords.
Image reproduced from Wshah *et al.* (2010), © 2010 IEEE.

1.2.3 Specific document knowledge

Several lexicon-reduction approaches use application dependent knowledge to improve the system's recognition rate. For the transcript mapping problem with ancient document images, Tomai *et al.* (2002) propose recognizing each word of a document image by reducing the lexicon to specific lines of the transcript. Morita *et al.* (2002) have taken advantage of the date field structure for the recognition of handwritten dates on bank checks. Milewski and Govindaraju (2004) use an application-specific lexicon for word recognition on medical forms, while Farooq *et al.* (2009) have proposed estimating the topic of a query document from the output of a word recognizer. As the performance of a word recognizer is very low without a priori knowledge, Farooq *et al.* (2009) used the n best hypotheses for each word, instead of only the first, to infer the document topic. Once the document topic has been found, the query document is submitted again to the word recognizer, but this time with the topic-specific lexicon.

1.3 Features for sequential word recognition

In this section, we present the word image features used for recognition. The features are obtained by sliding a frame window horizontally over the word image and computing the features from each frame. They have been organized into four categories: distribution features, concavity features, visual-descriptor-based features and automatically learned features. The first three categories correspond to handcrafted features, and, when one of these features overlaps several categories, we assigned it to the most relevant one. Due to the large amount of proposed features in the literature, we limit our description to the most relevant features.

1.3.1 Distribution features

Distribution features characterize the density of foreground pixels within an image frame. They typically relate to the number of foreground pixels, the number of foreground/background position and to the lower and upper word shape profile. They capture the presence of ascenders and

descenders in the word image, which represents important cues for correct word recognition. Two distribution features are described here in detail.

The first feature has been proposed by Rath and Manmatha for handwritten word spotting in historical manuscript (Rath and Manmatha, 2003b). Each word image is described as a sequence of 4D feature vectors, namely upper and lower profile, projection and background to foreground transition (Figure 1.6). The minimum and maximum distance of the positions of foreground pixels are considered as lower and upper profile. Projection profile is the number of foreground pixels in the corresponding column. The number of transitions between foreground and background pixels is used as transition features. In word spotting application, the features extracted from two word images are matched using Dynamic Time Warping (DTW) for similarity. This feature is popular for its simplicity and robustness to image degradation.

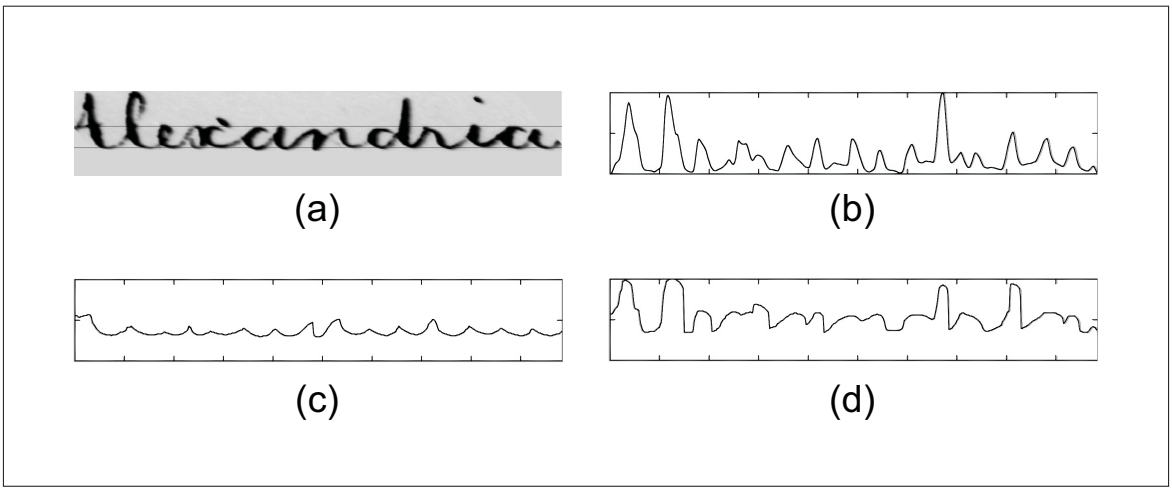


Figure 1.6 Distribution features. (a) Original image with upper and lower baseline displayed. (b) Projection profile (values inverted). (c) Lower profile. (d) Upper profile. Image reproduced from Rath and Manmatha (2003b), © 2003 IEEE.

The second feature has been proposed by Marti and Bunke (2001), and it has been used by many researchers for handwritten text recognition with HMM. Nine features are computed from the set of foreground pixels in each image column. Three global features capture the fraction of foreground pixels, the center of gravity, and the second order moment. The remaining six local features consist of the position of the upper and lower profile, the number

of foreground to background transitions, the fraction of foreground pixels between the upper and lower profiles and the gradient of the upper and lower profile with respect to the previous column, which provides dynamic information.

1.3.2 Concavity features

Concavity features relate to the word shape geometry. They provide stroke direction and concavity information (Al-Hajj Mohamad *et al.*, 2009). They are computed with a hit–or–miss transform given morphological patterns. Azeem and Ahmed (2012) proposed a set of concavity features (CCV feature) for Arabic word image. It has proved to be effective for Arabic text recognition using HMM, where 88.5% recognition accuracy has been reported without image pre-processing. First, the stroke thickness is normalized to 3 pixel width by a thinning operation followed by dilation. Then the response of the normalized image to 8 directional morphological filters is computed, leading to 8 binary directional images. Vertical frames of 6 pixels width are used to extract features, with an overlap of 3 pixels between two consecutive frames. Within each frame and for each directional image, the number of '1' pixels, as well as the normalized gravitational center of these pixels are extracted as feature. The final feature vector also includes dynamic information (delta and acceleration) and therefore contains 48 features per frame.

1.3.3 Visual-descriptor-based features

Visual-descriptor-based features are inspired by the advances in computer vision. Most descriptors are based on histograms of the image gradient. One of the most popular descriptors is SIFT (scale-invariant feature transform) (Lowe, 2004). It is computed at keypoints, which are defined as local extrema in locations and in the scale space. The scale of the descriptor is set to the scale space extrema. The area surrounding the keypoint is divided into 4×4 regions. An 8 bin orientation histogram is computed for each region, from the gradient orientation of the region sample points, weighted by the gradient magnitude. This leads to a 128D feature vector. Rotation invariance is built by aligning the descriptor orientation with the main orientation of the keypoint area.

One integration attempt of the SIFT descriptor has been proposed by Rothacker *et al.* (2012). The keypoints are detected using the Harris corner detector. Then, the SIFT descriptor is computed around keypoints using a fixed scale (determined experimentally) and without orientation alignment (Figure 1.7). The descriptors are then quantized into *visual words* using k-means clustering. Finally, from each slice of the sliding window, a bag–of–words (BOW), i.e. a histogram representing the number of occurrences of each visual word in the slice, is computed as feature for the WRS.

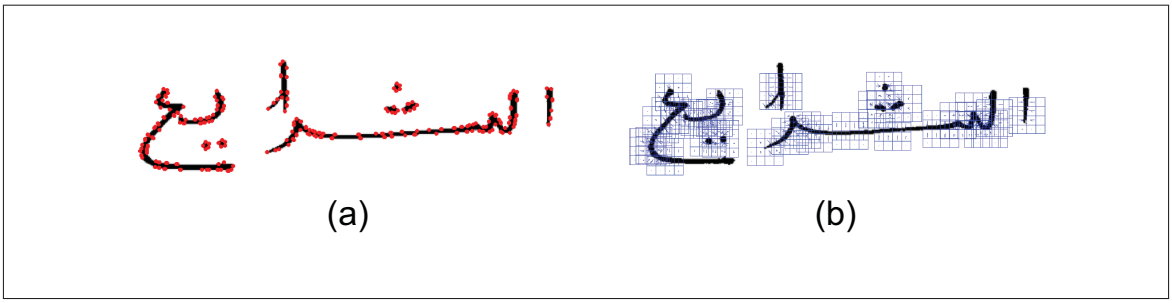


Figure 1.7 SIFT computation from Arabic word image. (a) Keypoints detected with Harris corner detector. (b) A few SIFT descriptors.
Image reproduced from Rothacker *et al.* (2012), © 2012 IEEE.

Rodríguez-Serrano and Perronnin (2009) developed a SIFT like feature called LGH features in their word spotting application. The image is divided into overlapping frames. The region in each frame is divided into 4×4 regular cells. Next, in each cell a histogram of gradients is computed (8 bins) and the final vector is the concatenation of the 16 histograms which results in 128D feature vector for each frame. Each feature vector is scaled to unit norm for local contrast normalization. The image is smoothed by a Gaussian filter before the gradient computation. They have shown that LGH feature provides better performance accuracy in handwritten text recognition.

1.3.4 Automatically learned features

Automatically learned features are based on the neural network technology. The main advantage of NN is that they are discriminative models and provide better results. The first use of NN

has been done through the combination of a multilayer perceptron (MLP) with HMM in the so-called NN/HMM hybrid system, where the observation probability is based on the output of MLP instead of the classical Gaussian mixture model (GMM). This idea has been extended to tandem systems, where the MLP is used as a feature extraction module (Hermansky *et al.*, 2000; Dreuw *et al.*, 2011). The training of tandem system involves several steps. First, the word image frames are given the label of their characters, either manually or by using a previously trained HMM in the forced alignment mode. Then, the MLP is trained to recognize the label of the frames without feature extraction. Finally, the output of the MLP followed by dimensionality reduction is considered as the extracted feature for a new HMM model.

The RNN based WRS can also automatically learn features using the MDLSTM neural network architecture (Graves and Schmidhuber, 2009). This network is a multidimensional extension of the LSTM network. In this setting, the multidimensional data is scanned as a 1D sequence, by setting the scanning directions and the dimensions scanning priority. For example, in a 2D image, we can choose to scan forward along the x dimension and backward along y, with a higher priority for the x than for y, so that during the scan, the x index will be updated before the y index according to the scanning directions. Each hidden layer memory block has a recurrent connection with the memory blocks one step back according to the scanning direction for every dimension. One such layer provides the network with full context along the scanning direction. As there are 4 possible directions in 2D images (for example forward x and y, backward x and forward y and so on), 4 layers are necessary to have full context along all directions (Figure 1.8). Similarly to the LSTM layer, it is possible to have multiple layers scanning in the same direction, and to combine them to form multiple feature maps at the output layer. Also, a hierarchy of MDLSTM layer can be built, with 2D subsampling between layers.

1.4 Current limitations

In this section we highlight the limitations of the current state-of-the-art features for Arabic script LR and for handwriting recognition.

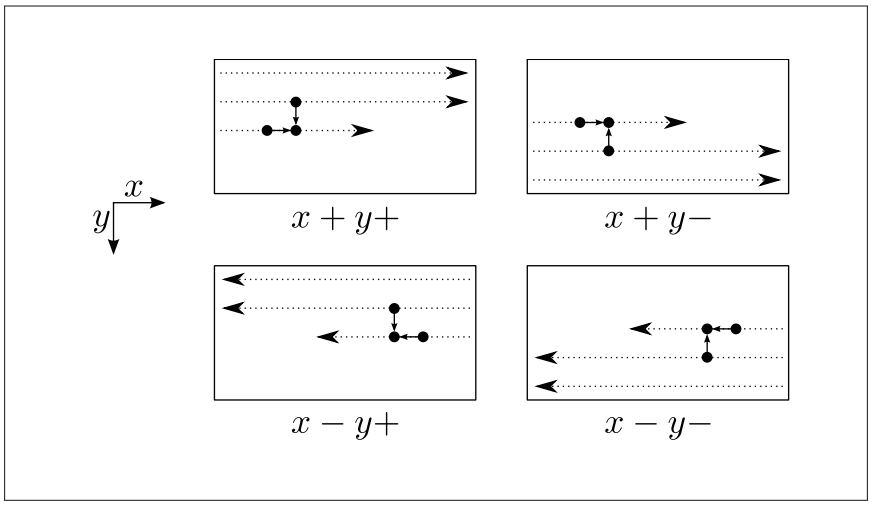


Figure 1.8 2D MDLSTM scanning directions and context propagation in hidden layers. The priority direction is x. $+$ forward direction and $-$ backward direction.

1.4.1 Limitation 1: Lack of descriptors tailored for Arabic script lexicon reduction

Several works exist in the literature for LR on Latin script, they are based on salient high-level features (ascenders/descenders, shape codes), which are relatively easy to map with word shape belonging to the lexicon using error tolerant schemes. Nevertheless, until recently, LR for Arabic script has received little attention and very few methods have been developed. Because the Arabic script has a different appearance than the Latin script, existing LR methods are based on a different set of salient features. Arabic script graphemes have a strong structure, where subwords and diacritics are important units, unlike in Latin script. Indeed, the subword concept is nonexistent in Latin script. Existing methods for Arabic LR started to build on diacritic and subword count features. Nevertheless, is it possible to go beyond and integrate more information? For example, can we consider the shape of the graphemes? A question that yet remains to be answered is how can we efficiently represent all the salient information of Arabic word images for LR? A descriptor with such properties would improve our understanding of relevant features for the recognition of the Arabic script and how to efficiently represent them.

1.4.2 Limitation 2: Lack of methods to identity relevant features for handwriting recognition

The literature exhibits a large body of features for handwriting recognition in Latin and Arabic script, and the quest for the 'ultimate' feature is certainly not over yet. Existing features are based on different models originating from various fields such as pattern recognition, computer vision or machine learning. Because of their different backgrounds, it is very difficult to compare them on theoretical bases. Moreover, they are often used on different databases, with different protocols and recognition systems. Therefore, it is difficult to decide which feature one should use for new applications. Indeed, the literature does not provide clear guidelines and it is mostly limited to a listing of all the features ever proposed. This situation leads to the proposal of more and more features, with no principled design for the task of handwriting recognition. Although new features certainly achieve significant contribution in their respective fields (computer vision, machine learning, etc.), their contribution is not clear into the context of handwriting recognition, which uses features from several fields. Therefore, it is important to compare existing features first and identify the most promising ones. Nevertheless, no tool exists for this task, except the evaluation based on the recognition rate, but this approach just provides a shallow insight on the features. Therefore, the creation of efficient tools for feature evaluation is needed, so that the next generation of features can be more efficient for handwriting recognition.

CHAPTER 2

GENERAL METHODOLOGY

In this chapter we expose our general methodology as well as the rationale. It is in accordance with the main purpose of this thesis, which is to improve our understanding of relevant features for WRS systems. This knowledge will help to improve word recognition systems with better features. Understanding what features are relevant for handwriting recognition requires two components, first the features, and second a proper evaluation methodology. Nevertheless in the case of new applications, no or few features already exist, so that new features must be designed, usually from expert knowledge. Therefore, *feature design* and *evaluation* are the two main components of our pattern recognition framework. They will be investigated through two specific aspects of WRS. The first one is LR, and we focus specifically on Arabic script because this field of research has not been much investigated yet. The second aspect is word recognition and its features for cursive script (both Latin and Arabic). First, the research objectives arising from the limitation of current literature are defined. Then, the general approach of this thesis is explained.

2.1 Research objectives

As stated in the introduction, the **main purpose of this thesis is to improve our understanding of relevant features for WRS**. It will be achieved with 3 specific objectives, related to descriptors for Arabic LR and feature evaluation for WRS.

2.1.1 Objective 1: to design a descriptor for Arabic subword shape with application to LR

Existing methods for LR in Arabic scripts focus mainly on the number of subwords and type of diacritics present in the word and ignore the shape of the subwords. Therefore, the first objective is to propose a LR method based on the shape of Arabic subwords. Arabic subwords are components of Arabic words that are quite easy to identify, as they are separated by spaces.

Moreover, they exhibit a strong structure given their loops and branches. These observations led our interest towards a holistic approach, where a subword is considered as a whole, and to graph modeling to represent their rich structure. Nevertheless, graph-based methods are known to have a high complexity. This is not compatible with LR methods which should be fast enough to avoid any significant overhead to the recognition system. A similar problem exists in the shape analysis community, where shapes of individual objects represented by graphs must be efficiently compared with a large database of sample shape. Therefore, our proposed approach for LR is inspired from structural shape analysis methods, and it will be developed in Chapter 3. It provides a first model for Arabic subword shape descriptor in the context of LR and a new holistic strategy for LR. Nevertheless, it does not consider the symbolic features of Arabic words.

2.1.2 Objective 2: to efficiently embed all Arabic word features into a descriptor with application to LR

Arabic words have two main components, the subwords and the diacritics, which are themselves described by their topology and geometry. Currently, there is no LR framework based on all these features of Arabic words. The second objective is to define a single descriptor embedding efficiently all these features. The descriptor, as a feature vector, allows efficient comparison and therefore a low computational overhead. One of the key challenges lies in embedding all the relevant information into this feature vector. Indeed, how to describe low-level features (subwords structure) and high-level features (subwords, diacritics) into a single vector? Also, in order to embed information about diacritics and subwords, they should be identified first. This would require an explicit classification which can be time-consuming. An interesting avenue is to extract structural information for each component (subwords and diacritics), and combine them altogether using efficient heuristics instead of explicit identification. The method will be developed in Chapter 4. It provides the first holistic descriptor for Arabic words in the context of LR. It has a low computational overhead and the potential to lead to holistic Arabic handwriting recognition.

2.1.3 Objective 3: to efficiently evaluate features for the task of handwriting recognition

It is important to evaluate existing features and assess their strengths and weaknesses, in order to build better features. The evaluation of features for handwriting recognition is a difficult task because their potential is revealed through the performance of the WRS. The basic approach based on the recognition rate only provides a relative ranking between features. It has the weakness of ignoring the complementary aspect of features. Indeed, in ranking based on the recognition rate, the top two features may just be variants of each other, and therefore not complementary, while the last ranked feature could be complementary to the first. Such information, missing from the basic ranking, is essential for efficient feature evaluation. Therefore, the third objective is to develop a framework for the evaluation of features used in handwriting recognition, providing insight on their efficiency and complementarity. It relies on the hypothesis that features can be evaluated through the performance of WRS in a combination scheme, where the WRS are not evaluated individually, but with respect to each other. The method will be detailed in Chapter 5. The contribution of this approach is to attribute a score to each feature, which is easy to interpret. This allows identifying the most promising features and understand which models are complementary.

2.2 General approach

New descriptors and methods have been developed in this thesis, for a better understanding of relevant features for handwriting recognition. They are directly linked to the previously mentioned objectives, and they are split in two main themes: descriptor design for lexicon reduction in Arabic script, and feature evaluation for handwriting recognition in Latin and Arabic scripts.

2.2.1 Descriptor design for Arabic lexicon reduction

Two new descriptors have been proposed for Arabic lexicon reduction. They integrate relevant features for the description of Arabic word shapes and improve LR performance.

The first objective investigated a new strategy for lexicon reduction, based on the shape of Arabic subwords. Because they have a strong structure, with many branch points and loops, graph-based modeling is appropriate. Therefore, our method is based on graph indexing, where a signature vector is extracted from each graph for efficient comparison against a large graph database. For this purpose, we propose the weighted topological signature vector (W-TSV) framework for *directed acyclic graph* (DAG) with weighted edges, in which a feature vector is built using the eigenvalues of the weighted adjacency matrix of the graph and a careful construction to preserve the graph structure. It is an extension of the TSV framework (Shokoufandeh *et al.*, 2005) for (non-weighted) DAG to weighted DAG. The shape of Arabic subwords is modeled in a DAG as follows. First, the shape of Arabic subwords is identified in the image as a connected component (CC). Its skeleton is computed in order to highlight its structure (topology and geometry). It is then modeled as a DAG based on its skeleton keypoints and the geometry of the skeletal curves. Three alternative models are proposed, which integrate different geometrical information such as skeletal curve length and curvature. The weighted DAG is embedded into a low-dimensional vector space using the W-TSV framework. Lexicon reduction is achieved by comparing the W-TSV vector of a query shape with the vectors of a labeled database. The labels of the nearest neighbors of the query shape in the W-TSV space constitute the reduced lexicon. The main contribution of this work is to provide a new framework (W-TSV) for efficient indexing of weighted DAG. Therefore, it introduces a new framework for lexicon reduction based on shape indexing. Moreover, three DAG models for subword shapes have been proposed and compared. The proposed method outperformed previous approaches based on diacritics for lexicon reduction at the subword level (Chapter 3).

The second objective investigated the design of a descriptor for Arabic words integrating all the Arabic words main features: namely the structure (geometry and topology) of Arabic subwords and symbolic information such as the count of subwords and diacritics. For this purpose we developed the Arabic word descriptor (AWD) which is built in two stages. First, a structural descriptor (SD) is computed for each CC of the word image. It describes the CC shape using the BOW model for compact encoding, and it is constructed as follow. Filters with different geometrical patterns and scales, similar to the Haar-like filters (Viola and Jones, 2004), are

applied to the pixels of the CC skeletons. The output of the filters forms a feature vector for each skeletal point. The feature vectors are then quantized, by assigning them to their nearest visual words from a predefined codebook. The SD is then formed as a histogram representing the number of occurrences of each visual word. Finally, the AWD is formed by sorting and normalizing the SDs of all the CCs. The AWD implicitly encodes several levels of information. The structure of the subwords shapes is encoded into the SD, and subword count and diacritics by the length of the AWD. The subwords are expected to be ranked first and the diacritics last because the SD sorting is based on the number of pixels of each CC. Therefore, the distinction between subwords and diacritics is also implicitly present, based on their positions into the AWD. Lexicon reduction is performed using the database indexing scheme introduced in the first method, but using the AWD as index. The impact of the LR approach on a holistic subword recognition system and an analytic WRS has been tested, regarding the processing time and recognition rate. The main contribution of this method is to provide a holistic descriptor for Arabic words' shape, which seamlessly integrates low-level features (geometry and topology of subwords) and high-level features (subword counts and diacritics). Furthermore, its construction has a low computational cost because an efficient heuristic is used to implicitly discriminate between subwords and diacritics, which avoids layout analysis and an explicit classification altogether. Finally, the proposed method provides the best results for LR in Arabic script on two benchmarks, and it has been demonstrated that with a proper LR approach, shape matching methods can be applied for Arabic subword recognition with low processing time (Chapter 4).

2.3 Feature evaluation for handwriting recognition

The third objective proposes a framework for feature evaluation for handwriting recognition. Features are indirectly evaluated through the performance of a reference WRS using RNN. In this framework, at least one instance of the reference system is trained for each feature, and we refer to these instances as agents of that feature. All the agents are then evaluated w.r.t. each other by using a combination scheme at the decision level (recognized words). Each agent is assigned a fixed weight, and the final recognition is taken according to a variant of

the weighted vote. More precisely, each agent votes for a given word, and the word with the highest number of weighted votes is selected as the final recognition. The weights of this combination are optimized during a training phase, in order to maximize the weighted vote of the true word label. The weights are set based on the collective performance and not individual performance because the decisions of all the agents are known during optimization. Therefore, the weights represent the contribution of each agent to the vote, based on their mutual strength. The weights are then converted into easy to interpret scores and are assigned to the features of the agents. The main contribution here is to provide a feature evaluation framework, which assigns a score to each feature, thus measuring their importance relatively to each other. Five features have been evaluated among the following categories: distribution features, concavity, visual-descriptor based and automatically learned features. These categories have been chosen either because of their state-of-the-art performance (distribution and concavity features), or because they represent recent trends in feature design, inspired by computer vision and machine learning. As an outcome of this study, the results show that distribution features are the most efficient, and complementary with visual-descriptor and automatically learned features (Chapter 5).

CHAPTER 3

ARTICLE I - W-TSV: WEIGHTED TOPOLOGICAL SIGNATURE VECTOR FOR LEXICON REDUCTION IN HANDWRITTEN ARABIC DOCUMENTS

Youssouf Chherawala and Mohamed Cheriet

Synchromedia Laboratory, École de Technologie Supérieure
1100 Notre-Dame Ouest, Montréal, QC, Canada

Published in Elsevier Pattern Recognition
Volume 45, Issue 9, September 2012, Pages 3277-3287

Abstract

This paper proposes a holistic lexicon-reduction method for ancient and modern handwritten Arabic documents. The word shape is represented by the weighted topological signature vector (W-TSV), which encodes graph data into a low-dimensional vector space. Three directed acyclic graph (DAG) representations are proposed for Arabic word shapes, based on topological and geometrical features. Lexicon reduction is achieved by a nearest neighbors search in the W-TSV space. The proposed framework has been tested on the IFN/ENIT and the Ibn Sina databases, achieving respectively a degree of reduction of 83.5% and 92.9% for an accuracy of reduction of 90%.

Keywords

Lexicon reduction, Arabic handwritten documents, Ancient documents, Weighted topological signature vector (W-TSV), Graph indexing, IFN/ENIT, Ibn Sina database.

3.1 Introduction

Handwritten word recognition systems have improved in a number of ways in recent decades, across many applications, from the recognition of the legal amount on bank checks and of postal addresses (Kaufmann and Bunke, 2000; Kim *et al.*, 2001; Srihari, 1993; Liu *et al.*, 2002; Al-Hajj Mohamad *et al.*, 2009) to the automated transcription of ancient documents (Lavrenko *et al.*, 2004; Feng *et al.*, 2006; Vamvakas *et al.*, 2008; Wüthrich *et al.*, 2009; Fischer *et al.*,

2010). While the vocabulary for a bank check application is small (fewer than 30 words), it is large for postal applications (1,000 words) and unconstrained for historical documents (several thousand words). A vocabulary of valid words that are expected to be recognized by the system is called a *lexicon* (Koerich *et al.*, 2003). A large lexicon generates a high computational complexity, as all the word hypotheses must be tested, and recognition performance decreases as the number of allowed hypotheses grows. To address this problem, lexicon-reduction methods are used. When a query word shape is submitted for recognition, the lexicon is pruned by keeping only the shapes that are most likely to correspond to the query word class (Koerich *et al.*, 2005), or by using application-dependent knowledge (Tomai *et al.*, 2002). Then, the recognition system considers the word hypotheses remaining in the pruned lexicon. The performance of a lexicon-reduction method is classically evaluated based on its accuracy of reduction α (the probability that the query word class was included in the pruned lexicon), the degree of reduction ρ (the decrease in the size of the lexicon after pruning), and the reduction efficacy η, which is a combination of the two previous criteria. Computational complexity is also a major factor in lexicon reduction, as one of its goals is to speed up the recognition process. In this paper, we propose a lexicon-reduction method for handwritten Arabic documents, both ancient and modern.

The Arabic language has an alphabet of 28 letters. The script is cursive and written from right to left. One important feature of Arabic letters is that their shapes are context-dependent, which means that a letter shape is usually determined by its position in a word, i.e. initial, medial or final. The letters have no cases and many share the same base shape. They are distinguishable by the addition of diacritical marks. The diacritics used in Arabic for this purpose are dots, one, two, or three of them appearing below or above the base shape. If we ignore the dots, we obtain the archigraphemes (Figure 3.1), where a single grapheme (letter shape) can represent many letters. Four archigrapheme letter shapes (‘A’, ‘D’, ‘R’, ‘W’) can be connected only if they are in the final position. If they appear in the middle of a word, the word is divided into subwords, also known as *pieces of Arabic word* (PAW).

A	B	G	D	R	S
ا	ب	ح	د	ر	س
C	T	E	F	[F] - Q	K
ص	ط	ع	ف	ق	ك
L	M	[B] - N	H	W	[B] - Y
ل	م	ن	ه	و	ى

Figure 3.1 Arabic transliteration table. If a transliteration is defined in brackets, it is used when the letter is not in the final position in a subword.

The goal of this paper is to provide a lexicon-reduction strategy for Arabic documents, based on the structure of Arabic subword shapes, which is described by their topology and geometry. First, the topological and geometrical properties of the subword shapes are extracted from the shape skeleton. Then these properties are encoded in a *directed acyclic graph* (DAG) in order to preserve information about their relationship in the skeleton. Finally, the subword DAG is transformed into a vector using the weighted topological signature vector (W-TSV), which is an extension of the TSV (Shokoufandeh *et al.*, 2005) for weighted DAGs. Like the classical TSV, the W-TSV is a powerful tool for encoding structured data, such as a DAG, mapping the DAG to a low-dimensional vector space for fast matching. Also, it has good discriminatory power for DAGs with different topologies, because it preserves their topological properties to some extent. Unlike the TSV, the W-TSV can also discriminate between DAGs sharing the same topology, but with different weights, and it is more robust to topological perturbation than the TSV under small weight perturbation. In this work, lexicon reduction is performed by pruning the reference database of subword/word shapes. This is achieved by selecting the i nearest shapes in the database to a query shape in the W-TSV space. First, the database is indexed by ordering its shapes in ascending order, based on their distance from the query shape; next, the lexicon is reduced by selecting the first i elements of the indexed lexicon as candidates. The value of i is evaluated during a training phase in order to reach the accuracy of reduction level selected for the application. The same i value is then applied for all the query shapes during the lexicon reduction process. From the reduced database of shapes, it is then

possible to build a reduced lexicon of subwords/words from the labels of the selected shapes (Figure 3.2).

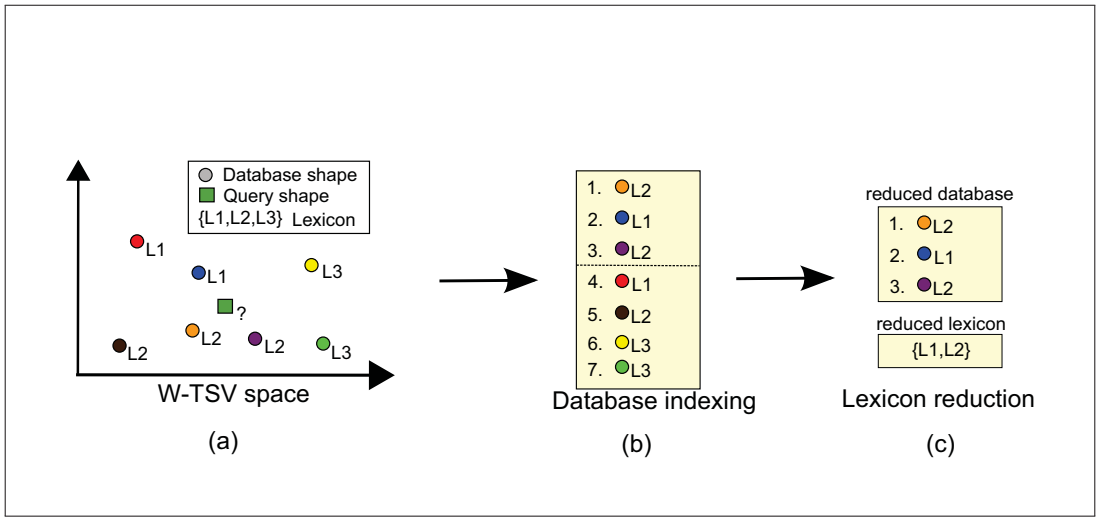


Figure 3.2 Lexicon reduction based on the weighted topological signature vector (W-TSV). (a) query shape comparison in the W-TSV space; (b) database indexing based on W-TSV distance; (c) lexicon reduction by selection of the first 3 candidates.

This paper is organized as follows. The features of lexicon reduction for ancient and modern Arabic documents are described in section 3.2. Related work on lexicon reduction is reviewed in section 3.3. The details of the W-TSV scheme and of the formation of the Arabic subword DAG are respectively provided in section 3.4 and section 3.5. Finally, the details of our experiments and our results are given in section 4.6, followed by the conclusion in section 4.7.

This paper is an extension of the work published by Chherawala *et al.* (2011). The underlying methodology, as well as the experimental evaluation, have been significantly improved.

3.2 Features of ancient and modern Arabic documents for lexicon reduction

The nature of ancient Arabic documents is different from that of the Arabic documents used in modern applications. The study of ancient documents is motivated by their cultural significance, and a vast number of them have been scanned as digital images in order to protect them from aging. Pre-modern Arabic documents were written during the medieval period. They can

be written in a variety of calligraphic styles, depending on when and where they were copied. The appearance of a written text changes greatly from one style to another. For example, the Kufic style consists of straight lines and angles, while the Naskh style is curved and supple (Figure 3.3). The diacritics, when they are included at all, tend to float around the subword shape, and their location is more often determined by esthetic considerations than by their immediate proximity to the corresponding letter. This makes it difficult to assign the diacritics to the correct subword, especially when the line spacing is reduced. Most of the time, these documents are written by a single author. The lexicon is unconstrained, and the segmentation of Arabic subwords into words is not known a priori.

(a) Kufic style (b) Naskh style

Figure 3.3 Pre-modern Arabic documents.

Arabic word recognition at the subword level is therefore well suited to ancient Arabic documents, as subwords can be easily identified, usually as connected components. In spite of the fact that the diacritical marks, especially the dots, are important cues for discriminating between different letters, this feature is unreliable in these documents for the reasons explained above. They must be ignored in the first stage, so that the correct archigrapheme can be recognized. In this work, the lexicon for ancient documents is composed of a vocabulary of naked subwords (Arabic subwords written with archigraphemes). Lexicon reduction is performed in this step, which it is not in the classical approaches. This is because the number of different subwords is smaller than the number of Arabic words, and also because many subwords dif-

ferientiated only by diacritical marks correspond to the same naked subword. The recovery of the correct subword from a naked subword can be achieved in a post-processing step by considering the neighboring diacritical marks. A W-TSV is assigned to each subword shape for the lexicon reduction process.

The study of modern Arabic documents is motivated by specific application needs. The recognition system has to deal with a wide variety of writers and the vocabulary is usually large. The segmentation of Arabic text into words can be estimated from the layout of the document, and the diacritics are usually well positioned. Thus, the lexicon for such documents is composed of Arabic words directly, according to the application needs. For lexicon reduction, a W-TSV is assigned to each connected component of the word image (subwords and diacritics), and these are combined into a single W-TSV for the word shape.

3.3 Related works

Lexicon reduction can be performed by comparing the optical shapes of the lexicon words to improve recognition speed. When the word's optical shape is used, the simplest criterion for lexicon reduction, but still efficient, is word length, as this makes it easy to discriminate between long words and short words. More refined knowledge about the word's shape can also be used. Zimmermann and Mao (1999) propose the concept of key characters, which are characters that can be accurately identified without a full contextual analysis. Lexicon reduction is performed by considering only the lexicon entries that match the regular expression generated by the key characters. They also estimate the letter count in a word using a neural network for further reduction. A similar approach is proposed by Palla *et al.* (2004), where regular expressions are built from the detection of ascenders and descenders in the query word image. Bertolami *et al.* (2008) propose mapping each character of a word to a shape code. There are fewer shape codes than characters, as they only discriminate between characters based on their ascenders/descenders and basic geometry. The mapping is performed by a hidden Markov model (HMM), which outputs the n best shape-code sequences for a query word. The lexicon is reduced by considering only the words that correspond to one of the shape-code sequences. Kaufmann *et al.* (1997) propose a holistic approach, using the quantified feature vectors as

shape descriptors. These vectors are used by the HMM recognizer, so there is no overhead for the extraction of these features. A model is created for each class of the lexicon, and the word hypotheses are ranked according to the distance between their models and the shape descriptor of the query word. Several other holistic approaches for lexicon reduction extract a string-based descriptor for each shape, which is further matched using dynamic programming, the lexicon entries with the smallest edit distances being considered part of the reduced lexicon. Madhvanath *et al.* (2001) holistic approach is based on using downward pen-strokes descriptors. These pen strokes are extracted from the word shape using a set of heuristic rules, and categorized according to their positions relative to the baseline. Then, lexicon reduction is performed by matching the word descriptors to the ideal descriptors extracted from the lexicon's ASCII string. Carbonnel and Anquetil (2004) compared two lexicon-reduction strategies, one based on lexicon indexing and the other on lexicon clustering. Using ascender/descender-based shape descriptors, the indexing approach showed better performance. Arabic word shapes have a rich structure, with their loops, branches, and diacritics (Lorigo and Govindaraju, 2006; Abuhaiba *et al.*, 1994; Zidouri, 2004). These structural features have been used for lexicon reduction. Mozaffari *et al.* (2008a) propose a two-stage reduction of an Arabic lexicon. In the first stage, the lexicon is reduced based on the number of subwords of the query word. In the second stage, the word's diacritical mark types and positions are encoded into a string, and the lexicon is reduced based on the string edit distance. Mozaffari *et al.* (2008b) extended the previous approach to Farsi handwritten words, which contain more letters than the Arabic alphabet. Wshah *et al.* (2010) propose a similar algorithm, in which the diacritic detection stage is improved by the use of a convolutional neural network. Farrahi Moghaddam and Cheriet (2009) have devised a word-spotting algorithm for pre-modern Arabic documents based on the shape structure of subwords. The first stage of the algorithm consist of lexicon reduction using a self-organizing map (SOM). The SOM is trained using a feature vector of the topological and geometrical properties of the subword skeleton. Once a query shape has been fed to the SOM, only the lexicon of the activated cell and the neighboring cells is considered for further matching. Several lexicon-reduction approaches use application dependent knowledge to improve the system's recognition rate. For the transcript mapping problem with ancient document images, Tomai *et al.* (2002) propose recognizing each word of a document image by reducing the

lexicon to specific lines of the transcript. Morita *et al.* (2002) have taken advantage of the date field structure for the recognition of handwritten dates on bank checks. Milewski and Govindaraju (2004) use an application-specific lexicon for word recognition on medical forms, while Farooq *et al.* (2009) have proposed estimating the topic of a query document from the output of a word recognizer. As the performance of a word recognizer is very low without a priori knowledge, Farooq et al. used the n best hypotheses for each word, instead of only the first, to infer the document topic. Once the document topic has been found, the query document is submitted again to the word recognizer, but this time with the topic-specific lexicon.

3.4 Weighted topological signature vector (W-TSV)

3.4.1 Background

The classical topological signature vector (TSV) is an efficient encoding of the topology of structured data, such as a directed acyclic graph (DAG). The topology of a given DAG G can be represented by its adjacency matrix A, where $A(i,j) = 1$ if an edge goes from vertex v_i to vertex v_j, $A(i,j) = -1$ if an edge goes from vertex v_j to vertex v_i, and $A(i,j) = 0$ in all other cases. The adjacency matrix is therefore antisymmetric. From the adjacency matrix, a signature S_G for the graph G can be extracted as the sum of the magnitude of its m eigenvalues:

$$S_G = |\lambda_1| + \ldots + |\lambda_m| \tag{3.1}$$

In order to enrich the signature representation of the graph, such a signature is extracted from all the subgraphs of V, the source of the DAG (vertex with no incoming edges). If V has a degree n, the n signatures of its subgraphs and the graph signature are sorted by descending order and concatenated to form the TSV:

$$\chi(G) = \begin{bmatrix} S_G & S_{G_1} & \ldots & S_{G_n} \end{bmatrix}^T \tag{3.2}$$

The largest signature corresponds to the DAG with the richest topology. Therefore, the signature of the graph source S_G will always be larger than the signature of the subgraphs of the

source, and will always be the first dimension of the TSV. As the degree of the source of the DAG changes from one graph to another, the size of the TSV is set, in advance, to a given value p. If the size of the TSV of G is smaller than p, then the TSV vector is padded with 0, and if the size of the TSV is larger than p, then the TSV is truncated. The truncation removes the less informative signatures, so it is safe to remove them when needed. The value of p can be set according to the maximum degree of the source of the DAGs of the database, or according to a chosen complexity for the indexing process. An illustration of the formation of the TSV of G with a source V is presented in Figure 3.4. The source V has two subgraphs G_a and G_d, and so the topological signature is computed for G, G_a and G_d. Their signatures are sorted in decreasing order to form the TSV $\chi(G)$ of size $p = 5$, with the appropriate padding by 0. The adjacency matrix of G_a is also shown.

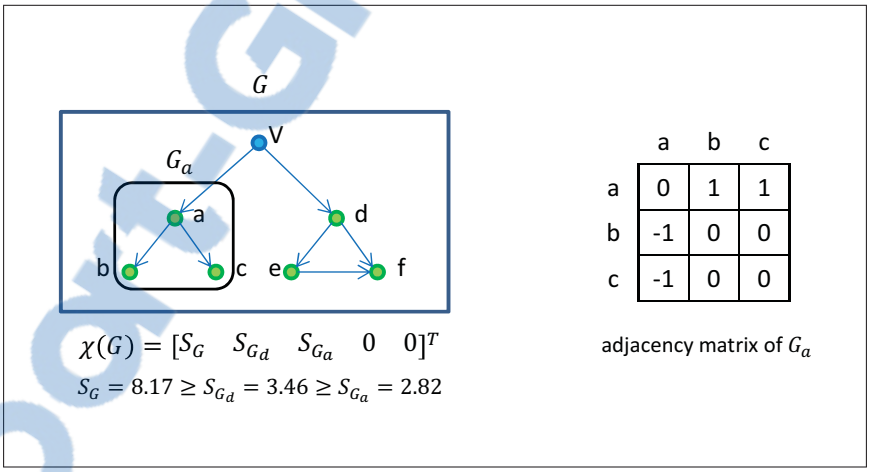


Figure 3.4 Topological signature vector formation for the DAG G.

The TSV has many properties that make it well suited to the indexing of DAG databases. First, it is invariant to consistent reordering of the graph branches. Such reordering does not affect the graph's topology, but it does lead to a different adjacency matrix. In fact, the branch reordering is equivalent to a permutation of the adjacency matrix. As the eigenvalues of an antisymmetric matrix are invariant to any orthonormal transformation, such as a permutation, the TSV is also invariant. Second, the TSV has been shown to be robust to minor perturbations of the graph structure. More precisely, the error between the eigenvalues of an adjacency

matrix and its perturbed version is bounded by the largest eigenvalue of the perturbation matrix (see Section 3.4.3). This property is very useful, as natural data are often noisy and it is difficult to avoid minor perturbations, such as vertex splits or merges, in practice. The last but not the least property of the TSV is to map structured data into a low-dimensional vector space. The matching of structured data such as DAG has polynomial complexity, while the matching of vectors has linear complexity on the dimension of the vector space. Therefore, the TSV achieves a substantial decrease in complexity and makes the indexing of a DAG database efficient.

3.4.2 Generalization to weighted DAG

The TSV only considers the topology of the DAG. Nevertheless, since DAG edges are often weighted, this information can be useful for discrimination. This leads us to propose a new formulation, the weighted TSV (W-TSV), where the weight information is added to the adjacency matrix. One of the main idea behind the W-TSV is that edges with large weights are more important than edges with small weights. Let $W_G = \{w_{ij}\}$ be the set of edge weights of DAG G, such that w_{ij} represents the weight associated with an edge extending from vertex v_i to vertex v_j and $w_{ij} > 0$. The weighted adjacency matrix A of G can be constructed as follows: $A(i,j) = w_{ij}$ for an edge from vertex v_i to vertex v_j, $A(i,j) = -w_{ij}$ for an edge from v_j to v_i, and $A(i,j) = 0$ otherwise. A weight $w_{ij} = 0$ means that there is no edge between v_i and v_j. In the rest of the paper, the weights of A will refer to W_G, and $w_{ij}(A)$ will refer to $|A(i,j)|$, i.e. the weight of the edge between v_i and v_j, irrespective of its direction. The W-TSV is computed in a same manner as the TSV, the only difference being that the weighted adjacency matrix is used instead of the classical adjacency matrix. Consider the function $\Gamma : \mathbb{R}^+ \rightarrow \{0,1\}$:

$$\Gamma(w) = \begin{cases} 1 \text{ if } w > 0 \\ 0 \text{ otherwise} \end{cases}$$

When applied to all the weights of A, Γ removes the weight information completely, but it preserves the topological property, mapping the weighted adjacency matrix to the classical definition of the adjacency matrix. We note that the classical adjacency matrix is a special case

of the weighted adjacency matrix, so from now on we will use the term 'adjacency matrix' instead of 'weighted adjacency matrix'. The discriminative power of the W-TSV over the TSV for weighted DAG is illustrated in Figure 3.5. The 3 DAGs share the same topology, but they have different edge weights. As a result, their TSV is identical, while their W-TSV is different. The TSV and W-TSV are identical for G_1, because all its weights are equal to 1.

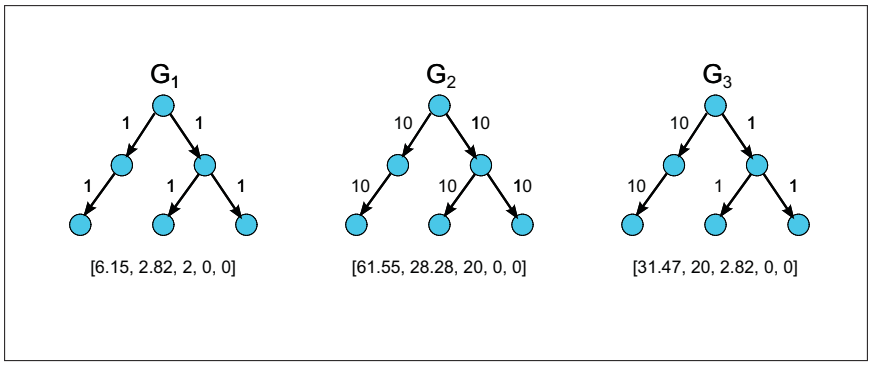


Figure 3.5 Three DAGs with different weights, but sharing the same topology, and their corresponding W-TSV. They have different W-TSV, but the same TSV (for G_1, its TSV and W-TSV are equal).

3.4.3 Stability and robustness of the W-TSV

The W-TSV uses topological and weight information. In order to be an efficient encoding, it must remain stable and robust under topological and weight perturbations, i.e. the changes in the W-TSV values induced by a perturbation must be commensurate with the perturbation level. In this section, we show the stability of the W-TSV , and its robustness compared to the TSV, under the assumption of small weights perturbation (the notion of 'small' is further explained in Proposition 3). The idea here is that noise will be more likely to introduce small weight perturbation than large weight perturbation. The stability of the W-TSV will be studied using graph spectral theory. Consider the graph G and its $m \times m$ adjacency matrix A. A lifting operator $\Psi : \mathbb{R}^{+m \times m} \rightarrow \mathbb{R}^{+n \times n}$ can be used to create an $n \times n$ adjacency matrix $\Psi(A)$ ($n \geq m$) equivalent to A upto vertex relabeling. This operator will first add $n - m$ zero-valued rows and columns to A, forming the matrix A'. Then given a permutation matrix P, the vertices

are relabeled so that $\Psi(A) = PA'P^T$. As A and A' have the same spectrum up to additional 0 elements, and A' and $\Psi(A)$ have the same spectrum, $\Psi()$ is a spectrum preserving operator.

A perturbed graph H can be built from G using the lifting operator and an $n \times n$ perturbation matrix E, where $B = \Psi(A) + E$ represents the adjacency matrix of H. A weight $w_{ij}(\Psi(A))$ is perturbed by adding (substracting) if $\Psi(A)_{ij}$ and E_{ij} have the same (opposite) sign. We can distinguish three types of perturbation:

- weight perturbation: $w_{ij}(\Psi(A)) > 0$ and $w_{ij}(E) \neq 0$,
- edge addition: $w_{ij}(\Psi(A)) = 0$ and $w_{ij}(E) > 0$,
- edge deletion: $w_{ij}(\Psi(A)) = w_{ij}(E) > 0$, and $E_{ij} = -\Psi(A)_{ij}$.

We will assume that E is well conditionned, i.e. $B = \Psi(A) + E$ represents the weighted adjacency matrix of a valid DAG.

We can now show the stability of the W-TSV. Let $\lambda_i(A)$ denote the i^{th} largest element of the set of magnitudes of matrix A eigenvalues. Consider the following result (Shokoufandeh *et al.*, 2005):

Proposition 1. *If A and E are $n \times n$ antisymmetric matrices, then:*
$|\lambda_i(A + E) - \lambda_i(A)| \leq |\lambda_1(E)|$, *for* $i \in \{1, \ldots, n\}$.

Proposition 1 shows that the eigenvalues of the perturbed matrix $B = \Psi(A) + E$ are bounded by $\lambda_1(E)$. In the case where E represents a topological perturbation matrix (all the weights are equal to 1), $\lambda_1(E)$ is bounded by $\sqrt{k}$, where k is the number of edges of E (Neumaier, 1982). We generalize this result to weighted adjacency matrices, as follows:

Definition 3.1. *The weight vector $W(E)$ of an adjacency matrix E is the vector formed by concatenation of all the weights of E.*

Proposition 2. *If E is an $n \times n$ antisymmetric matrix, then the magnitude of its largest eigenvalue is bounded by the Euclidean norm of its weight vector: $\lambda_1(E) \leq \|W(E)\|$.*

Proof. E is antisymmetric hence $\lambda_1 j$ and $-\lambda_1 j$ are eigenvalues of E, where j is the imaginary unit. Therefore $2{\lambda_1}^2 \leq \sum {\lambda_i}^2 = -\mathrm{tr}\,(E^2) = \sum_{i,k} {w_{ik}}^2(E)$. Notice that $\sum_{i,k} {w_{ik}}^2\,(E)$ is the sum of all the elements of the matrix $[{w_{ik}}^2\,(E)]$ which is symmetric and has the diagonal equal to 0. It can be represented by its upper triangle matrix U such that $[{w_{ik}}^2\,(E)] = U + U^T$. The non zero entries of U are exactly the squared weights of E. Therefore the sum of all elements of U is equal to $W(E)^T W(E)$ and $\lambda_1(E) \leq \|W\,(E)\|$. □

Using Proposition 1 and Proposition 2, it is clear that the magnitude of the spectral distortion of a matrix $\Psi(A)$ from a perturbation matrix E is bounded by the magnitude of the weights of E. The W-TSV is therefore stable under minor weight perturbation of its corresponding DAG.

We will now show that the W-TSV is more robust to topological perturbation than the TSV, under the assumption of small weight perturbation. For this purpose, the weighted perturbation of A by E will be compared to the equivalent topological perturbation of $\Gamma(A)$ by $\Gamma(E)$. As the TSV is invariant to weight perturbation, we will consider only weighted topological perturbation, represented by a matrix E containing only edge addition and deletion. Nevertheless, the influence of E on the spectrum of A is related to the weights of A: it will be larger for small weights of A than for large weights. By contrast, the influence of $\Gamma(E)$ on the spectrum of $\Gamma(A)$ is not related to the weights of $\Gamma(A)$, as all the weights are equal to 1 in the topological case. Therefore E needs to normalized with respect to the weights of A, or $\Gamma(A)$ and $\Gamma(E)$ need to be rescaled with the weights of A, in order to compare the W-TSV and TSV fairly. We thus introduce the notion of scale for an adjacency matrix, as follows:

Definition 3.2. *The scale of an adjacency matrix A is the average value of all its weights.*

If the topological perturbation of $\Gamma(A)$ by $\Gamma(E)$ is performed at the same scale as A, the effect of the difference in magnitude between A and $\Gamma(A)$ is removed during the evaluation of their respective spectral distortion. The scale of $\Gamma(A)$ is 1 because all its weights are equal to 1. Let μ be the scale of A, the topological perturbation at this scale is performed by multiplying all the elements of $\Gamma(A)$ and $\Gamma(E)$ by μ. As a result, their respective eigenvalues are also multiplied by μ. The need for the notion of scale becomes obvious if we consider the same topological

perturbation but at different scales: $\mu_1 = 1$ and μ_2, such that $\mu_1 \ll \mu_2$; the distortion at scale μ_1 will be lower than at scale μ_2, although they both represent the same topological perturbation. In fact, the rescaling procedure handles such problems. Let $B(A, E)$ denote the upper bound of the magnitude of spectral distortion of A by E, then:

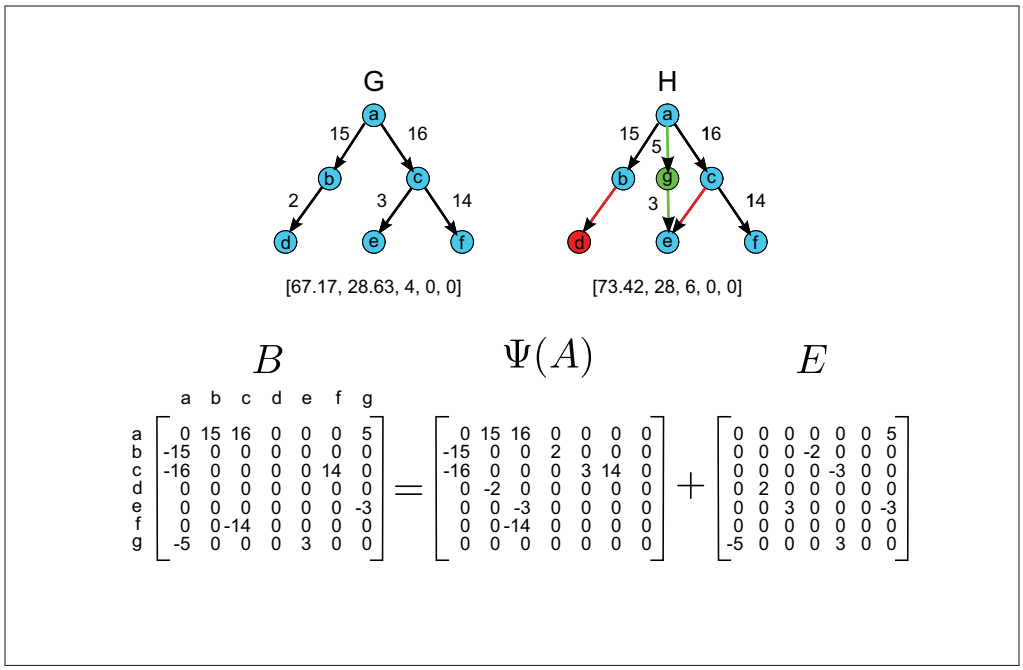
Proposition 3. *$\Psi(A)$ and E are antisymmetric matrices and E represents a weighted topological perturbation. If the root mean square (RMS) of the weights of E is smaller than the scale μ of A, then the upper bound $B(\Psi(A), E)$ is smaller than the upper bound of the equivalent topological perturbation at scale μ: $B(\Psi(A), E) < B(\mu\Gamma(\Psi(A), \mu\Gamma(E))$.*

Proof. Consider that E represents the weighted addition/deletion of k edges. Then $B(\mu\Gamma(\Psi(A), \mu\Gamma(E)) = \lambda_1(\mu\Gamma(E)) = \mu\sqrt{k}$. Also $B(\Psi(A), E) = \|W(E)\| = \sqrt{k}\alpha$ where $\alpha = \|W(E)\| / \sqrt{k}$ is the RMS of the weights of E. Given that $\alpha < \mu$, the result follows. □

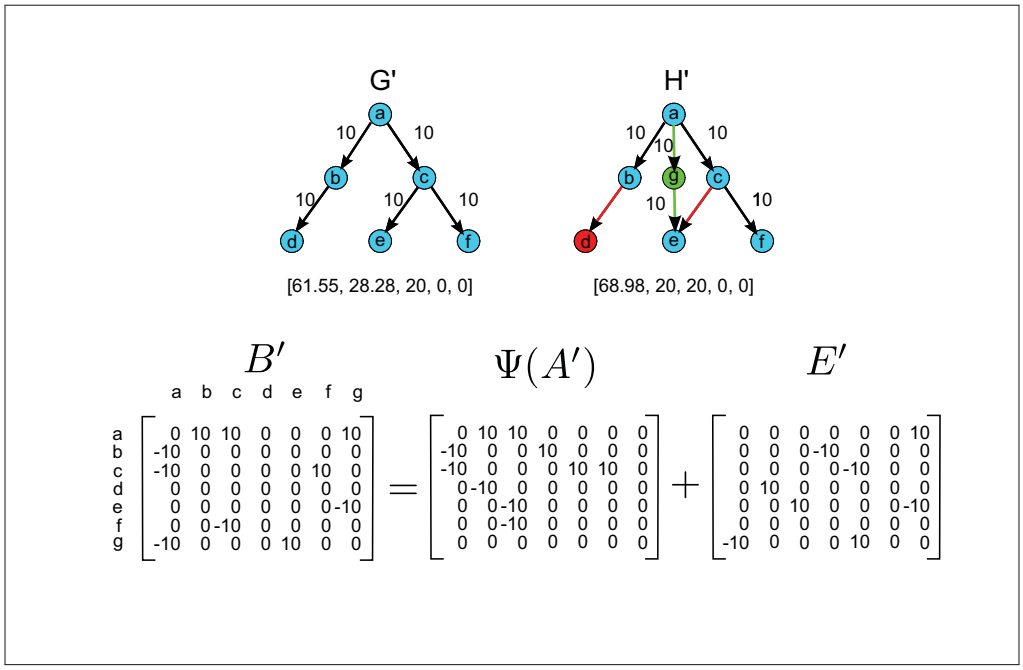
From Proposition 3, we can see that if the weights of the perturbation matrix E are small enough compared to the weights of A, the W-TSV is more robust than the TSV for topological perturbation at the same scale. An example is shown in Figure 3.6: in Figure 3.6a, the adjacency matrix A of the DAG G is perturbed by E, resulting in the DAG H and its adjacency matrix B. In Figure 3.6b, G' and E' represent the topological equivalent of G and E at the same scale as G. The adjacency matrix A' of the DAG G' is perturbed by E', resulting in the DAG H' and its adjacency matrix B'. If we assimilate the distortion of a TSV to the Euclidean distance between the original TSV and its perturbed version, the distortion of the W-TSV (6.59) is smaller than the distortion of the scaled TSV (11.04).

3.4.4 Proposed fast computation

The topological signature (TS) of a DAG is based on the magnitude of the eigenvalues of its adjacency matrix. For the TSV, the TS is solely based on the structure of the underlying graph, while for the W-TSV it is based on both the weights and structure of the underlying graph. Nevertheless, the computation of the TS involves a singular value decomposition (SVD) of the adjacency matrix, which has a computational complexity of $O(n^3)$ for an $n \times n$ matrix.



(a) Perturbation of G by E. H: perturbed DAG. A and B: adjacency matrices of G and H.



(b) Perturbation of the DAG G' by E'. H': perturbed DAG. A' and B': adjacency matrices of G' and H'.

Figure 3.6 Comparison of the perturbation of DAG G (scale 10) by E and its topological perturbation at the same scale. (a) Perturbation of G by E. The W-TSVs of G and H are also shown. (b) Perturbation of G' by E', the topological equivalent of G and E at scale 10. The scaled TSVs of G' and H' are shown. The deleted/added edges and vertices are shown respectively in red/green on H and H'. Here, the distortion of the W-TSV (6.59) is smaller than the distortion of the scaled TSV (11.04).

It is possible to evaluate the TS with a computational complexity of $O(n)$ based only on the weights of the DAG and by ignoring its structure. For fast computation, we simply define the TS of a DAG G as the sum of all its weights:

$$\mathrm{TS} = \sum(w_{ij})$$

The fast computation provides the TS with a better interpretation of its value with respect to its weights. This computation is linear with respect to the weights, so it is stable under minor perturbation by a matrix E. As already stated, the cost of the fast computation is the loss of structural information, and the performance of the TSV can be particularly affected by this computation as all its weights are equal. Although the structure of the graph is lost at the TS level, it is retrieved to some extent during construction of the W-TSV.

3.5 Proposed Arabic subword graph representation

In this section, our holistic method for encoding the structure of Arabic subword shapes into a DAG is presented. We chose the DAG representation because it is more expressive than the vector representation, thanks to the relational information it contains. The saliency of an Arabic subword derives from its topology and its geometry, which are highlighted by the shape skeleton. Therefore, relevant pieces of information are extracted from the shape skeleton, giving rise to 3 DAG representations, each of which integrates more information than the previous one. First, we can distinguish three types of points on a skeleton: the end points which only have 1 neighbor, curve points which have 2 neighbors, and branch points which have 3 neighbors, or more. Neighboring curve points can be grouped together and considered as skeletal curves. The end points and branch points provide information about the topology of the shape, while the skeletal curves provide information about its geometry. This is because the skeleton approximates the loci of the center of the pen while the subword is being written. A skeletal curve contains information about the geometry of the shape through its length and curvature:

$$\kappa = \frac{\dot{x}\ddot{y} - \dot{y}\ddot{x}}{(\dot{x}^2 + \dot{y}^2)^{3/2}} \tag{3.3}$$

The most salient parts of a curve are given by the curvature extrema and inflection points. Once the curvature extrema are obtained, an inflection point is inserted between two consecutive extrema if their curvature signs are different.

The first DAG representation is the topological DAG (T-DAG), which only contains information about the shape topology. The end points and branch points of the skeletal graph are set as the vertices of the T-DAG, and each skeletal curve will represent an edge of the DAG, connecting two vertices if they were connected by the skeletal curve in the skeleton image. The second DAG representation is the length DAG (L-DAG), which further integrates information about the skeletal curve lengths, by weighting the edges of the T-DAG by the length of the corresponding skeletal curve. The last DAG representation is the curvature DAG (C-DAG), which contains additional information about the curvature of the skeletal curve. For this, each skeletal curve is split at the position of the extrema and inflection points. The curvature extrema and the inflection points of the skeletal curves are added as additional vertices of the L-DAG, where the weight of the edges is equal to the length of the split curves.

The three graphical representations defined previously are, in fact, undirected graphs. In order to transform them into DAGs, a partial order is defined over the graph vertices. This is done by assigning a formation time to each vertex that is equal to its distance from the nearest end point of the skeleton. The distance between two vertices is defined as the weight of the shortest paths between the vertices, i.e. the sum of the weights of the edges traversed by the shortest path. For this transformation, the length of the corresponding skeletal curves will temporarily be assigned to the T-DAG edges as weight. The distance from each vertex to the end points can be obtained using the Dijkstra algorithm, as this task corresponds to a single-source shortest-path problem on a graph. The following partial ordering is used on the graph vertices:

$$u \leq v : d_u \geq d_v \tag{3.4}$$

where u and v are vertices of the graph and d_u and d_v are their shortest distances from an end point respectively. A path of directed edges between u and v exists only if the partial ordering $u \leq v$ is respected. This ordering puts vertices corresponding to the skeleton's end points as

leaves of the graph, because their nearest end point is themselves, and so the distance is zero. The goal of this ordering is to make the central part of the subword the source, which can be any type of vertex, even an end point, if the graph only contains end points. The process of formation of the subword DAGs from a subword shape is illustrated in Figure 3.7. First, the shape skeleton is computed. Then, the graph's topological vertices are identified on the skeleton. The shape in the example contains two end points and no branch points. The T-DAG and the L-DAG are extracted from this set of vertices and skeletal curves. The curvature-based vertices are also identified from the skeletal curve. In this example, we have two curvature extrema and one inflection point. The C-DAG is extracted from this new set of vertices.

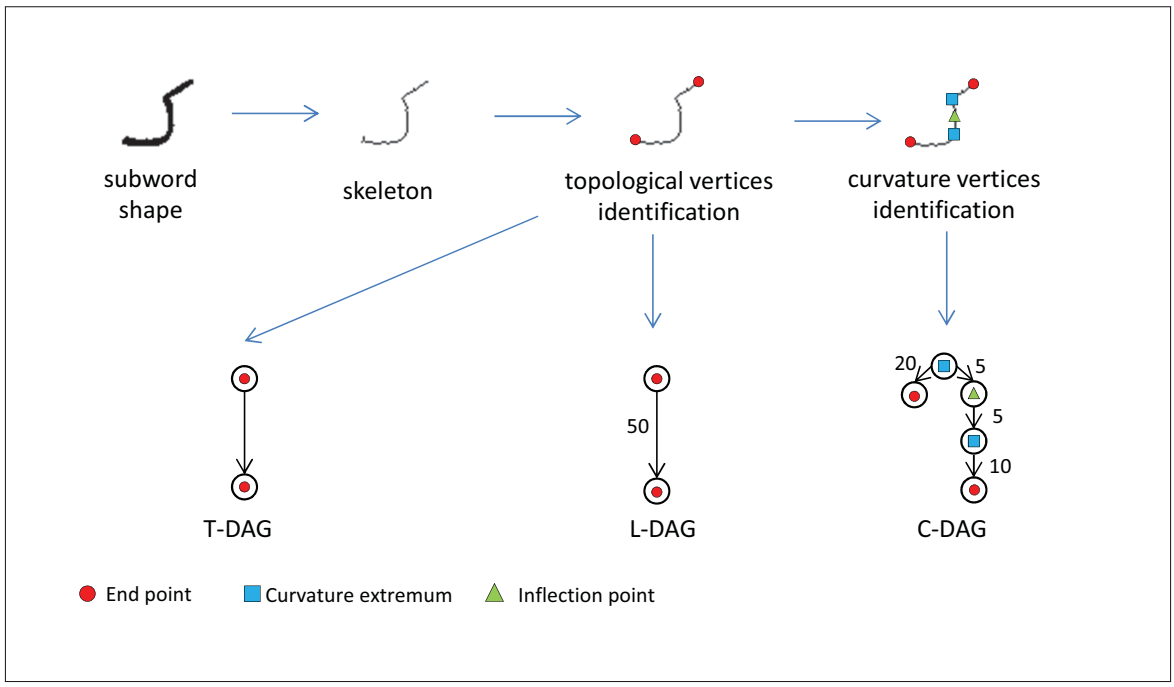


Figure 3.7 Formation of the various subword graphs. Topological DAG (T-DAG), length DAG (L-DAG), curvature DAG (C-DAG).

3.6 Experiments

3.6.1 Databases

We evaluated this approach on the Ibn Sina database (Farrahi Moghaddam *et al.*, 2010) for ancient Arabic documents and the IFN/ENIT database (Pechwitz *et al.*, 2002) for modern Arabic documents. The Ibn Sina database is based on a commentary on an important philosophical work by the famous Persian scholar Ibn Sina. This database consists of 60 pages and approximately 25,000 Arabic subword shapes written in the Naskh style (Figure 3.3b). The document images were binarized with a dedicated algorithm (Farrahi Moghaddam and Cheriet, 2010) to preserve the shape's topology. Each page contains approximately 500 subword shapes. There are 1,200 different classes, but the distribution of the database is highly unbalanced; some classes have up to 5,000 entries, while others have fewer than 5. The diacritics are ignored, and a W-TSV is assigned to each subword shape. The W-TSV size is set to $p = 3$ as most of the skeletal points have at most 3 neighbors.

The IFN/ENIT database was built for a postal application, and contains the names of 946 Tunisian towns and villages spread over 26,459 word images. Approximately four hundred writers participated in its creation. For each connected component of a word shape (subwords and diacritics), a W-TSV of size $p = 1$ is computed. Then all these individual W-TSVs are sorted in descending order and concatenated, in order to form the word shape W-TSV (size $p = 10$). The size of the W-TSV is set according to the maximum number of subwords in a word.

3.6.2 Experimental protocol

The W-TSV is extracted in the following way from a single connected component shape image. First the skeletal graph of the shape is obtained using the divergence ordered thinning algorithm (Dimitrov *et al.*, 2000), with the threshold parameter, which is used to discard irrelevant skeletal branches, set to -7. In order to prevent the formation of loops in the DAG, the holes of the shape are filled in prior to this step. The fork points of the graph are merged

into a single point once the graph is extracted. The curvature extrema points are found using the algorithm described by He and Yung (2008), and the extrema near the ends of the skeletal curve (distance less than 5 pixels) are ignored. If a shape's DAG contains more than one source, the W-TSV is computed for each source, and all the W-TSVs are added to form the final shape's W-TSV. For simplicity, the curve length is set to its number of pixels; for an 8-connected curve, it corresponds to the L_∞ metric. With the 3 DAG representations and the fast and classical computation of the W-TSV, 6 different W-TSVs are evaluated (fast TSV, fast L-TSV, fast C-TSV, TSV, L-TSV, C-TSV).

Some examples from the Ibn Sina database of archigraphemic subwords C-DAG and their fast C-TSV are shown in Figure 3.8. For each shape, the skeleton image is labeled by the C-DAG graph-vertex index. The vertices of the C-DAG are labeled by two numbers. The first number represents the index of the vertex in the C-DAG, and the second number after the colon represents the point type. The meaning of the point type value and its corresponding color on the skeleton image is detailed in Table 3.1. Notice that the C-DAG and the fast C-TSV of the subword shapes are quite different.

Table 3.1 Value and color code of the vertex types

Vertex type	Value	Color
End point	1	red
Branch point	3	yellow
Curvature	10	blue
Inflection	11	green

The lexicon-reduction method is evaluated on the degree of reduction of the shape database, as well as on the degree of reduction of the lexicon, over the entire query database, achieved for a given accuracy of reduction. A leave-one-out strategy is used for the evaluation; each shape is selected alternately as the query shape and the remaining shapes are considered as constituing the shape database. The results are averaged over the entire database. For the Ibn Sina database, only the first 50 pages are used for this experiment.

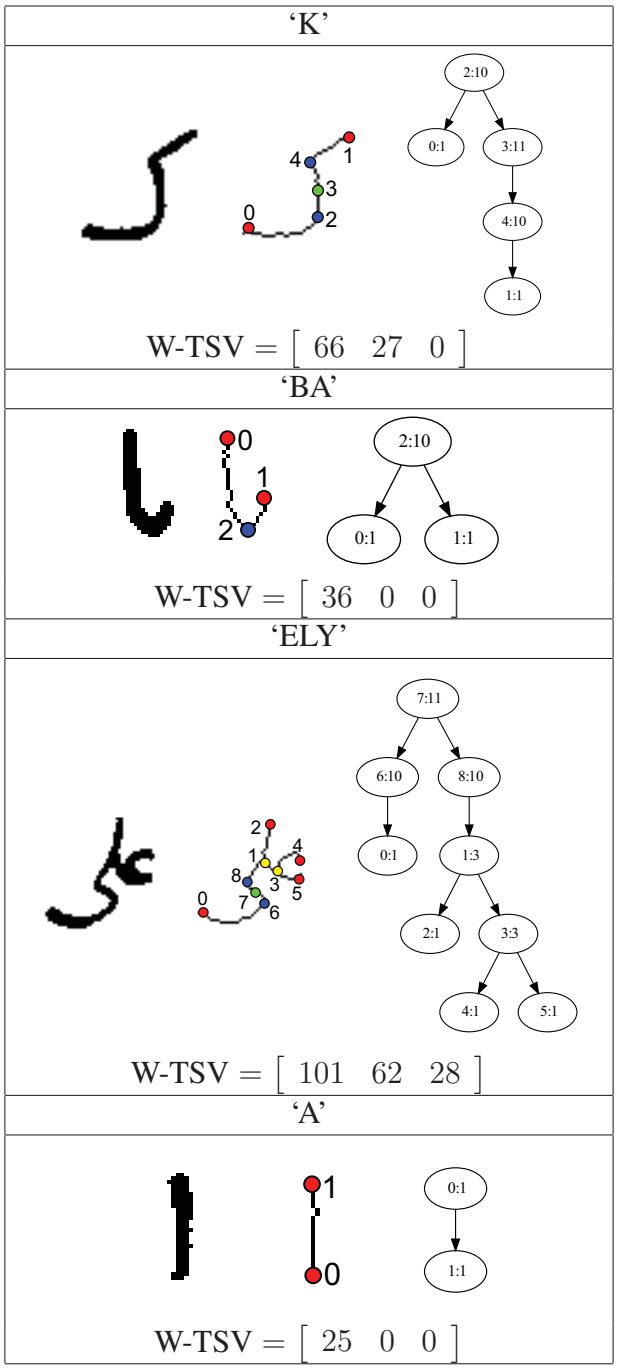


Figure 3.8 Arabic archigraphemic subword skeletal graphs, C-DAG (edges weights not shown) and fast C-TSV.

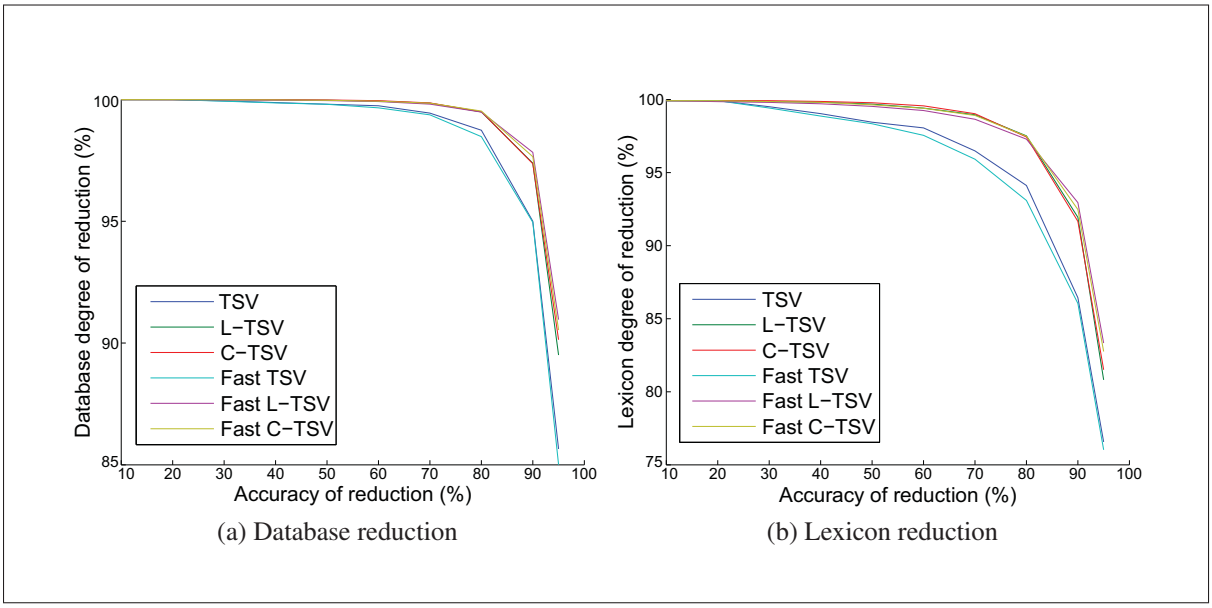
3.6.3 Results and discussion

The lexicon reduction performance on the Ibn Sina and IFN/ENIT databases is shown in Figure 3.9 and Figure 3.10. The trend of the curves is the same for both databases, but also for the shape database reduction, as is the case for the degree of reduction of the lexicon. The performances of the W-TSVs, including geometrical information (L-TSV and C-TSV), are very similar, and better than the performances of the pure topological TSVs, as they achieve a higher degree of reduction for a given accuracy of reduction. Detailed results for specific accuracies of reduction are shown in Table 3.2 and Table 3.3. On the Ibn Sina database, the best performance is achieved by the fast L-TSV, with a database degree of reduction $\rho = 90.96\%$ and lexicon degree of reduction $\rho = 83.33\%$ for an accuracy of reduction of 95%. On the IFN/ENIT database, the best performance is achieved by the fast C-TSV with a database degree of reduction $\rho = 94.97\%$ and lexicon degree of reduction $\rho = 71.33\%$ for an accuracy of reduction of 95%.

On a 2.30 GHz processor and for fully preprocessed shapes, the lexicon reduction time for each query shape against the lexicon database is approximately 7.5 milliseconds for the Ibn Sina database and 10 milliseconds for the IFN/ENIT database. The preprocessing time on the Ibn Sina database is, on average, 2.5 milliseconds, and 53 milliseconds on the IFN/ENIT database.

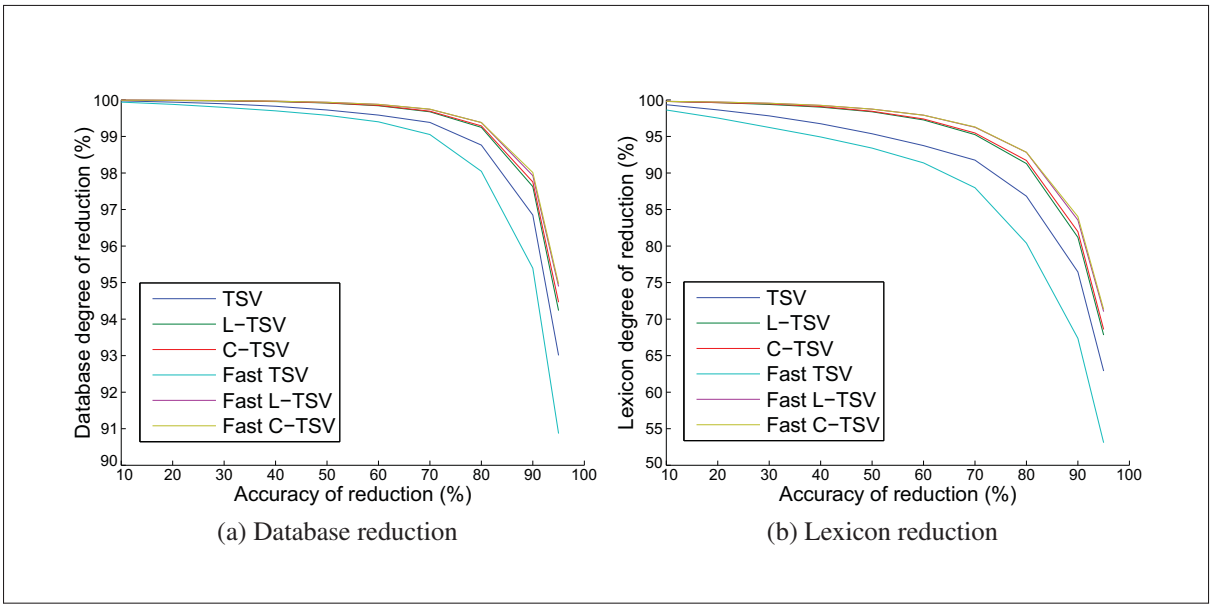
Table 3.2 Lexicon reduction performance on the Ibn Sina database

W-TSV type	Accuracy of reduction			
	α = 90%		α = 95%	
	Database ρ (%)	Lexicon ρ (%)	Database ρ (%)	Lexicon ρ (%)
TSV	95.00	86.43	85.65	76.57
L-TSV	97.38	91.94	89.51	80.81
C-TSV	97.36	91.66	90.14	81.51
Fast TSV	94.96	86.05	85.00	76.03
Fast L-TSV	97.83	92.94	90.96	83.33
Fast C-TSV	97.65	92.47	90.53	82.76



(a) Database reduction (b) Lexicon reduction

Figure 3.9 Lexicon reduction performance for different accuracies of reduction on the Ibn Sina database.



(a) Database reduction (b) Lexicon reduction

Figure 3.10 Lexicon reduction performance for different accuracies of reduction on the IFN/ENIT database.

The W-TSV approach shows better performance for Arabic documents than the classical TSV, both for database pruning and vocabulary reduction. Indeed, most of the subwords share the

Table 3.3 Lexicon reduction performance on the IFN/ENIT database

W-TSV type	Accuracy of reduction			
	α = 90%		α = 95%	
	Database ρ (%)	Lexicon ρ (%)	Database ρ (%)	Lexicon ρ (%)
TSV	96.85	76.45	93.01	62.89
L-TSV	97.63	81.19	94.23	67.82
C-TSV	97.75	81.97	94.46	68.58
Fast TSV	95.39	67.36	90.87	53.08
Fast L-TSV	97.93	83.56	94.89	71.02
Fast C-TSV	98.01	84.03	94.97	71.33

same topology, despite having different shapes. Geometrical information is thus needed to improve the discriminative power of the TSV. For W-TSVs including geometrical information, the fast computation show slightly better results than the classical computation, showing the importance of the weights over the structure of the DAG for Arabic word databases. As expected, the fast computation decrease the performances of the TSV. It can be noted that the performance of the L-TSV and C-TSV are very similar. The curvature feature, which modify the structure of the DAG, doesn't significantly improve the W-TSV performance once the DAG is weighted by the length of the curves. This further shows the importance of the length feature over the structure of Arabic subwords. The main source of error is the variability in the appearance of the word/subword, either because of the writing variations allowed by the Arabic script style or because of the large panel of writers.

The impact of lexicon reduction on a 1-NN archigraphemic subword shape classifier (3.9) has been tested on the Ibn Sina database. The first 50 pages form the shape reference database, and the last 10 pages form the test database. The lexicon was reduced using the fast L-TSV representation, and by keeping the i nearest shapes needed to achieve a given accuracy of reduction (on average, over the cross validation) in the previous experiment. The results, detailed in Table 3.4, show that the decrease in the recognition rate is in the same order as the decrease in the accuracy of reduction, which means that the decrease in the recognition rate is effectively controlled by the accuracy of reduction.

Table 3.4 Impact of lexicon reduction on the archigraphemic subword shape classifier

Accuracy of reduction α (%)	Classifier recognition rate (%)	Reduced database size i
100	86.23	20681
95	84.57	1869
90	79.07	449

3.6.4 Comparison with other methods

The proposed method has been compared to existing approaches for Arabic script. These approaches first reduce the lexicon based on the subword counts, and then use a dot descriptor string. On the Ibn Sina database, only the dot descriptor is used, as the recognition is performed at the subword level. First, the dot string matching is evaluated, under the assumption that the dot descriptor extraction from the subword images has an ideal behavior. This experiment is referred to as *ideal diacritic matching* and will provide upper bound results for the dot-based approaches. Then, a rule based method, similar to that of Mozaffari *et al.* (2008a) was used to extract the dot descriptor from the images. Single, double and triple dots are detected and represented by a two-character label representing the number of dots and their positions (up or down) with respect to their base shape. Finally, all the labels are concatenated into a string. The lexicon is reduced based on the string-edit distance from the ideal dot descriptors of the lexicon. The edit cost is 1 for each missing/additional dot, and the value 2 is added to the cost in case of position mismatch. For an ease of comparison, the reduction efficacy measure $\eta = \alpha^k \cdot \rho$ is also used, with $k = 1$, in order to give equal importance to α and ρ. The results on the Ibn Sina database are shown in Table 3.5. The proposed method performs better than the dot-based approach, even for the ideal matching. This result shows the low discriminative power of the dot descriptor at the subword level. This is because most of the subwords have only one diacritical mark, or none at all. On the IFN/ENIT database, the performance of the proposed method is in the range of the other approaches (Table 3.6). The W-TSV approach uses the subword shape and the subword count in each word, as each connected component represents an element of the word W-TSV. The W-TSV approach is therefore complementary to the dot descriptor approach, and a combination of the two would improve the results. In-

spite of the lower performance of the W-TSV than the best method on IFN/ENIT, it has some advantages. First, no a priori knowledge is needed, while the other approaches must perform the identification of subwords and the recognition of diacritics. Also, for lexicon reduction, it has a computational complexity of the order of $O(N)$, where N is the length of the W-TSV vector, while the dot-based approaches have a complexity of the order of $O(M.N)$ due to the string-edit distance, where M and N are the lengths of the strings.

Table 3.5 Comparison with a dot matching lexicon-reduction method on the Ibn Sina database

Method	α (%)	ρ (%)	η (%)
Ideal diacritics matching	100	74.96	74.96
Diacritics matching	75.38	72.88	54.94
Proposed method (Fast L-TSV)	90.0	92.94	83.64

Table 3.6 Comparison with other lexicon-reduction methods on the IFN/ENIT database

Method	α (%)	ρ (%)	η (%)
Subword count and diacritics matching (Mozaffari *et al.*, 2008a)	74	92.5	68.5
Improved subword count and diacritics matching (Wshah *et al.*, 2010)	94.6	85.6	81.0
Proposed method (Fast L-TSV)	90.0	83.6	75.2

3.7 Conclusion

In this paper, we proposed the W-TSV representation, a generalization of the TSV for weighted DAG indexing. The stability and robustness to small weights perturbation of the W-TSV have been studied. The W-TSV has been applied for holistic lexicon reduction of handwritten Arabic words/subwords. The topology and the geometry of the word/subword shape is first converted into a DAG and then transformed into a low dimensional vector using the W-TSV representation. Three different DAG representations and a fast W-TSV computation approach have been proposed. The W-TSV has shown better performances than the original TSV. This approach is complementary to the dot based lexicon reduction approaches for Arabic documents. The

processing speed of this approach can be further improved by parallelizing the thinning algorithm and the nearest neighbors search. In future work, this approach will be extended to other scripts such as Chinese, the main challenge being to properly encode the shape loops into the DAG representation. The proposed DAG representations are invariant to shape rotation, and so directional information will be added to improve performance. Moreover, the combination of the W-TSV representation and other shape representations, such as geometrical moments, will be explored.

3.8 Acknowledgments

The authors thank the NSERC and SSHRC of Canada for their financial support.

3.9 Appendix - Archigraphemic subword shape classifier

The archigraphemic subword shape classifier is a holistic classifier, based on a nearest-neighbor strategy (1-NN). A contour-based representation is chosen for its complementarity with the skeleton representation used for lexicon reduction. The subword contour is represented using the square root velocity (SRV) representation (Joshi *et al.*, 2007; Srivastava *et al.*, 2011), where the contour is considered as a simple (non self-intersecting) closed curve. The curve is defined on the $\mathbb{L}^2$ Hilbert space, and has value in the $\mathbb{R}^2$ Euclidean space. This representation allows shape matching, while being invariant to translation and scaling by embedding the contour curve of the shapes on an appropriate manifold. The curve f is parameterized by t over the domain $D = [0, 1]$. First, f is normalized to unit length, in order to remove the effect of scale. The curve is then represented using the SRV representation:

$$q(t) = \dot{f}(t) / \sqrt{\left\| \dot{f}(t) \right\|} \tag{3.5}$$

This representation is invariant to translation as it uses the derivation of f. It also preserves the unit length constraint on f:

$$\int_D \|q(t)\|^2 dt = \int_D \left\| \dot{f}(t) \right\| dt = 1 \tag{3.6}$$

Therefore, the set of all curves under the SRV representation forms a unit hypersphere in $\mathbb{L}^2$. Furthermore, the original curve f can be recovered up to translation from q:

$$f(t) = \int_0^t q(s) \|q(s)\| ds \tag{3.7}$$

The geodesic distance between two curves q_1 and q_2 is defined as $d(q_1, q_2) = acos(\langle q_1, q_2 \rangle)$. The best curve alignment is sought, in order to decrease the influence of handwriting variability on the recognition process. As the contour curves are closed, the best origin of the curve parameterization is found first, and then the curves are aligned using dynamic programming. After lexicon reduction, only the shapes contained in the reduced lexicon are considered by the 1-NN classifier. Other values of k have been tested for this k-NN classifier, but without significant improvement.

CHAPTER 4

ARTICLE II - ARABIC WORD DESCRIPTOR FOR HANDWRITTEN WORD INDEXING AND LEXICON REDUCTION

Youssouf Chherawala and Mohamed Cheriet

Synchromedia Laboratory, École de Technologie Supérieure
1100 Notre-Dame Ouest, Montréal, QC, Canada

Submitted to Elsevier Pattern Recognition

Abstract

Word recognition systems use a lexicon to guide the recognition process in order to improve the recognition rate. However, as the lexicon grows, the computation time increases. In this paper, we present the Arabic word descriptor (AWD) for Arabic word shape indexing and lexicon reduction in handwritten documents. It is formed in two stages. First, the structural descriptor (SD) is computed for each connected component (CC) of the word image. It describes the CC shape using the bag–of–words model, where each visual word represents a different local shape structure, extracted from the image with filters of different patterns and scales. Then, the AWD is formed by sorting and normalizing the SDs. This emphasizes the symbolic features of Arabic words, such as subwords and diacritics, without performing layout segmentation. In the context of lexicon reduction, the AWD is used to index a reference database. Given a query image, the reduced lexicon is obtained from the labels of the first entries in the indexed database. This framework has been tested on Arabic word databases. It has a low computational overhead, while providing a compact descriptor, with state–of–the–art results for lexicon reduction on the Ibn Sina and IFN/ENIT databases.

Keywords

Arabic word descriptor, Shape indexing, Holistic representation, Lexicon reduction, Arabic handwritten documents, IFN/ENIT, Ibn Sina database

4.1 Introduction

Arabic word recognition is an active field of research (Lorigo and Govindaraju, 2006; Al-Hajj Mohamad *et al.*, 2009; Giménez and Juan, 2009; Märgner and El Abed, 2011; Slimane *et al.*, 2011; Dreuw *et al.*, 2012). Most word recognition systems (WRS) use a lexicon, which is made up of a set of accepted words, to limit their output to valid words. The recognition rate is improved by testing all the lexicon word hypotheses, although this is achieved at the expense of a loss of recognition speed. A processing time as long as 4 seconds for a single word (Märgner and El Abed, 2009; Märgner and El Abed, 2010), even in competitive Arabic WRS, is not acceptable in an industrial context. Lexicon reduction methods have been developed to alleviate this problem, which dynamically reduce the lexicon based on the input images. Unfortunately, the reduction process is prone to error, in that it may discard the true label of an input image. If this happens, not only does the accuracy decrease, but the WRS won't recover the true label. The sources of error are the same as those for word classifiers, which are affected by the handwriting variability (Park, 2002) of individuals, and even of a single individual, and the level of degradation of the documents, which is typically high in historical texts (Hedjam *et al.*, 2011). Lexicon reduction methods must manage the difficult trade–off between reducing the size of a lexicon and maintaining a high level of accuracy on the retained word hypotheses. In other words, these methods must improve the WRS processing speed without decreasing its recognition rate. In addition, a successful lexicon reduction system must be efficient to compute, in order to minimize its impact on the WRS processing speed, and it should capture discriminative lexicon word shape features to provide good performance.

Unlike Latin script, Arabic script is written from right to left, and the alphabet is composed of 28 letters instead of 26 (Figure 4.1). The shape of the letters is dependent on their position in the word, and is usually different if they are at the beginning, middle, or end of a word. Six letters ('ﺩ', 'D', 'Ḏ', 'R', 'Z', and 'W') can be connected only if they appear in a final position; if they appear in initial or medial position, a space is inserted after them and the word is broken into subwords. Several letters share the same base shape and are only distinguishable

by diacritics in the form of one, two, or three dots appearing above or below the shape. The features of Arabic words are illustrated in Figure 4.2.

ص	ش	س	ز	ر	ذ	د	خ	ح	ج	ث	ت	ب	ا
Ṣ	Š	S	Z	R	Ḏ	D	Ḫ	Ḥ	Ǧ	Ṯ	T	B	ʾ
ي	و	ه	ن	م	ل	ك	ق	ف	غ	ع	ظ	ط	ض
Y	W	H	N	M	L	K	Q	F	Ġ	ʿ	Ẓ	Ṭ	Ḍ

Figure 4.1 Arabic letters with their ISO 233 transliteration.

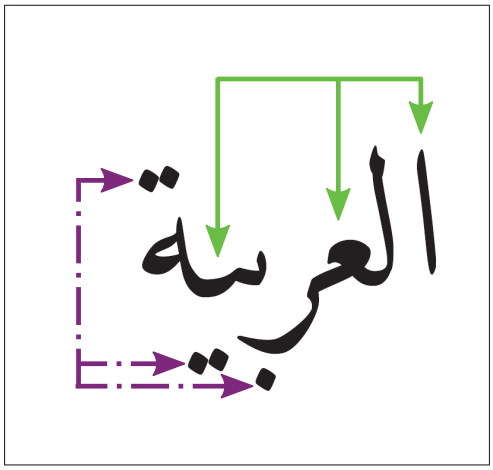


Figure 4.2 An Arabic word with its subwords (solid lines) and diacritics (dashed lines).

The problem of lexicon reduction was initially investigated for Latin script. The simplest method is based on the length of the word, as it allows discrimination between short and long words. The most common feature extracted from a word image is a sequence of ascenders and descenders (Carbonnel and Anquetil, 2004). The sequence is matched against features extracted from synthetic images of words in the lexicon, using regular expressions (Palla *et al.*, 2004) or the string edit distance (Madhvanath *et al.*, 2001). Lexicon reduction is then performed by discarding the unmatched lexicon entries. More advanced features are often used in combination with an analytic classifier. Zimmermann and Mao (1999) form a regular expression from key characters, which represent an unambiguous recognition of a character-level

classifier. Bertolami *et al.* (2008) propose a HMM based on shape code models, where each shape code represents multiple letters. A list of regular expressions is then obtained from the top ranked shape code sequences of the HMM.

Research on lexicon reduction has been given new impetus in recent years with the increasing interest in Arabic script (Mozaffari *et al.*, 2007). Novel methods are being built, based on the specificities of Arabic words, and they can be classified in two groups. One group of methods considers only the diacritic information and subword counts, ignoring the subword shape. Mozaffari *et al.* (2008a) proposed the first of these methods, in which the lexicon is pruned based on the estimated number of subwords, and then the diacritics are categorized according to their type (1, 2, or 3 dots) and their positions relative to the base shape (above or below); finally, a sequence of diacritics is formed and matched against synthetic models of the remaining lexicon words. The diacritic categorization step has since been improved by Wshah *et al.* (2010), thanks to a better estimation of their positions and the use of a convolutional neural network to recognize their type. The other group of methods considers the subword shape, and are based on the skeleton image. Chherawala and Cheriet (2012a) propose a spectral method for indexing skeleton shapes, where the skeleton is modeled as a weighted graph using topological and geometrical features. Lexicon reduction is then performed by indexing a reference database of subword shapes and selecting the labels of the top ranked database entries. Asi *et al.* (2012) propose a hierarchical organization of subword skeleton shapes, where the bottom layer represents the original shapes and the top layer their coarse representations. The shapes of a given layer are simplified and then clustered to form the next level. Given a query shape, the lexicon is reduced by traversing the hierarchy in top–down fashion and by skipping the less promising clusters.

In this paper, we propose to represent the shape of Arabic words using the Arabic word descriptor (AWD). It encodes the shape of the image connected components (CCs) while emphasing the symbolic features of Arabic words, such as subwords and diacritics.

A structural descriptor (SD) is used to encode the shape of each CC, based on the bag–of–words (BOW) model (Yang *et al.*, 2007), which has been successful for image retrieval and classifica-

tion (Lazebnik *et al.*, 2006; Quelhas *et al.*, 2007; Wu and Hoi, 2011; Zhou *et al.*, 2013), as well as for shape matching (Mori *et al.*, 2005). A set of pattern filters representing different patterns at different scales is used to extract local features, called pixel descriptor (PD), for each point of the CC skeleton image. The PDs are assigned to their nearest visual word from a predefined codebook of the feature space. The SD is then formed as a histogram representing the number of occurrences of each visual word. The SD is well suited for lexicon reduction, because it allows efficient shape matching by vector comparison. Finally, the AWD is formed by sorting and normalizing the SDs of all the CCs. It incorporates information about the shape and count of the subwords and diacritics into a single vector, without performing any word layout analysis. In the context of lexicon reduction, the AWD is used to index a reference database of word shapes. The labels of the top ranked database entries form the reduced lexicon. We show the AWD's high performance for lexicon reduction with low computational overhead.

This paper is an extension of the work published by Chherawala *et al.* (2012). In particular, the extension of the methodology includes a larger set of filters for image feature extraction. The experimental evaluation has also been significantly improved, by combining lexicon reduction with word recognition tasks.

The rest of this paper is organized as follows: Section 4.2 explains the concept of the pixel descriptor. Section 4.3 describes the formation of the structural descriptor. Section 4.4 explains the formation of the Arabic word descriptor. Section 4.5 gives an overview of the lexicon reduction system. Section 4.6 presents our experimental results.

4.2 Pixel descriptor

The pixel descriptor (PD) is a feature vector which describes the local shape structure. It is computed on the skeleton image, which highlights the shape structure. Note that only the skeleton pixels are considered, as they provide the most information on the shape of the word. The PD is formed from the output of various image filters, called pattern filters. We first describe the pattern filters and PD formation, and then we provide a structural interpretation of the PD.

4.2.1 Pattern filters and pixel descriptor formation

Pattern filters are designed to detect specific structural patterns at a given scale around each skeleton pixel of the skeleton image. We assume that the skeleton image I is binary, having skeleton pixels with a value of 1 and background pixels with a value of 0. For computational efficiency, we have chosen rectangular filters, because they can be efficiently computed using the integral image (Viola and Jones, 2004). We define a family of five patterns to describe the local structure of skeleton images. The patterns comprise a square and four lines of orientation 0, 45, 90, and 135 degrees (Figure 4.3). The square filter is the most isotropic, given the rectangular filter constraint. All the filters are square windows of width w, which also represents their scales, with masked areas to form the pattern. The square pattern have no mask, the 0° and 90° lines are masked by two rectangles of size $w \times w/4$, while the 45° and 135° lines are masked by two squares of size $w/2 \times w/2$. The patterns are similar to the Haar-like features, the difference is that the value of masked area are ignored instead of being substracted.

Each pattern filter defines a specific neighborhood around the skeleton pixel and it counts the number of skeleton pixels falling inside their patterns. The output of the filter is normalized by the filter scale. Considering the filter as an image, the value of the pixels of the pattern area is 1, and the value of the pixels of the masked area is 0. The output f of a filter F of scale w at a pixel of position (x, y) of the skeleton image I is given by:

$$f = \frac{1}{w} \sum_{0 \leq i,j < w} F(i,j) \cdot I\left(x + i - \lfloor \frac{w}{2} \rfloor, y + j - \lfloor \frac{w}{2} \rfloor\right)$$

where $F(i,j)$ and $I(i,j)$ are the values of F and I at the position (i,j), and $\lfloor x \rfloor$ represents the floor of x. The values outside the bounds of I are considered to be 0.

The PD is then formed from the concatenation of the output of n pattern filters $\mathrm{PD} = [f_1 \ldots f_i \ldots f_n]^T$, where f_i is the output of the filter F_i. All the filters F_i are unique, and are differentiated either by their patterns or by their scales. Each filter provides a differ-

ent insight into the pixel neighborhood. The PD is therefore a signature of the local structure surrounding the pixel.

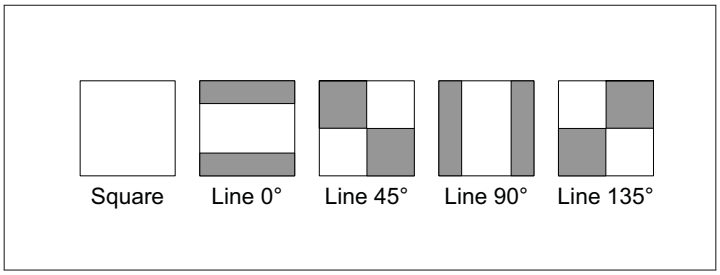


Figure 4.3 Pattern filters. The gray areas are masked.

4.2.2 Structural interpretation

The outputs of the pattern filters composing the PD provide a geometrical and topological interpretation of the skeleton pixels (Figure 4.4). When the response of a line filter is close to 1 for a given skeleton pixel, a local skeleton curve has the same orientation as the filter. The case of the square filter is more interesting. A response close to 1 indicates that the skeleton pixel belongs to a simple curve structure (curve with no self intersection), while responses that are significantly smaller or bigger are indicators of pixels in the neighborhood of end points and branch points respectively. The square filter is therefore an indicator of the local skeleton topology. All these considerations only hold on the condition that the filter scale is small enough to not be perturbed by spatially close structures.

4.3 Structural descriptor

The structural descriptor (SD) is a feature vector describing word shapes. It is based on the BOW model, which represents the distribution of image features extracted at selected keypoints. The skeleton shape image is considered to highlight the shape topology and geometry. All the skeleton pixels are considered as keypoints, as it has been shown that dense sampling provides better results for lexicon reduction (Chherawala *et al.*, 2011). Given a set of pattern filters, a set of PDs $\{\mathrm{PD}_1 \ldots \mathrm{PD}_n\}$ is extracted from the skeleton image, where n is the num-

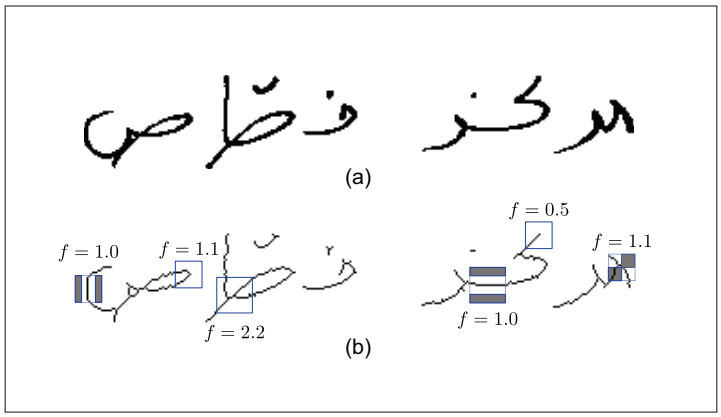


Figure 4.4 Response of pattern filters. (a) Original word shape. (b) Response of various pattern filters on the skeleton image.

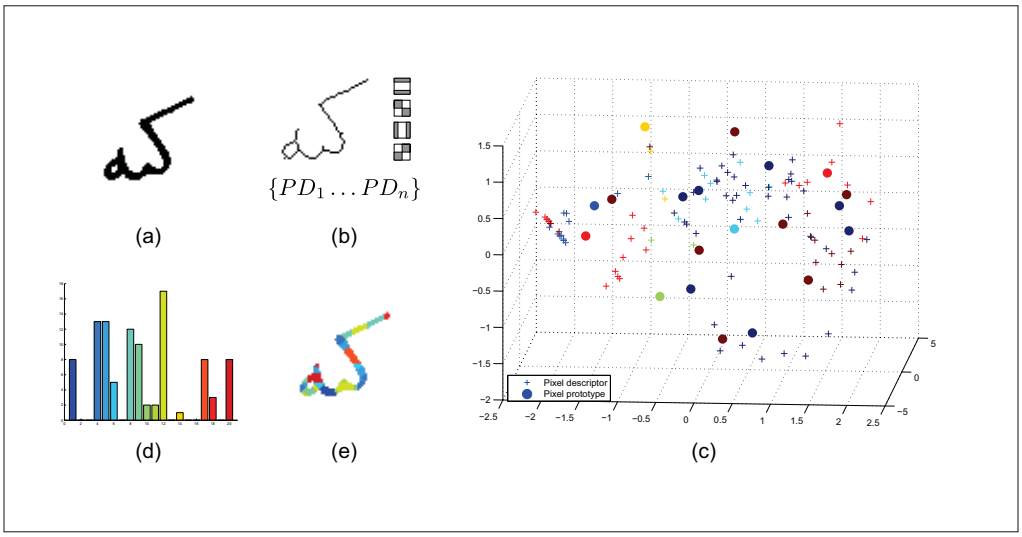


Figure 4.5 Formation of the structural descriptor. (a) Shape image. (b) Set of extracted pixel descriptors, given the skeleton image and a set of pattern filters. (c) Assignment of each pixel descriptor to the visual word of its nearest pixel prototype. (d) Structural descriptor: histogram of the occurrence of visual words. (e) Illustration of the structure encoded by each visual word on the original shape, the shape pixels are shown with the color of their pixel prototypes (for clarity, the original shape image is shown, instead of the skeleton image).

ber of skeleton pixels in the image. In order to build the codebook, the entries of the PDs are normalized to zero mean and unit variance across the reference database. The PD is quantized using the k-means algorithm, which outputs k pixel prototypes, representing the skeleton im-

age visual words. The SD of a given skeleton image is built by first assigning each of its PD to the visual word of its nearest prototype and then forming a histogram from the number of occurrences of each visual word in the image. The process of the formation of the SD is shown in Figure 4.5 and summarized in Algorithm 1.

We consider that the SDs are embedded in the vector space $\mathbb{R}^k$. Using the Euclidean metric, the complexity for computing the distance between two SDs is $O(k)$. Given two structural descriptors SD_1 and SD_2, representing two different shapes, where SD(i) represents the i^{th} entry of the descriptor, the distance between the two SDs is

$$\sqrt{\sum_{i=1}^{k} \left(\mathrm{SD}_1(i) - \mathrm{SD}_2(i)\right)^2}$$

The distance is commensurate with the number of unmatched visual words. This approach is therefore similar to the pairing of similar substructures between two skeleton images. It is adapted to the description of Arabic word shapes, as the vector quantization provides some tolerance to handwriting variability.

Algorithm 1 Structural descriptor computation

Input: shape image; pattern filter set E; database pixel descriptor statistics (mean and standard deviation); k-means prototypes $\{P_i\}$
Output: Structural descriptor
Compute the shape skeleton
Compute the pixel descriptor for each skeleton pixel using E
Normalize the pixel descriptors with the database statistics and assign them the visual word of their nearest P_i
Form the SD as the histogram of the visual words in the skeleton image

4.4 Arabic Word Descriptor

The SD is holistic, which means that it considers the image as a whole. This approach fails to incorporate symbolic information related to the various units forming the Arabic words, i.e. the subwords and the diacritics. In this section, we adapt the SD to the Arabic word descriptor

(AWD), which further integrates information on the subword counts and diacritics. We assume that the subwords and diacritics correspond to CCs of the image. First, the SD of each CC is computed. Then, like the idea introduced by Chherawala and Cheriet (2012a) that was never developed, the SDs are sorted in descending order with respect to the number of pixels in their respective CC skeleton. This ordering is expected to rank the largest subwords first and the diacritics last. The sorted descriptors are then concatenated into the Arabic word descriptor $\mathrm{AWD} = [\mathrm{SD}_1 \ldots \mathrm{SD}_c]^T$, where c is the number of CCs in the image and $\{\mathrm{SD}_i\}$ are the sorted CC descriptors ($1 \leq i \leq c$) – see Figure 4.6 for an illustration. This ordering has three main advantages:

- It avoids the difficult task of explicit classification of the CCs into diacritics or subwords, as confusion arises with single letter subwords (Wshah *et al.*, 2010).
- It avoids spatial ordering of the CCs, which is also a difficult problem because Arabic subwords can overlap each other horizontally, and the vertical ordering for the diacritics is based on the estimation of the baseline, which is a problem in itself (Pechwitz and Märgner, 2002).
- It is more tolerant to changes in topology, such as touching, broken or missing CCs like diacritics, as in most cases these modifications have a relatively small impact on the number of pixels in the original CC.

As the SDs are sorted, the first entries of the AWD will be more prominent than the last entries. In order to give equal importance to the subwords and the diacritics, all the AWD entries are normalized to have zero mean and unit standard deviation. The AWD size is set to contain m SDs. If the number of CCs in an image is smaller than m, the AWD is padded with zeros (absence of CCs). Otherwise, it is truncated.

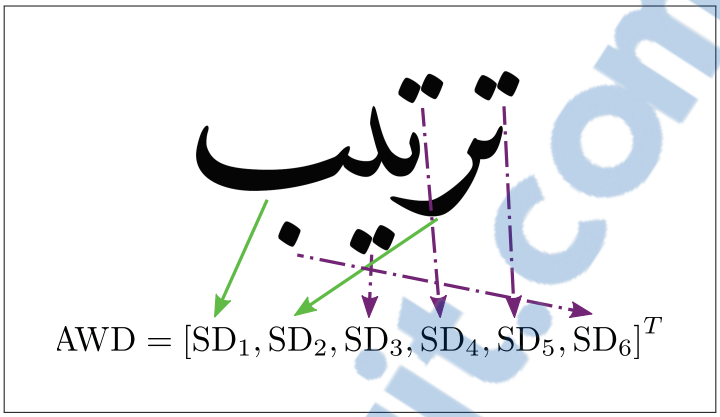


Figure 4.6 Construction of the Arabic word descriptor (AWD) – see text for details.

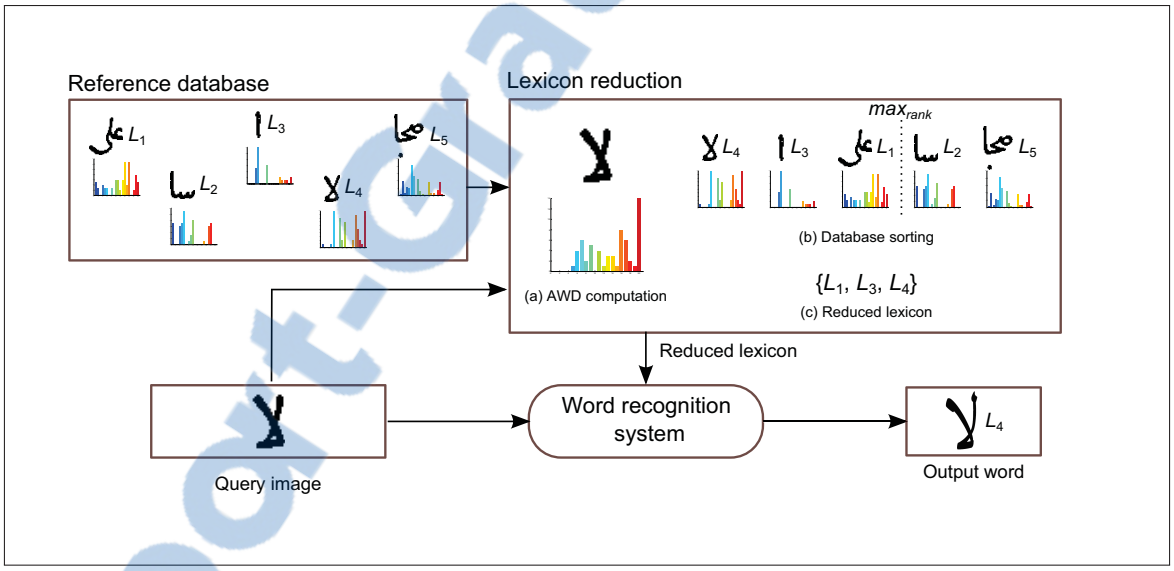


Figure 4.7 Lexicon reduction system overview.

4.5 Lexicon reduction system

4.5.1 System overview

The lexicon reduction system is based on shape indexing. A reference database is composed of word shape images with their corresponding labels L_i. The set of labels contained in the database forms the application lexicon, and so each lexicon word must be represented by at least one image. The more images there are per lexicon word, the better the modeling of

handwriting variability. This database is processed by computing the AWD for each of its images, given a set of pattern filters for local feature extraction. During the lexicon reduction phase, the system takes a word image segmented from the original document as input. The AWD of the query word is first computed, and then it is compared to the AWDs in the reference database in the AWD vector space. The reference database entries are then sorted in ascending order, according to their distance from the query word AWD. The reduced lexicon is finally obtained by considering the labels of the first max_{rank} entries of the sorted database, where max_{rank} is a parameter provided to the system. The reduced lexicon is then fed to the word recognition system. This lexicon-reduction system is illustrated with images segmented at the subword level in Figure 4.7.

4.5.2 Performance measure

When a query word is submitted to a lexicon-reduction system, two criteria are important to assess the performance of the system. The first is accuracy, with value 1 if the reduced lexicon contains the true label of the query word, otherwise 0, in which case the WRS is bound to fail. The second is lexicon size reduction, which is expressed as $1 - R/L$, where L is the size of the original lexicon and R is the size of the reduced lexicon. If we consider accuracy and reduction as random variables over a test dataset, their expected values are noted as the accuracy of reduction α and the degree of reduction ρ respectively. A system with an accuracy of reduction and a degree of reduction that are both close to 1 achieves good performance. However, it is difficult to optimize α and ρ at the same time, as a high degree of reduction increases the chances that the true label will be discarded. The reduction efficacy $\eta = \alpha \cdot \rho$ is also used as a unified measure. In this case, a lexicon-reduction system is evaluated using α, ρ, and η (Madhvanath *et al.*, 2001).

4.6 Experiments

4.6.1 Databases

We evaluate our approach on two Arabic word databases. The first is the Ibn Sina database (Farrahi Moghaddam *et al.*, 2010), which is based on a commentary on an important philosophical work by the Persian scholar Ibn Sina (Figure 4.8). It contains 60 pages from a manuscript copied by a single writer, and is labeled at the subword level. This represents approximately 25,000 subword images and 1200 different classes using archigrapheme label encoding (Cherawala and Cheriet, 2012a), which ignores diacritic information. The first 50 pages are used for the evaluation of our lexicon reduction system. The second is the IFN/ENIT database (Pechwitz *et al.*, 2002), which contains the names of Tunisian cities and villages (Figure 4.9). Approximately 400 writers participated in its creation. It is labeled at the word level, and contains 26,459 word images representing 946 classes. It is composed of five sets (A, B, C, D, and E), the first four being used for the evaluation of our lexicon reduction system. As the image resolution is high in this database, it has been decreased by 2 to improve the processing speed. Also it brings both databases approximately to the same scale.

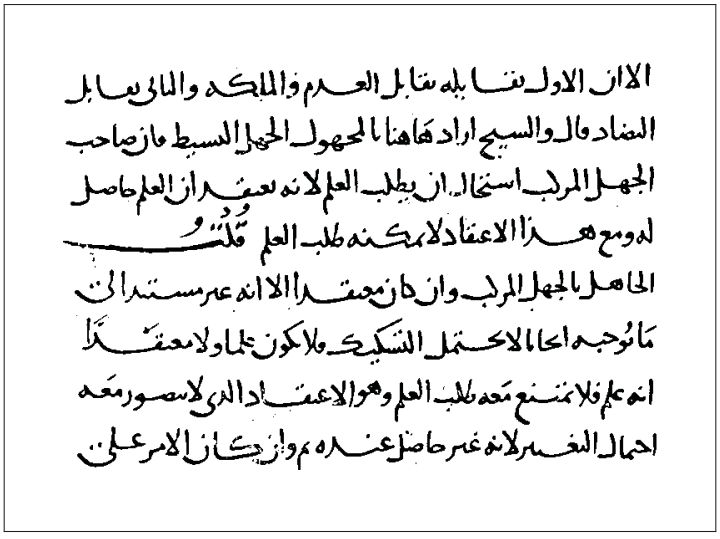


Figure 4.8 Text sample from a page of the Ibn Sina database.

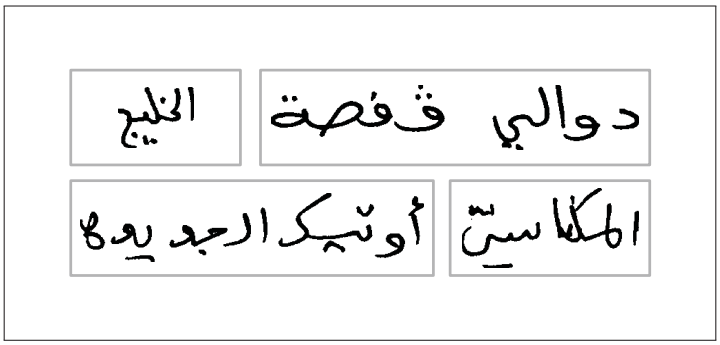


Figure 4.9 Sample words from the IFN/ENIT database.

4.6.2 Experimental protocol

The skeleton image is obtained with the thinning algorithm of MATLAB. Then, a set of 40 pattern filters is used for the feature extraction, containing the 5 patterns (square and lines at 0, 45, 90, and 135 degrees) each at 8 different scales (5, 9, 15, 21, 25, 31, 41, 51). The largest scale is bigger than the average size of the database subwords. Because the total number of pattern filters is small, no feature selection algorithm has been used. Therefore the only free parameters of the system are m, the maximum number of CCs in the AWD, and k, the number of pixel prototypes. The choice of m is guided by level of segmentation of the database. For the Ibn Sina database, labeled at the subword level with archigraphemes encoding, the AWD is built from only one CC ($m = 1$). This setting allows to focus on the subword body and implicitly ignore the diacritics. For the IFN/ENIT AWD, m is set to 20, in order to take into account all the subwords and most of the diacritics, even for large words. For the choice of k, different values have been tested on a subset of the database (results not shown). For Ibn Sina, the values {10, 20, 30, 40, 50, 60, 70, 80} were considered, and we obtained best results for 60 but with only a slight improvement of performance over 50. We therefore favored the simplest model among these two and chose $k = 50$. For IFN/ENIT, the values {1, 5, 10, 15} were considered. These values are smaller than for Ibn Sina, in order to limit the total size of the AWD ($k \times m$). The best results were obtained for $k = 5$. For the construction of the SD, the seeds of the k-means clustering are initialized using the k-means++ algorithm (Arthur and Vassilvitskii, 2007).

Our framework is evaluated by cross validation. The whole database is split into 10 folds for the evaluation (outer folds), where the folds are each considered successively as the test database and the remaining folds form the reference database of the system. The results on the outer folds are averaged, in order to provide a measure of performance on the whole database.

The experiments are performed on a computer with a 2.3 GHz AMD Phenom(tm) 9600B Quad-Core processor, 4 Go of RAM and Windows 7 Enterprise as OS. The code is single threaded, and has been implemented as MATLAB scripts, except the feature extraction with pattern filters which has been implemented in C++. The processing times are given for this configuration.

4.6.3 Lexicon reduction performance

The results of the lexicon reduction performance on both databases are shown in Figure 4.10. The degree of reduction ρ is plotted for different accuracies of reduction α. The degree of reduction remains high, even for $\alpha > 70\%$, and then it drops quickly with a large standard deviation as α approaches 100%. Detailed results are shown in Table 4.1 for specific reduction accuracies. The system performs better on the Ibn Sina database than on the IFN/ENIT one. In particular, for α up to 70%, $max_{rank} = 1$ on the Ibn Sina database, which means that the SD would achieve a recognition rate of 70% by considering the label of its nearest neighbor. Some results of reference database indexing are shown in Figure 4.11.

Table 4.1 Lexicon-reduction performance on the Ibn Sina and IFN/ENIT databases

α (%)	Ibn Sina		IFN/ENIT	
	ρ (%)	η (%)	ρ (%)	η (%)
90.0	99.8 $\pm$ 0.0	89.8	92.1 $\pm$ 1.0	82.9
95.0	97.4 $\pm$ 1.2	92.6	82.1 $\pm$ 1.8	78.0

Representative pixels of visual words are shown in Figure 4.12 (figure best viewed by zooming on a computer screen). A different color is assigned to each visual word. Note that the pixels are clustered according to their topology and geometry, with a different color for branch points and end points, as well as for different orientations.

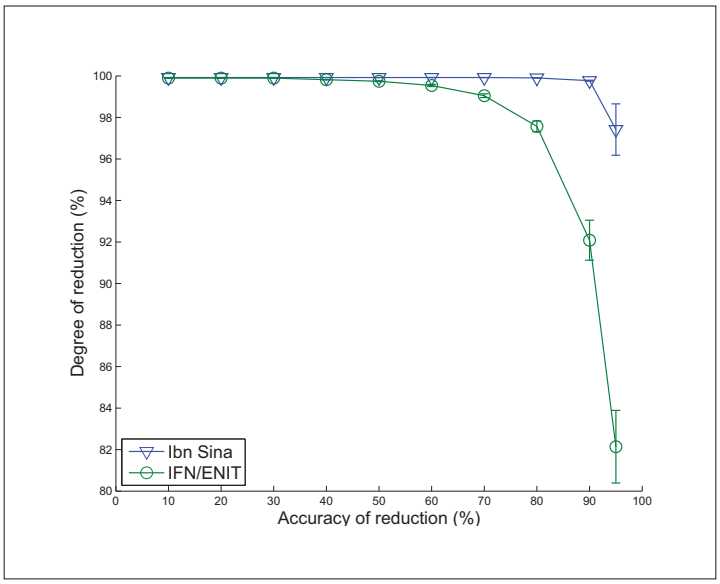


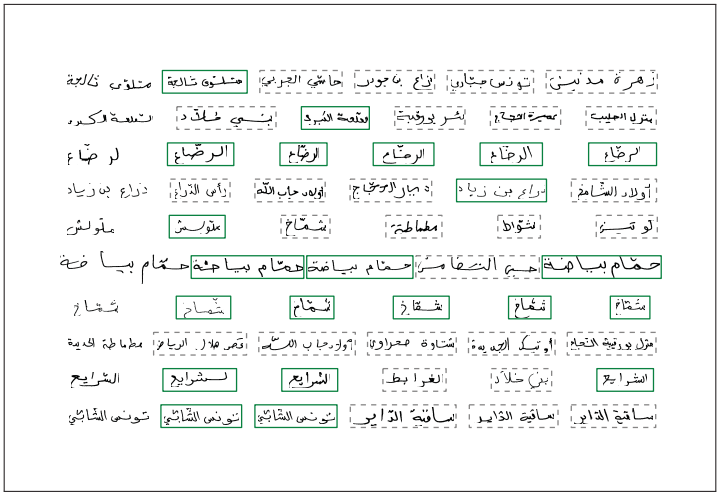
Figure 4.10 Lexicon reduction performance.

The proposed approach produces compact descriptors. The AWD is a 50D vector for the Ibn Sina database, and a 100D vector for the IFN/ENIT database. Also, the computational overhead is relatively small. The average processing time for each word, from the raw image to the formation of the AWD, is 7.0 ms on the Ibn Sina database and 14.0 ms on IFN/ENIT. The average time of lexicon reduction for each query word against the full database is 5.0 ms on Ibn Sina and 6.7 ms on IFN/ENIT.

4.6.4 Analysis of the ADW formation steps

The AWD formation relies on two main steps, sorting the SDs and normalize its entries. In this section, we analysis the relevance of these two steps for lexicon reduction. First, we compare the proposed sorting approach, based on the number of pixels of each CC's skeleton, against 3 simple approaches based on the position of each CC. These approaches sorts the CCs from right to left based on 3 different criteria, respectively the right end, the left end, and the horizontal centroid positions of each CC. All the formed descriptors are compared with and without normalization. The IFN/ENIT database is used for this experiment. The database is separated into 10 folds, 9 folds forms the reference database and the last fold the test database. The results are shown in Table 4.2 for target accuracies of reduction of 90% and 95%. We first

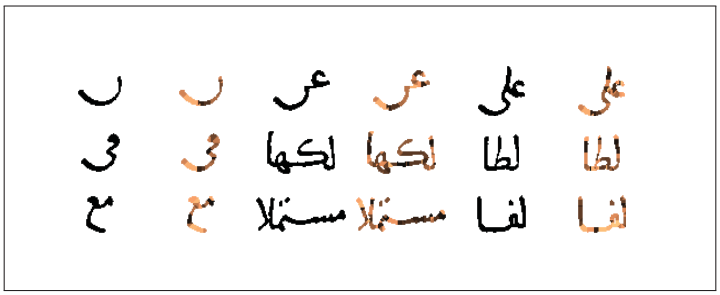
(a) Ibn Sina database



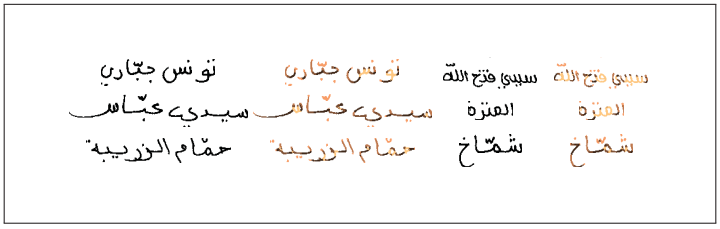
(b) IFN/ENIT database

Figure 4.11 Database indexing based on the AWD. For each row, the first element is the query word image, while the remaining images are the first elements of the sorted reference database. The elements sharing the same label as the query are surrounded by a solid line box.

notice that for all the sorting approaches, the normalization increases the degree of reduction. Among the position based sorting approaches, the best results are obtained for the right end criterium. This is certainly linked to the fact that Arabic is written from right to left. For



(a) Ibn Sina database



(b) IFN/ENIT database

Figure 4.12 Visual words on Ibn Sina and IFN/ENIT databases. The original word image(black) and its partition into visual words, where each color corresponds to one visual word. For clarity, the partition is shown on the original image instead of the skeleton, using the visual word color of the nearest skeleton pixel.

$\alpha = 90\%$ and with descriptor normalization, the right end approach is slightly better (1.4%) than the proposed approach based on the number of pixels. Nevertheless, for $\alpha = 95\%$, it is clearly outperformed (6.5%) by the proposed approach. It shows that sorting based on the number of pixel is more robust than position based criteria for high accuracy of reduction.

Table 4.2 Comparison of different AWD steps for lexicon reduction

Sorting approach	Norm.	ρ (%) ($\alpha = 0.90$)	ρ (%) ($\alpha = 0.95$)
Right end		91.8	73.7
	✓	93.6	77.3
Left end		83.6	58.2
	✓	88.6	62.7
Centroid		85.2	63.4
	✓	90.8	67.5
Num. pixel		87.3	72.3
	✓	92.3	83.8

4.6.5 Combination with a holistic word recognition system

The chosen holistic WRS performs recognition at the subword level. Each subword is described by the square–root velocity (SRV) representation (Srivastava *et al.*, 2011). The subword contour is projected on a Riemmanian manifold, where it is represented as a sequence of velocity points normalized by their square–root velocity. This representation is invariant to scaling and rotation, but the rotation invariance is removed for this application. The SRV is also tolerant to elastic deformations, which often occur during the handwriting process. The dynamic programming algorithm of Chherawala and Cheriet (2012b) is used for optimal SRV sequence alignment. The subwords are classified using a nearest neighbor classifier (1-NN) with the SRV metric. This system has been implemented in C++, and is evaluated on the Ibn Sina database, with the first 50 pages used as the reference database and the remaining 10 pages as the test set. The pixel prototypes are computed on the reference database, and then the SDs are formed for the whole database. During recognition, lexicon reduction is implicitly performed by ignoring all the reference database entries with a rank larger than max_{rank} in the indexed database.

The recognition rate, along with the actual degree of accuracy and degree of reduction on the test set, as well as the average processing time per subword, are shown in Table 4.3 for different values of max_{rank}, expressed here as a percentage of the size of the reference database. The value of max_{rank} goes from 0.1% of the reference database up to 100%, which corresponds to the case where the WRS is run without lexicon reduction. We can see that, as max_{rank} decreases, the accuracy of reduction as well as the classifier recognition rate both decrease, while the degree of reduction increases. A high accuracy of reduction is achieved with max_{rank} as small as 1% of the reference database, for a system 75 time faster and a drop in the recognition rate of just 1.5% compared to the classifier with the full lexicon.

The speed improvement is commensurate with max_{rank}, as only the entries ranked below it are considered during the nearest neighbor search. The computation of a single SRV distance is 0.26 milliseconds, and the matching against large databases takes several seconds. Therefore, database indexing is needed for fast Arabic handwriting recognition using shape analysis

methods, such as the SRV or the shape context (Belongie *et al.*, 2002). In the general case of holistic WRS, where there are as many word models as there are entries in the lexicon, the speed improvement is commensurate with the degree of reduction.

Table 4.3 Lexicon reduction influence on a holistic word recognition system on the Ibn Sina test set

max_{rank} (%)	α (%)	ρ (%)	Classifier recognition rate (%)	Avg. proc. time (ms)
100	100	-	86.2	6376
15	93.5	85.1	86.2	893
10	93.4	87.9	86.1	608
5	93.0	92.4	85.9	320
1	91.6	98.0	85.6	85
0.1	87.9	99.7	83.3	35

4.6.6 Combination with an analytic word recognition system

The analytic word recognition system is based on the well known HMM. We implemented the system proposed by Azeem and Ahmed (2012) which we first describe. A set of 16 concavity features are extracted from the word image using the sliding window approach. The frame width is of 6 pixels, and there is an overlap of 3 pixels between consecutive frames. The delta and acceleration features are also computed, leading to a total of 48 features for each frame. An HMM model with 6 emitting states and a mixture of 64 Gaussians per state is trained for each symbol of the alphabet. The word level HMM is built by concatenating the HMMs of the symbols forming the word. During the recognition, all the word level HMMs of the lexicon are tested and the word hypothesis having the highest likelihood is chosen as the recognized word. We used the HTK (Young *et al.*, 2006) implementation of HMM to build this system. It has been trained on the sets A, B, C, and D of the IFN/ENIT database, and tested on the set E.

Here as well, the pixel prototypes are computed from the training sets and then the AWDs are formed for the whole IFN/ENIT database. The lexicon is dynamically reduced using our approach for different values of max_{rank}. The results are shown in Table 4.4. We see that

the accuracy of reduction drops progressively with respect to max_{rank}. It is therefore harder to achieve high performance for both accuracy of reduction and degree of reduction. A good compromise between the classifier recognition rate and the average processing time per image is achieved by considering $max_{rank} = 15\%$, for a drop of the recognition rate of 3.2% for a speed improvement of approximately 20%, compared to the WRS with a full lexicon.

Table 4.4 Lexicon reduction influence on an analytic word recognition system on the IFN/ENIT set E

max_{rank} (%)	α (%)	ρ (%)	Classifier recognition rate (%)	Avg. proc. time (s)
100	100	-	88.1	4.7
15	95.5	45.5	84.9	3.9
10	92.5	55.8	82.6	3.8
5	85.9	70.3	77.7	3.5
1	64.7	89.7	60.5	2.8
0.1	32.6	98.2	31.6	1.4

4.6.7 Comparison with other methods

The proposed method has been compared with other available approaches (Table 4.5). The ideal diacritic matching method extracts a sequence of diacritics directly from the subword label and reduces the lexicon by removing from it unmatched sequences. It therefore represents an upper bound for all the methods based only on diacritic matching on the Ibn Sina database, as there is no error in the sequence extraction process. The sparse descriptor and the Arabic word descriptor comes from the earlier version of this work (Chherawala *et al.*, 2012), where only a single square pattern filter is used. The other methods were briefly detailed in Section 5.1. Our method shows the best reduction efficacy on both databases. Furthermore, it is the only method that is competitive, at both the subword and the word level. Note that, because a training set and a testing set were not clearly defined in the previous experimental protocols, we used cross validation to estimate our system parameters. Our protocol is therefore slightly different from the one used in previous methods, but we believe the results are comparable.

Table 4.5 Comparison with other lexicon-reduction methods

Database	Method	α (%)	ρ (%)	η (%)
Ibn Sina	Ideal diacritics matching	100	75.0	75.0
	W-TSV (Chherawala and Cheriet, 2012a)	90.0	92.9	83.6
	Sparse descriptor (Chherawala *et al.*, 2012)	90.0	95.2	85.7
	Proposed method	**95.0**	**97.4**	**92.6**
IFN/ENIT	Subword and diac. (Mozaffari *et al.*, 2008a)	74	92.5	68.5
	Improved diacritics (Wshah *et al.*, 2010)	94.6	85.6	81.0
	W-TSV (Chherawala and Cheriet, 2012a)	90.0	83.6	75.2
	Arabic word desc. (Chherawala *et al.*, 2012)	90.0	90.1	81.1
	Proposed method	**90.0**	**92.1**	**82.9**

4.7 Conclusion

In this work, we proposed an Arabic word descriptor for word indexing and lexicon reduction. It encodes the shape of each connected component of the image through a structural descriptor (SD) based on the bag–of–words model. The sorting and normalization of the SDs emphasize the symbolic features of Arabic words, such as the subwords and the diacritics. Experiments on Arabic word databases demonstrate the suitability of the AWD for lexicon reduction, thanks to its computation efficiency and high accuracy of reduction. In future work, the AWD will be combined with complementary shape representations, in order to improve its performance for very high accuracy of reduction, and spatial constraints will be added as features. In order to reduce the impact of the errors introduced by the lexicon reduction system, a rejection mechanism will be added at the output of the word recognition system. The broader scope of this work is to reduce the processing time of individual word recognition systems, so that multiple word recognition systems can be run efficiently, in order to improve recognition accuracy by combining their outputs.

4.8 Acknowledgments

The authors thank the NSERC and SSHRC of Canada for their financial support.

CHAPTER 5

ARTICLE III - FEATURE EVALUATION FOR OFFLINE HANDWRITING RECOGNITION USING SUPERVISED SYSTEM WEIGHTING

Youssouf Chherawala, Partha Pratim Roy and Mohamed Cheriet

Synchromedia Laboratory, École de Technologie Supérieure
1100 Notre-Dame Ouest, Montréal, QC, Canada

Submitted to the IEEE Transactions on Pattern Analysis and Machine Intelligence

Abstract

A large body of features for handwriting recognition exists in the literature, but no method has yet been proposed to identify the most promising of these, other than a superficial comparison based on the recognition rate. In this paper, we propose an advanced framework for feature evaluation in handwriting recognition. A combination scheme has been designed for this purpose, in which each feature is represented by an agent, which is an instance of a reference recognition system trained with that feature. The decisions of all the agents are combined using a weighted vote, in which the weights are optimized during a training phase. Finally, the weights are converted into a numerical score assigned to each feature, which is easily interpreted with this model. The main contribution of this work is to quantify the individual importance of the evaluated features, as our scheme allows the efficiency and complementary nature of the features to be assessed. We used the recurrent neural network (RNN) as the reference system. The second contribution is to provide the first feature benchmark using this RNN recognition system. We evaluated several features on Arabic and Latin word databases, which provided us with interesting insights for future feature design.

Keywords

Feature evaluation, Analytical word recognition, System combination, Recurrent neural network, IFN/ENIT, RIMES

5.1 Introduction

The recognition of handwritten text is a challenging task, owing to the huge variation in writing styles of individual writers. As text is formed as a sequence of characters, this sequential behavior is reproduced at image level for text decoding, where a text line is decomposed into a sequence of vertical frames. Features are extracted from each frame and fed into a decoding system to retrieve the text sequence of characters. In spite of extensive research with hidden Markov models (HMM) and hybrid neural network-HMM models for sequential data transcription (Rabiner and Juang, 1986; Morgan and Bourlard, 1995; Vinciarelli *et al.*, 2004) documented in the literature, feature extraction remains a challenge.

The goal of features is to remove unnecessary variability, in the form of individual writing style, from a word image, and keep only the information relevant for word recognition. Their use goes from word-spotting (Rath and Manmatha, 2003a; van der Zant *et al.*, 2008; Lladós *et al.*, 2012; Rodríguez-Serrano and Perronnin, 2012), where information about word labels is seldom used, to word recognition (Plamondon and Srihari, 2000; Vinciarelli, 2002; Vinciarelli *et al.*, 2004; Lorigo and Govindaraju, 2006), using word label information during system training. Nevertheless, feature design (Rath and Manmatha, 2003b; Adamek *et al.*, 2007; Chherawala and Cheriet, 2012a; Slimane *et al.*, 2012) for handwritten word shape is a difficult task, because the requirements for good features of word images cannot be explicitly defined (i.e. by a set of rules) when the word image is degraded or the handwriting is variable.

For this reason, there is a large body of features in the literature for handwriting recognition in Latin and Arabic scripts (Chherawala *et al.*, 2012; Eraqi and Abdelazeem, 2012; Li *et al.*, 2012), and the search for the 'ultimate' feature is far from over. Existing features are based on models devised in various fields, such as pattern recognition, computer vision, and machine learning. Because of their different backgrounds, it is very difficult to compare these models on a theoretical basis. Moreover, they are often used on different databases, with different protocols and recognition systems. This makes it difficult to decide which feature should be used for a new application. The literature does not provide clear guidelines on relevant features, and it is mostly reduced to a listing of all the features ever proposed. As a result, more

and more features are proposed, with no principled design for the task of handwriting recognition. Although new features certainly make a significant contribution in their respective fields (computer vision, machine learning, etc.), that contribution is not clear in the context of handwriting recognition, where features from a number of fields are used. It is therefore important to compare existing features first, and then identify the most promising of these. However, no tool exists for this task, except evaluation based on the recognition rate, but this approach provides only a superficial insight into the features and totally ignores their complementarity. What is needed are efficient tools for feature evaluation, so that the next generation of features can improve the efficiency of the handwriting recognition process.

In this paper, we propose a framework for feature evaluation in analytic handwriting recognition. Features are represented by *agents*, which are instances of a reference word recognition system based on the recurrent neural network (RNN). All the agents are then evaluated using a combination scheme at the decision level, based on a variant of the weighted vote. The weights assigned to each agent are optimized, in order to maximize the combination recognition rate. The weights, symbolizing the importance of each agent, are then converted into easily interpreted feature scores. We evaluated a total of five features, including *handcrafted* features, which are designed based on expert knowledge, and *automatically learned* features based on machine learning models. Specifically, we considered the following categories of features: distribution, concavity, visual descriptor-based, and automatically learned.

The main contribution of this work is to provide a feature evaluation framework capable of quantifying the relative importance of each feature using a numerical score. That score provides insight into existing features strength and complementarity, information which is useful for the design of the next generation of features. The combination scheme used in this framework also improves the confidence level of the true word label during recognition. The second contribution is to provide the first feature benchmark using the state–of–the–art RNN model. This RNN outperformed the classic HMM on several handwriting tasks (Märgner and El Abed, 2009; Grosicki and El Abed, 2009), however no feature benchmark is available yet for this recently proposed approach.

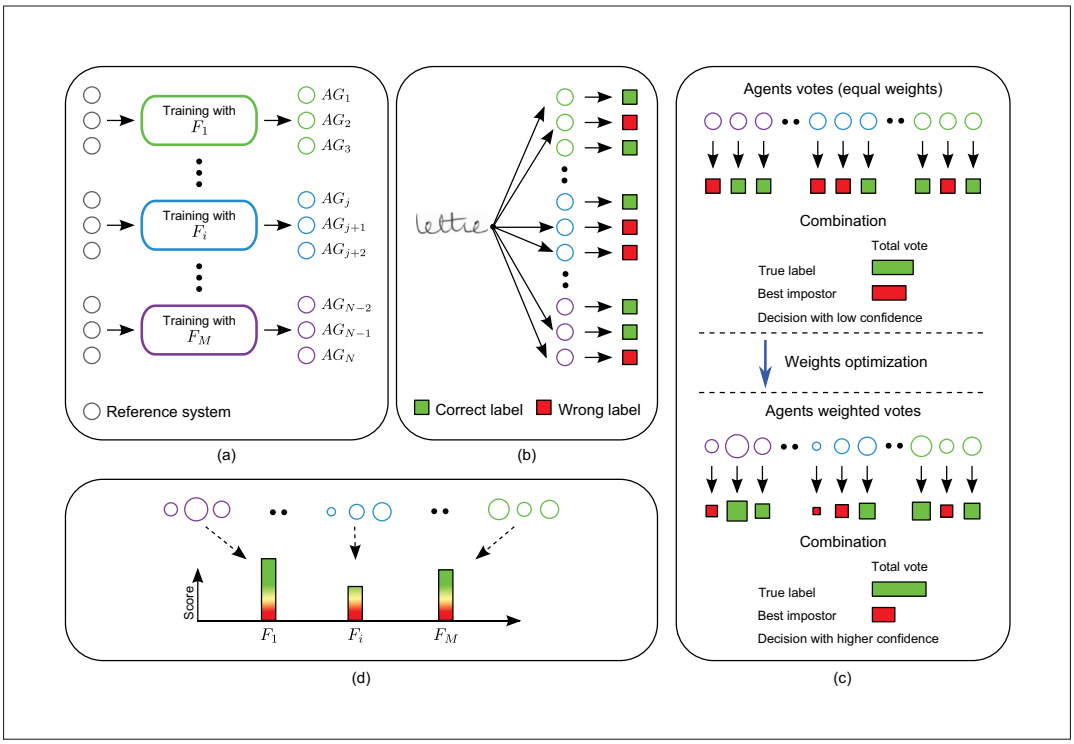


Figure 5.1 Evaluation framework. (a) Agent training: each agent AG_j is obtained by training a reference system with a specific feature F_i. (b) Word recognition of the agents. (c) Agent combination based on a weighted vote: the weights are optimized to increase the confidence of the true label (the size of the agents is proportional to their weights). (d) Feature evaluation: the agents' weights are converted into scores for each feature. (For an accurate visualization of the colors of this figure, please refer to the Web version of this article.)

This paper is an extension of the work published by Chherawala *et al.* (2013). In particular, that extension includes feature evaluation based on combining agents, and the use of a complete reference system, with the integration of the token passing algorithm. The experimental section has also been significantly improved by considering two databases to test our framework.

The rest of the paper is organized as follows. Related work is reviewed in Section 5.2. We provide an overview of our framework in Section 5.3. We describe the RNN recognition system in Section 5.4 and the evaluated features in Section 5.5. The agent combination scheme is presented in Section 5.6. Finally, the experimental setup is given in Section 5.7 followed by our results and a discussion in Section 5.8.

5.2 Related work

One way to design features is to benefit from expert knowledge. In this case, features are handcrafted by experts in the field based on their knowledge and experience. Handcrafted features exhibit the word shape structure, and combine the shape geometry and topology. However, these shape properties are difficult to capture explicitly, and are therefore expressed as a count of specific patterns or through the spatial distribution of foreground pixels (Rath and Manmatha, 2003b). Distribution features characterize the density of these pixels in an image frame (Al-Hajj Mohamad *et al.*, 2009). These features typically relate to the number of foreground pixels, the number of foreground–to–background transition and to the lower and upper word shape profile. They capture the presence of ascenders and descenders in the word image, which are important cues for correct word recognition. For Arabic word shapes, the geometry is often extracted through concavity features, which provide stroke direction and concavity information (Al-Hajj Mohamad *et al.*, 2009; Azeem and Ahmed, 2012). These are computed with a hit-or-miss transform, based on morphological patterns. Also, recent advances in computer vision have produced efficient visual descriptors, such as SIFT (Lowe, 2004), SURF (Bay *et al.*, 2008), and HOG (Dalal and Triggs, 2005), which are based on local histograms of gradient orientation. These descriptors have inspired new features for word shape. For example, Rothacker et al. built bag-of-word features from SIFT descriptors in combination with HMM (Rothacker *et al.*, 2012). These visual descriptors have also been adapted to the specificity of word images for word-spotting applications (Rodríguez-Serrano and Perronnin, 2009; Terasawa and Tanaka, 2009).

A popular alternative to handcrafted features is the use of dimensionality reduction methods (Roweis and Saul, 2000) for automatic feature extraction. In such settings, new features can be extracted either in a supervised fashion (using the target label information) or unsupervised one. Principal component analysis (PCA) performs linear dimensionality reduction, and is among the most popular unsupervised feature extraction methods. Nonlinear methods, such as kernel PCA (Scholkopf *et al.*, 1999) and autoencoder neural networks (Vincent *et al.*, 2008), can explain nonlinear dependencies among the input variables.

Feature extraction can also be performed in a supervised fashion, where the target recognition task has a direct influence on the extraction process. This is typically the case in Multi-Layer Perceptron (MLP) neural network. The output of each hidden layer consists of features extracted by a nonlinear combination of the features of the previous layer, and the weights of the combination are learned during the training phase. For handwriting recognition, however, MLP lacks the ability to deal with unsegmented data, unlike HMM for example. To combine the strengths of both models, combining the MLP neural network with HMM in the so–called hybrid neural network/HMM system has been proposed, where the HMM observation probabilities are based on the output of the MLP, instead of the classical Gaussian mixture model. This idea has been extended to tandem systems, where the MLP is used as a feature extraction module (Hermansky *et al.*, 2000; Dreuw *et al.*, 2011). The training of the tandem system involves several steps. First, the word slices are given the label of their characters, either manually or by using a previously trained HMM in forced alignment mode. Then, the MLP is trained to recognize the label of the image slices without feature extraction. Finally, the output of the MLP followed by dimensionality reduction is considered as the extracted features for a new HMM model. This use of a neural network follows the sliding window approach for features. Another approach is based on vision and image recognition, where neural networks are given a specific architecture to emulate the behavior of the visual cortex. In convolutional neural networks (LeCun *et al.*, 1998), the weights act as local image filters and produce multiple feature maps at each layer. Each feature map is a 2D image, produced in two steps. First, the output of the previous layer is convolved with a set of weights, and then it is usually subsampled with max-pooling. The activation of the feature maps of the first layer typically corresponds to the image edges. When multiple layers of hidden layers are stacked – forming a deep neural network – a hierarchy of more and more abstract features is created. This architecture has been combined with RNN (Graves and Schmidhuber, 2009) and provides an alternative model for automatic feature extraction.

Feature evaluation has been proposed for handwritten numeral recognition (Oh *et al.*, 1999) based on their class separation and recognition capabilities. However, this approach is not applicable to analytical systems. For word recognition, feature evaluation is based on the com-

bination of reference classifiers at the decision level (De Oliveira *et al.*, 2002). Each classifier is trained with a single feature, and the performance of features is based on the recognition rate of their system combination. However, this approach doesn't measure the individual contribution of each feature. Several other combination methods exist in the literature (van Erp *et al.*, 2002). The simplest one is the plurality vote, where each classifier votes for a word hypothesis, and the one with the largest number of votes is selected. One of its variants is the sum rule, where the vote of each classifier is weighted by its confidence. One drawback of the sum rule is that the confidence of the classifier must be well scaled for good performance. Other famous approaches are based on ranking, in which each classifier provides an N-best list. The Borda count method selects the word candidate with the highest average rank. However, none of these methods provides an evaluation of the classifier. Re-ranking methods have been proposed by Al-Hajj Mohamad *et al.* (2009); Bianne-Bernard *et al.* (2011), where an MLP is trained to select the true word hypothesis, given the confidence of various classifiers over their N-best list as input. Unfortunately, MLP neural networks are not explicit models and can't be used to evaluate the relative strength of the base classifiers. In Menasri *et al.* (2012), the voting weights of each classifier are learned in a supervised scheme, which can be derived to evaluate the classifiers explicitly.

5.3 Feature evaluation framework overview

As mentioned in the introduction, we propose a framework for feature evaluation in analytic handwriting recognition. Features are indirectly evaluated by means of a reference word recognition system. At least one instance of the reference system is trained for each feature, and we refer to an instance as an *agent* of that feature. Given a query word image, the agent proposes a word hypothesis. Then, the agent votes for its recognized word, and all the votes are gathered using a weighted vote variant. Each agent is assigned a weight for its vote, which is optimized during a training phase to maximize the confidence of the true word label over the *best impostor*, that is, the word with the highest confidence, but different from the true label. Because the decisions of all the agents are known during optimization, the weights are set based on collective performance, and not individual performance. Therefore, the weights represent

the contribution of each agent to the vote based on their collective strength. The weights are then converted into easily interpreted scores and assigned to the features of the agents. The framework is illustrated in Figure 5.1.

5.4 RNN-based reference recognition system

We have chosen the recurrent neural network (RNN) as the reference recognition system for our framework for two reasons. First, RNNs have been shown to perform better than HMMs for several sequence-decoding problems, in particular handwriting recognition (Graves *et al.*, 2009). This is because RNNs are discriminative models, while standard HMMs are generative. Second, RNNs are able to seamlessly learn features from the input image in a supervised fashion, which HMMs can't. This makes the RNN a good representative for a system based on learned features. The RNN-based recognition system is made up of two distinct neural networks. The first is the long short-term memory (LSTM) network, which can access a long range temporal context. The second is the connectionist temporal classification (CTC) output layer, which is able to transcribe unsegmented data.

The architecture of the system differs, depending on whether the features are handcrafted or learned. Handcrafted features are first extracted from the input image, and then they are fed in frame-wise fashion to the LSTM neural network. Finally, the CTC decoding layer provides the recognized character sequence as output. For learned features, the input image is directly fed to a multidimensional LSTM (MDLSTM) neural network, and then to the CTC decoding layer. In fact, the MDLSTM network replaces both the handcrafted feature extraction module and the LSTM neural network of the handcrafted feature system. The architecture of the system for both types of features is illustrated in Figure 5.2. Below, we describe the core of our recognition system, that is, the LSTM and CTC layers. The MDLSTM layer is described in Subsection 5.5.4.

5.4.1 Long short-term memory (LSTM) layer

The LTSM layer is made up of nodes with a specific architecture called a *memory block*, which is capable of preserving contextual information over a long period of time. Each memory block

contains a memory cell, and its interaction with the rest of the network is controlled by three multiplicative gates: an input gate, an output gate, and a forget gate. For example, if the input gate is closed, the block input has no influence on the memory cell. Similarly, the output gate has to be open, so that the rest of the network can access the cell activation. The forget gate scales the recurrent connection of the cell. The gate behavior is controlled by the rest of the network. For the specific task of handwriting recognition, the 'past' and 'future' contexts are necessary for better performance. Therefore, the bidirectional LSTM (BLSTM) layer is used, where one LSTM layer processes the feature sequence in the forward direction, while another layer processes it in the backward direction. The output of the two layers is combined at the next layer as a feature map. As with the convolutional neural network architecture, it is possible to have multiple forward and backward layers in each LSTM layer, as well as multiple feature maps at the output layer, and to stack multiple LSTM layers using max-pooling subsampling.

5.4.2 Connectionist temporal classification (CTC) layer

Usually, most RNNs require pre-segmented training data or postprocessing to transform their output into transcriptions. To avoid this process, the CTC output layer has been designed to label unsegmented sequences. This layer is trained to predict the probability $P(w|\mathbf{O})$ of an output character sequence, that is, a word w, given an input feature sequence $\mathbf{O}$, making the training discriminative. The output activation function provides the probability of observing each character for each time of the sequence. The CTC is trained to minimize the negative log probability of the ground truth label over the entire training set. Once the network is trained, the labeling of an unknown input sequence $\mathbf{O}$ is performed by choosing the word $\hat{w}$ with the highest conditional probability from a given lexicon, that is:

$$\hat{w} = \arg\max_{w} p(w|\mathbf{O}) \tag{5.1}$$

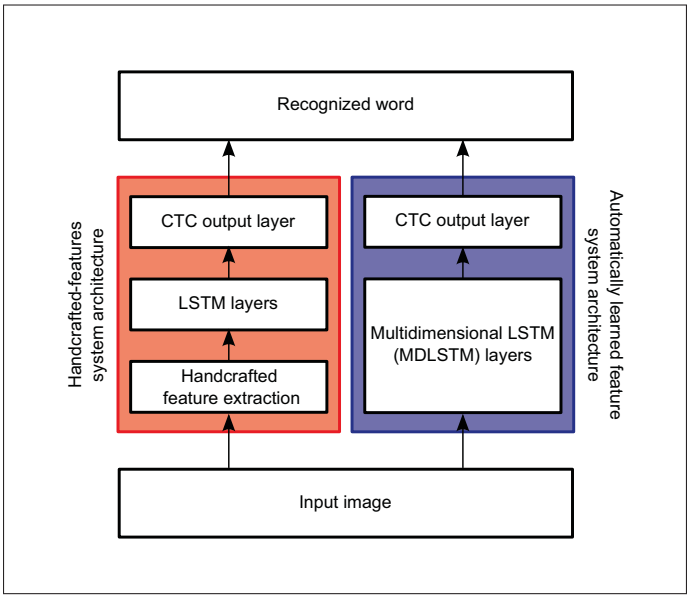


Figure 5.2 Recognition system architectures. On the left is the system for handcrafted features. On the right is the system for automatically learned features (see text for more details).

5.5 Word image features

In this section, we present the image features evaluated for word recognition systems. We provide justification for their selection, and we detail their extraction procedure. They have been organized into four categories: distribution features, concavity features, visual descriptor-based features and automatically learned features. These categories have been chosen either because of their state-of-the-art performance (distribution and concavity features), or because they represent recent trends in feature design, inspired by computer vision and machine learning. The first three categories correspond to handcrafted features, and, when one of these features overlaps several categories, we assign it to the most relevant one. The handcrafted features are obtained by sliding a frame window horizontally over the word image and computing the features in each frame.

5.5.1 Distribution features

Two distribution features are described here. They are both extracted in column–wise fashion. The first feature was proposed by Rath and Manmatha (2003b) (the R-M feature) for handwritten word–spotting in historical manuscript. Each word image is described as a sequence of 4D feature vectors: the upper and lower profiles, the projection profile, and the background–to–foreground transition profile. The minimum and maximum positions of the foreground pixels are considered as the lower and upper profiles. The projection profile is the number of foreground pixels in the corresponding column. The number of transitions between the foreground and background pixels is used as the transition profile. In word–spotting, the features extracted from two word images are matched using Dynamic Time Warping for similarity measurement. This feature is popular because it is simple and robust to image degradation.

The second feature was proposed by Marti and Bunke (2001) (the M-B feature), and has been used by many researchers for handwritten text recognition with HMM. Nine features are computed from the set of foreground pixels in each image column. Three global features capture the fraction of foreground pixels, the center of gravity, and the second order moment. The remaining six local features are: of the position of the upper and lower profiles, the number of foreground–to–background transitions, the fraction of foreground pixels between the upper and lower profiles, and the gradient of the upper and lower profile with respect to the previous column, which provides dynamic information.

5.5.2 Concavity feature

Azeem and Ahmed (2012) proposed a set of concavity features (the CCV feature) for Arabic word images, which has proved to be effective for Arabic text recognition using HMM, where a recognition accuracy of 88.5% has been reported without image preprocessing. First, the stroke thickness is normalized to a 3–pixel width by a thinning operation followed by dilation. Then, the response of the normalized image to 8 directional morphological filters is computed, leading to 8 binary directional images. Vertical frames 6 pixels in width are then used to extract the feature, with an overlap of 3 pixels between two consecutive frames. In each frame and for

each directional image, the number of '1' pixels, as well as the normalized gravitational center of these pixels, is extracted as a feature. The final feature vector therefore contains 16 features per frame. The original feature also includes dynamic features (delta and acceleration), but these additional features are not included in our framework, as we expect the LSTM network to capture the temporal dependencies.

5.5.3 Visual descriptor-based feature

Rodríguez-Serrano and Perronnin (2009) developed a SIFT–like feature called the LGH feature in their word–spotting application. The image is divided into overlapping frames. The region in each frame is divided into 4×4 regular cells. Next, a histogram of gradients (8 bins) is computed in each cell, and the final vector represents the concatenation of the 16 histograms, which results in a 128D feature vector for each frame. Each feature vector is scaled to unit norm for local contrast normalization. Note that the construction of the LGH can be summarized in two steps: image filtering followed by local sum-pooling for subsampling. These steps are typical of vision-based features. The authors have shown that the LGH feature provides better performance accuracy in handwritten text word–spotting (Rodríguez-Serrano and Perronnin, 2009). The same frame width and overlap as for the concavity features are used here.

5.5.4 Automatically learned feature

The automatically learned feature is based on the MDLSTM neural network (Graves and Schmidhuber, 2009). This network is a multidimensional extension of the LSTM network. In this setting, the multidimensional data are scanned as multiple 1D sequences, by setting the scanning directions and the priority of the dimensions during scanning. For example, in a 2D image, we can choose to scan forward along the x dimension and backward along y dimension, with a higher priority for the x than for y, so that, during the scan, the x index will be updated before the y index, according to the scanning direction. Each hidden layer memory block has a recurrent connection with the memory blocks one step back, according to the scanning direction for every dimension. One such layer provides the network with full context along the scanning direction. As there are 4 possible directions in 2D images (i.e. forward x and y, back-

ward x and forward y and so on), 4 layers are necessary to have full context in all directions (Figure 5.3). As with the LSTM layer, it is possible to have multiple layer scanning in the same direction, and to combine them to form multiple feature maps at the output layer. Moreover, a hierarchy of the MDLSTM layer can be built, with 2D subsampling between layers. Because of this architecture, specifically at the first layers (image filtering with MDLSTM layers followed by subsampling), the MDLSTM can also be considered as a vision-based feature.

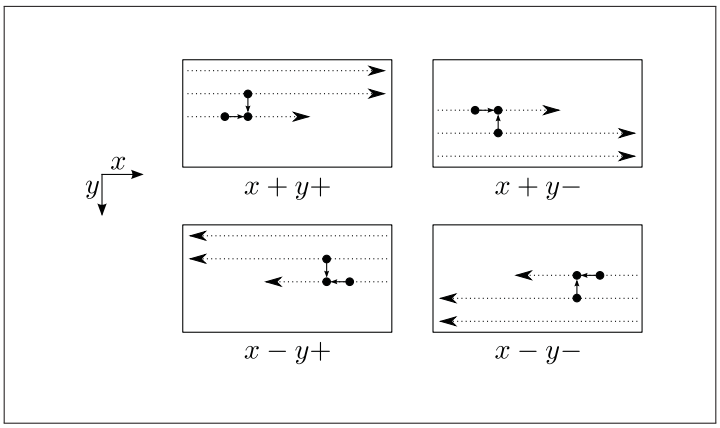


Figure 5.3 2D MDLSTM scanning directions and context propagation in hidden layers. The priority direction is x. $+$ represents the forward direction and $-$ the backward direction.

5.6 Feature evaluation using agent combination

In this section, we present our strategy for feature evaluation. Each feature is represented by an agent, which is an instance of the reference RNN system trained with that feature. The evaluation is based on the combination at the decision level of N agents AG_i representing the evaluated features, using a weighted vote approach. Only the best recognition of each agent is considered for the vote. The weights of the agents are determined during a learning process and are transformed into scores for the agent's feature. A feature is not limited to a single agent, and it can have several agents. Such a scenario is needed in the case of RNN, because different models can be obtained with the same feature, owing to different initialization of the parameters. First, we detail our combination strategy, and then we describe our definition of the feature evaluation score.

5.6.1 Supervised agent weighting

Our agent combination approach is based on the weighted vote introduced by Menasri *et al.* (2012). It is similar to the traditional plurality vote, except that the vote of each agent is weighted:

$$n_w = \sum_{i}^{N} \alpha_i D_i(w) \tag{5.2}$$

where n_w is the sum of the weighted votes received by the word hypothesis w, $D_i(w) = 1$ if AG_i votes for w (i.e. if its best recognition is w), else 0, and α_i is the weight associated with AG_i. This sum is converted into a confidence value bounded in $[0, 1]$ using the logistic function:

$$\sigma(n_w) = \frac{1}{1 + \exp(-n_w - b)} \tag{5.3}$$

where b is a bias parameter and $\sigma(n_w)$ is denoted σ_w in short notation. Finally, the word hypothesis from the lexicon with the highest confidence is selected:

$$\hat{w} = \arg\max_{w} (\sigma_w) \tag{5.4}$$

The weights α_i and the bias b are optimized during the learning procedure. The original approach minimizes the following loss function based on the true word hypothesis w_{gt} and the best word impostor $w_{imp} = \arg\max_{w|w \neq w_{gt}} (\sigma_w)$:

$$\begin{aligned} \sum_{j}^{K} & -\log\left(1 - \sigma\left(w_{gt}^{j}\right)\right) \\ & - \left[\log\left(\sigma\left(w_{imp}^{j}\right)\right) \quad \text{if} \quad \sigma\left(w_{gt}^{j}\right) < \sigma\left(w_{imp}^{j}\right) \right] \end{aligned} \tag{5.5}$$

where j represents the index of the K samples of the database. The second term is involved in the optimization only if the best impostor has a higher confidence than the true word hypothesis[1].

Instead, we propose to directly maximize the margin between w_{gt} and w_{imp}. The motivation for this choice will be explained in the experimental section. We therefore maximize the following objective function:

$$O = \sum_{j}^{K} \sigma\left(w_{gt}^{j}\right) - \sigma\left(w_{imp}^{j}\right) \tag{5.6}$$

This function can be optimized using stochastic gradient ascent. For an easy interpretation in the evaluation step, the weights α_i are constrained to be positive. The procedure used to optimize the combination parameters is described in Algorithm 2. Line 8 represents the positiveness constraint. The parameters found after the convergence are used for the combination scheme. As will be shown in the experimental section, the main advantage of this combination is to improve the confidence in the true word label during recognition.

Algorithm 2 Gradient-ascent optimization for the combination

Input: Best recognition of the N agents for the K database samples
Output: Combination parameters $p = [\alpha_1, \ldots, \alpha_N, b]$
Parameters: Learning rate η and momentum m

1: **repeat**
2: Randomly shuffle the database samples
3: **for** $j = 1 \rightarrow K$ **do**
4: // Update rule
5: $Q_j = \sigma\left(w_{gt}^{j}\right) - \sigma\left(w_{imp}^{j}\right)$
6: $\Delta p_t = \eta \nabla Q_j + m \Delta p_{t-1}$
7: $p \leftarrow p + \Delta p_t$
8: $\alpha_i \leftarrow \max(\alpha_i, 0)$
9: **end for**
10: **until** convergence

[1]Menasri *et al.* (2012) used multiple word hypotheses per agent, and $D_i\,(w)$ represents the recognition confidence for each hypothesis.

5.6.2 Score definition

As explained in the previous section, each AG_i is associated with a weight α_i that we consider to be the contribution of the feature represented by AG_i toward the combination. For an easy interpretation, the weights α_i are converted into scores s_i, by normalizing them to unit sum ($s_i = \alpha_i / \sum \alpha_i$).

We can therefore quantify the contribution of each feature through a given agent in the form of a percentage. Furthermore, in the case where multiple agents represent the same feature, all their scores are summed and assigned to that feature. Therefore, s_i will hereafter refer to the score of an individual agent, while $\sum s_i$ will refer to the score of a feature. Because all the features were considered during optimization, the weights not only reflect their relative strength, but also their complementarity. The complexity of this approach, based on combination at the decision level, is relatively low in practice, as only a small number of parameters have to be optimized. In fact, it only requires that an agent be trained with a single feature. This is far less costly than combination at the feature level, because of the large number of combinations and the long convergence time required to train all the recognition systems.

5.7 Experimental setup

5.7.1 Databases

We used two databases for our experiments. The first is the IFN/ENIT database (Pechwitz *et al.*, 2002) for Arabic script, and the second is the RIMES database (Grosicki *et al.*, 2009) for Latin script (Figure 5.5).

The IFN/ENIT database is composed of 32,492 images of Tunisian city and village names written by several hundred different writers. This database is divided into five sets: A, B, C, D, and E. From each of the first 4 sets, we randomly chose 500 images as the validation set, and the remaining images as the training set. We used the set E for testing.

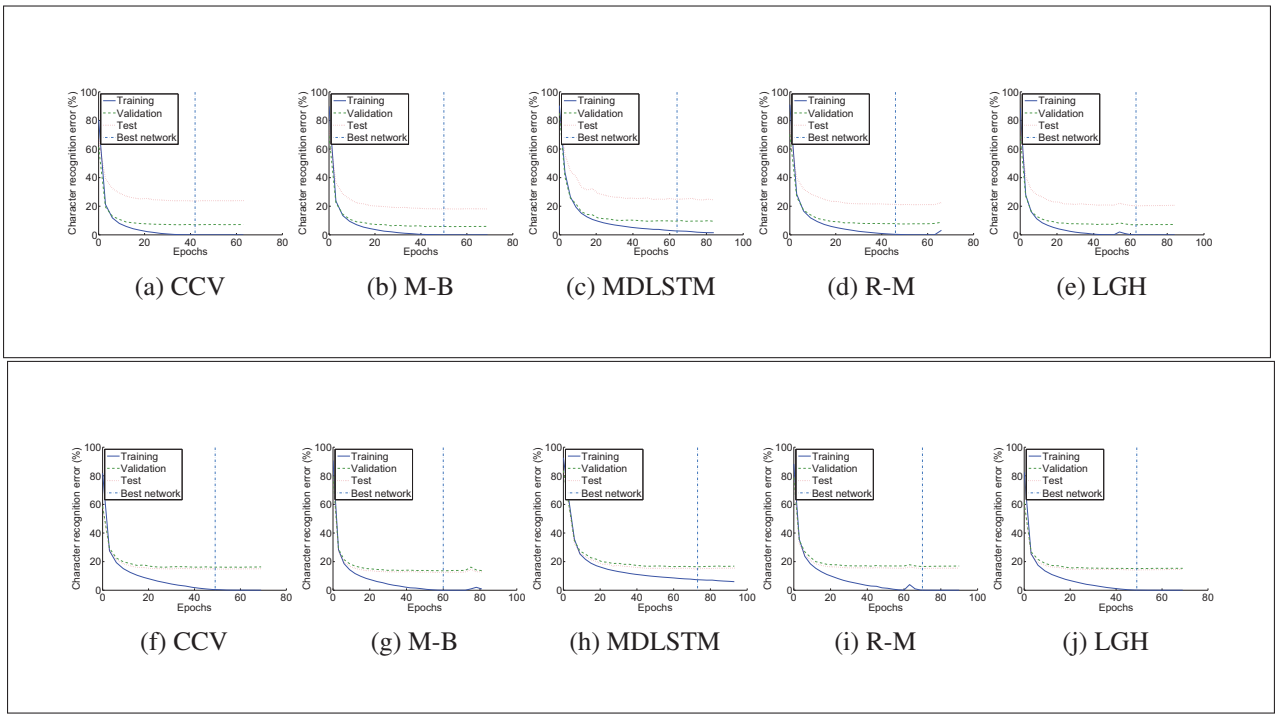


Figure 5.4 Character recognition error rate during neural network training for different features on the IFN/ENIT database (first row) and the RIMES database (second row). The best model on the validation set for each feature is shown.

The RIMES database is composed of more than 12,000 mails written in French, all annotated at the word level. We used the 2009 version of the database, which is divided into training, validation, and test sets, containing 59,203, 7,542 and 7,464 images respectively. The images are in gray level, and so they have been binarized using the Otsu (1979) algorithm for all the features, except for the MDLSTM model. The decision with respect to the MDLSTM model is justified by the results of Menasri *et al.* (2012), which show similar performance using binarized or gray-level images. Moreover, we kept the distinction between characters with and without accents, for example e, é, and è are considered as different characters.

5.7.2 Experimental protocol

For both the handcrafted and learned features, the network architecture is made up of a hierarchy of three LSTM/MDLSTM layers. In Table 5.1, we provide the details of each level of the hierarchy. The layers of the last level are directly fed to the CTC network. For the

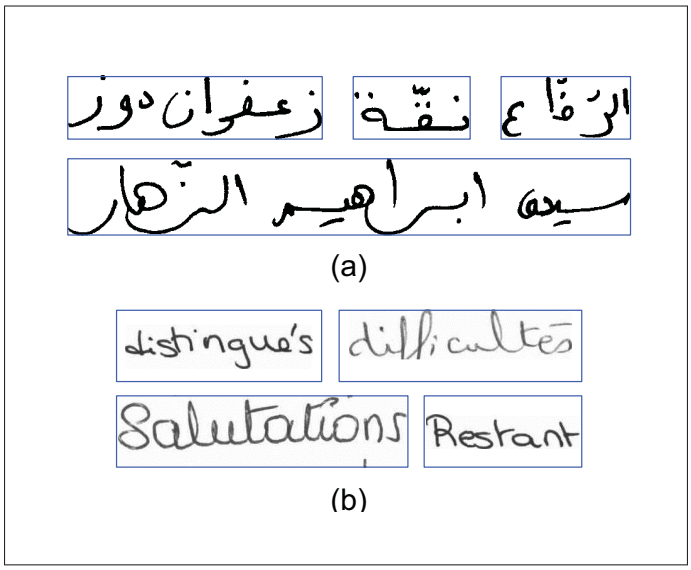


Figure 5.5 Sample images from the experiment databases. (a) IFN/ENIT. (b) RIMES.

MDLSTM features, we use the same network architecture as Graves and Schmidhuber (2009). For further details, please refer to Graves and Schmidhuber (2009). For all the networks, the learning rate has been set to 10^{-4} and a momentum of 0.9 has been used. The training stops after 20 iterations without improvement for the character level error rate on the validation set. The experiment is reproduced 5 times for each feature, because of the random initialization of the neural network during the training phase. This leads to a total of 25 agents. For our experiments, we used the RNNLIB implementation of the recurrent neural network (Graves).

The combination is optimized on the validation set of each database using all 25 agents. Again, the learning rate and momentum have been set to 10^{-4} and 0.9 respectively during weight optimization. The algorithm saves the parameters providing the largest average margin, and stops after 50 iterations without improvement over the recognition rate of the combination. The initial values of the parameters are set to $\alpha_i = 5/N$ and $b = -2.5$, as described by Menasri *et al.* (2012). An extra set could have been used for this step, for example by selecting some data from the training set, but, in preliminary experiments we noted that this decreased the performance of individual agents, leading to a poorer combination performance.

Table 5.1 Architecture of the neural networks

Handcrafted feature system architecture				
Hierarchy	Hor. samp.	Ver. samp.	Layers	Feature maps
Input	1	-	-	-
Level 1	2	-	2×20	20
Level 2	2	-	2×60	60
Level 3	1	-	2×180	-
Automatic feature system architecture				
Hierarchy	Hor. samp.	Ver. samp.	Layers	Feature maps
Input	3	4	-	-
Level 1	3	4	4×2	6
Level 2	2	4	4×10	20
Level 3	1	1	4×50	-

5.8 Results and discussion

5.8.1 Optimization results

We first verify that all the networks have enough capacity, that is, they have enough neurons to learn complex recognition models. The learning curves of the best repetition for all the features are shown in Figure 5.4. Note that all the networks have enough capacity, as the error on the training set keeps decreasing, even after the error on the validation set has reached its minimum. We show in Figure 5.6 the evolution of the margin between the confidences of the true word and the best impostor during weight optimization on the validation set. For both databases, the average margin with equal weights is 0.2 (iteration 1). It then increases quickly, and almost reaches its final value in less than 10 iterations. Note that the curves for the test sets follow the same trend as the curves for the validation sets. After convergence, the average margin reaches a value near 0.9 on both test sets. This is very high, as the maximum value is 1. Therefore, the confidence in the recognition is higher after the weight optimization than before.

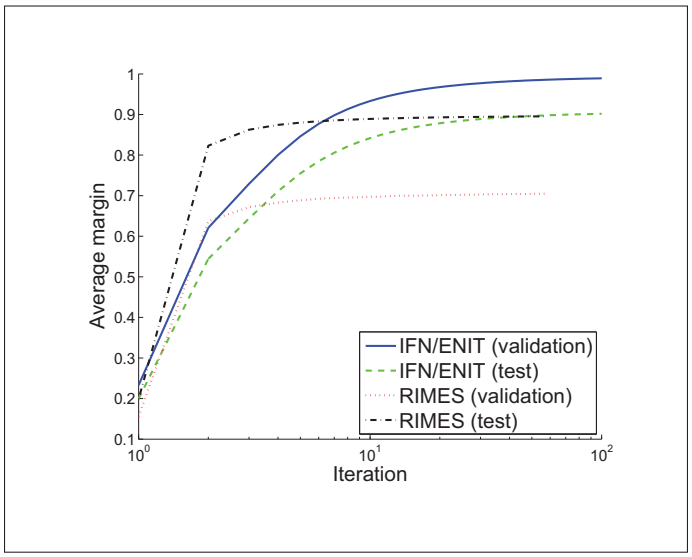


Figure 5.6 Margin evolution during weight optimization. (Iteration shown in log scale.)

5.8.2 Feature evaluation

The results of the feature evaluation are shown in Table 5.2. The average recognition rate of the agents of each feature, as well as the feature scores, are detailed. For both the IFN/ENIT and RIMES databases, the M-B feature obtains the best recognition rate, as well as the best score. This highlights the strength of this feature.

For the specific case of the IFN/ENIT database, the CCV feature set has the second highest score. This make sense, as this feature was specifically designed for Arabic script. It is followed by vision-based features (MDLSTM and LGH). Finally, in spite of having the second highest recognition rate, the R-M feature has the lowest score. This result, which seems surprising at first, is understandable, because the R-M feature is a subset of the M-B feature, and so its contribution when combined with the M-B feature is low. This allows us to make a point, which is that the feature score is not fully correlated with the recognition rate. This behavior is expected, as the objective of the score is to reflect the contribution of the various features in the combination scheme, unlike the recognition rate, which considers the agents individually. The ranking of the features according to their scores is similar on the RIMES database, although based on different values. Here we see that the vision-based methods are in the second and

third place (MDLSTM and LGH), followed by CCV, and finally R-M. Considering the observation on the two databases, we conclude that the M-B feature set is the most efficient, with good complementarity with vision-based features (LGH and MDLSTM). This result suggests that the current LSTM system is not able to extract abstract features, similar to distribution features, from vision-based features efficiently.

Results for individual agents are shown in Table 5.3. The best agent for both databases is based on the M-B feature, with a recognition rate of 93.8% and 90.7% for IFN/ENIT and RIMES respectively. Also note that the score of several agents is 0.0, or close to it. Weight optimization has filtered out the weak agents based on the MDLSTM feature for the IFN/ENIT database.

Table 5.2 Average recognition rate and score ($\sum s_i$) for each feature. (Best values highlighted.)

	IFN/ENIT		RIMES	
Feature	Rec. (%)	$\sum s_i$ (%)	Rec. (%)	$\sum s_i$ (%)
CCV	88.5 ± 0.6	23.0	87.6 ± 0.5	11.8
M-B	**93.2 ± 0.5**	**41.0**	**90.1 ± 0.3**	**35.0**
MDLSTM	83.3 ± 9.1	21.0	88.8 ± 1.0	21.6
R-M	91.6 ± 0.4	3.0	88.3 ± 0.5	10.1
LGH	89.7 ± 0.8	11.9	88.5 ± 0.3	21.5

5.8.3 Combination comparison

As a result of the feature evaluation, we obtain a recognition system based on the combination of several agents. In Table 5.4, we compare our approach with other combination methods: plurality vote, sum rule, and max rule, in which the agent with the highest confidence is selected. Our weighted vote approach provides the best results, with a slight improvement over the plurality vote. Although this improvement is negligible, it shows that our method is competitive in a system combination context, with the main advantage being to provide a score s_i for each system, unlike other methods. The improvement gained from the combination compared to the best agent is 1.7% and 4.1% for the IFN/ENIT and RIMES database respectively.

Table 5.3 Recognition rate and score (s_i) for each agent. (Best values per feature highlighted.)

IFN/ENIT							
Feature	Repetition		1	2	3	4	5
CCV	Rec.	(%)	88.2	88.8	**89.4**	88.1	88.2
	s_i	(%)	**8.9**	0.0	3.6	7.4	3.2
M-B	Rec.	(%)	92.7	92.9	92.9	93.5	**93.8**
	s_i	(%)	9.3	2.6	**14.0**	9.0	6.2
MDLSTM	Rec.	(%)	68.4	80.9	87.1	89.9	**90.0**
	s_i	(%)	0.0	0.0	5.9	7.1	**8.0**
R-M	Rec.	(%)	92.0	91.4	90.9	91.8	**91.8**
	s_i	(%)	0.8	**2.3**	0.0	0.0	0.0
LGH	Rec.	(%)	90.0	89.2	88.6	90.3	**90.5**
	s_i	(%)	1.8	0.0	**4.0**	2.9	3.2
RIMES							
Feature	Repetition		1	2	3	4	5
CCV	Rec.	(%)	88.0	87.1	87.2	**88.1**	87.7
	s_i	(%)	0.0	3.9	1.5	0.8	**5.6**
M-B	Rec.	(%)	90.0	89.9	89.9	**90.7**	90.1
	s_i	(%)	1.9	5.6	6.6	**12.3**	8.6
MDLSTM	Rec.	(%)	87.9	87.6	89.4	**89.7**	89.6
	s_i	(%)	1.9	3.0	7.3	**7.3**	2.2
R-M	Rec.	(%)	87.7	**89.1**	88.3	88.1	88.4
	s_i	(%)	3.2	**5.0**	0.5	0.0	1.4
LGH	Rec.	(%)	88.1	88.7	**88.7**	88.6	88.5
	s_i	(%)	3.1	1.0	**7.2**	4.5	5.8

We note that the confidence–based methods (sum rule and max rule) don't perform as well as the others. We also show the result for the Or rule, in which the recognition is considered successful if any of the agents provides the true word as output. It is near 99% for both databases, which shows the potential of the combination of agents using a more advanced scheme, such as the one proposed by Bianne-Bernard *et al.* (2011). In Figure 5.7, we show some of the images incorrectly recognized by all the agents, which are characterized by slanted handwriting or ambiguous character shapes.

We also perform an in-depth comparison of our approach with the more straightforward plurality vote. For this, we randomly select k agents out of the 25, and combine them using both

Table 5.4 Comparison of the recognition rate (%) of different combination methods. (Best values highlighted.)

Method	IFN/ENIT	RIMES
Weighted vote	**95.46**	**94.79**
Plurality vote	95.38	94.73
Sum rule	94.36	93.61
Max rule	93.93	92.75
Or rule	98.86	98.97

Figure 5.7 Sample images incorrectly recognized by all agents. (a) IFN/ENIT. (b) RIMES.

approaches. The parameter k varies from 1 to 25. The random selection is repeated 10 times for each k, and the recognition rates are averaged. The weights of the combination are optimized for each random selection. The results are shown in Figure 5.8. We observe that the results of both methods are very similar on both databases. The curve of the weighted combination is just slightly above the curve of the plurality vote. Therefore, weight optimization doesn't provide any significant advantage for recognition over the plurality vote. However, we note that the recognition rate increases as we combine more and more agents. The impact of adding

more agents decreases when the number of agents is already high, but the slope of the curve suggests that more agents will still improve the recognition.

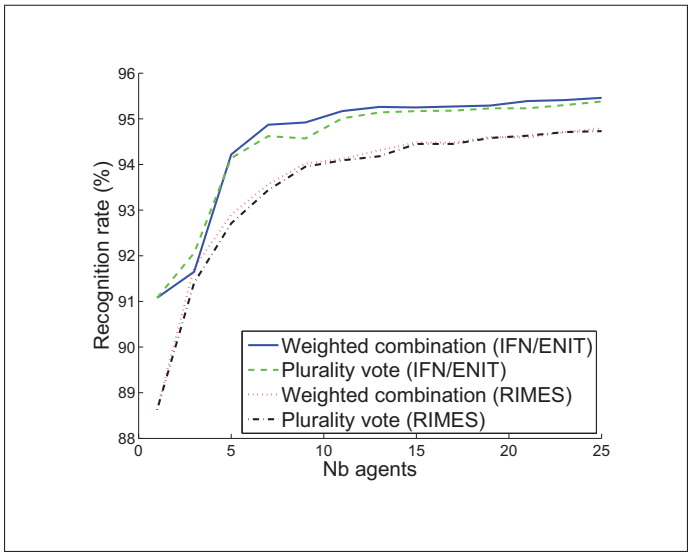


Figure 5.8 Comparison of the weighted combination with the plurality vote.

We also compare our approach with state–of–the–art systems (Table 5.5). All of these are based on either the HMM or LSTM model, or both. The TUM MDLSTM (Grosicki and El Abed, 2009) is the classic MDLSTM model, but with hyper parameters optimization (the size of hidden layers, etc.). The system of Menasri *et al.* (2012) is based on a weighted combination of 7 systems: one hybrid MLP-HMM, two tandem GMM-HMMs, and four MDLSTMs. The system of Bianne-Bernard *et al.* (2011) combines HMMs with and without context–dependent models using an MLP neural network. The system of Doetsch *et al.* (2012) is based on a tandem LSTM-HMM with horizontal positioning normalization. The system of Graves and Schmidhuber (2009) is the original MDLSTM architecture. Finally, the HMM system of Rothacker *et al.* (2012) is based on the bag–of–features model. The results show that the proposed approach is competitive with state–of–the–art methods, and actually obtains the best recognition rate on the IFN/ENIT database. Also note that, unlike most of the other systems, our approach is based on a single recognition method (RNN), and that no preprocessing is performed at image level, except for binarization. Finally, we compare our margin-based optimization with the method

provided by Menasri *et al.* (2012) on the RIMES database based on their results. Our method yields an accuracy improvement of 1.2%, or a relative error rate reduction of 18%, with the weighted vote compared to the sum rule. The method of Menasri *et al.* (2012) only yields an accuracy improvement of 0.5%, or a relative error rate reduction of 9.6%, with the weighted vote compared to the sum rule. We conclude that our method is better, although it was tested with different recognition systems.

Table 5.5 Comparison of the recognition rate (%) with other methods. (Best values highlighted.)

Method	IFN/ENIT	RIMES
TUM MDLSTM (Grosicki and El Abed, 2009)	-	93.2
Hybrid-HMMs/MDLSTM comb. (Menasri *et al.*, 2012)	-	**95.2**
HMMs comb. (Bianne-Bernard *et al.*, 2011)	-	89.1
LSTM-HMM Tandem (Doetsch *et al.*, 2012)	95.2	90.3
MDLSTM (Graves and Schmidhuber, 2009)	91.4	-
HMM (Rothacker *et al.*, 2012)	92.9	-
Proposed method	**95.5**	94.8

5.9 Conclusion

Features are a crucial component of analytical handwriting recognition systems. A large body of features is available in the literature, but no method is capable of quantifying both their efficiency and their complementarity. To fill this need, we proposed an advanced framework for feature evaluation using a combination scheme. Each feature is represented by an agent, which is an instance of a reference recognition system trained with that feature. The decisions of all the agents are combined using a weighted vote. The weights are optimized to maximize the recognition rate and are converted into scores assigned to the features for evaluation. We tested five features from four different feature categories: distribution, concavity, visual descriptor–based, and automatically learned. The results on Arabic and Latin word databases show that distribution features (Marti-Bunke feature) are the most efficient, with good complementarity with vision-based features (LGH and MDLSTM). In future work, this framework will be applied to guide the design of novel features, and it will be extended to compare the nature

of various recognition methods in terms of strength and complementarity, HMM with LSTM models, for example.

5.10 Acknowledgments

The authors thank the NSERC and SSHRC of Canada for their financial support.

CHAPTER 6

GENERAL DISCUSSION

This thesis has addressed the general problem of feature design for handwriting recognition. The literature reviewed in Chapter 1 showed the limitations of current features and their design methods for handwriting recognition. Two questions were specifically investigated: a) how to improve the description of Arabic word shapes for lexicon reduction and b) how to evaluate existing features for handwriting recognition. The general methodology described in Chapter 2 established three research objectives that led to the development of two original descriptors for Arabic word shape and a framework for evaluation of features used in handwriting recognition. First, a new method for lexicon reduction in Arabic script based on subword shapes was developed. Second, a new holistic descriptor for Arabic word shape was developed, with application to LR. Third, a framework for feature evaluation in handwriting recognition has been proposed. These methods made their own contributions and were presented, evaluated and discussed in Chapter 3, Chapter 4 and Chapter 5. The aspects studied in this thesis are independent and complementary. Together they form our general framework. They are now discussed in the following sections with a global perspective on their general advances made in the state of the art of handwriting recognition, with a focus on their strength and limitations.

6.1 Shape indexing based lexicon reduction framework for Arabic script

Few methods existed for LR in Arabic script and were based on subwords count and diacritics. Their knowledge about Arabic word shapes is extracted from ideal word templates. For example, to model an Arabic word by its number of subwords, the accurate spelling of the word would be used to count the number of subwords. Unlike these approaches, our method models Arabic word directly from the data. This has the advantage to consider the noise of the handwriting process, for example the omission of diacritics, which is frequent in practice. This has led to our database indexing framework for LR, where actual handwritten words are used as samples of word shapes. Indexing is efficiently performed by encoding Arabic word shape into a descriptor.

This has led to our first attempt to characterize the shape of Arabic subword into a descriptor for LR (Chapter 3). The shape is modeled as a weighted DAG based on its topology and structure, and a descriptor (W-TSV) is extracted based on the DAG adjacency matrix. The proposed approach has the advantage to be grounded on the solid TSV theoretical framework for (non-weighted) DAG indexing (Shokoufandeh *et al.*, 2005). Moreover, a formal analysis of the stability and robustness of the method has been provided. As this method is based directly on subwords shape, it is very competitive for LR at the subword level, unlike diacritic-based method. This is because the variety of diacritics is limited at the subword level and therefore they are not very discriminant. Moreover, the indexing scheme has a low computational complexity. Nevertheless, manipulating graph data structure is difficult and it results in implementation difficulties, which could potentially break the practical usage of the method. One major problem of this approach is the amount of information that the W-TSV vector can encode. Its construction is based on the weighted adjacency matrix, which ignores the label of the vertex, if any. In addition, the graph model is designed by an expert, which is a difficult task. Another issue is that subwords shape are identified as CCs of the image because in practice CCs can be under/oversegmented by the image binarization algorithm and therefore each CC doesn't always correspond to a subword, leading to a decrease of the performance. Finally, unlike existing approaches focusing on subword counts and diacritics, our method almost neglects this information. A number of these limitations have been addressed in our next descriptor.

6.2 Holistic descriptor of Arabic word shape for lexicon reduction

In Chapter 4, the goal of the proposed method was to provide a holistic descriptor for Arabic words, which not only describes the structure (geometry and topology) of individual subword shapes, but also emphasizes symbolic features of Arabic words such as subword counts and diacritics. For this purpose we developed the Arabic word descriptor (AWD) which is built in two stages, by first encoding subwords shapes into a descriptor (SD) using the BOW model, and combining all the SDs based on efficient heuristics.

Because this approach is based on the BOW model, rich features can be encoded in the SD. Moreover, it is possible to integrate new and possibly more powerful filters to extract features

from the subword skeleton pixels. One drawback of BOW models is that the spatial relationship between the considered pixels is lost. Nevertheless, we can assume that the spatial relationship in implicitly present in the visual word because we used filters with large scales (larger than the average subword size). The proposed method is simple to implement and has a low computational complexity, especially thanks to heuristics which avoid the difficult explicit classification of CC into subwords or diacritics. In addition, this approach provides with the best LR efficiency both at subword and word level. Despite these advantages, the proposed descriptor still has some limitations. Again, the identification of subwords relies on CC, therefore, this approach is dependent on the performance of the binarization algorithm, and on the writing style. Moreover, despite the fact that the AWD is performing well without considering the spatial relations between the CCs of the images, this information is missing from it and could potentially improve further the performance. Finally, as all LR methods, the proposed approach can induce WRS into errors that would not occur without LR. A mechanism should be provided to minimize this negative side-effect.

6.3 Holistic Arabic subword recognition

We tested our LR methods on a holistic classifier of Arabic subwords in Chapter 3 and Chapter 4. We chose a shape matching approach for the classification. The contour of a query subword shape is matched against all the shape contour of a labeled database, and the label of the most similar shape from the database is assigned to the query subword. Such approach is usually very computationally expensive for two reasons. First, shape matching requires computationally demanding registration and alignment to compensate for shape deformations or in our case, the handwriting variability. Second, repeating the matching over large databases increases the computation load. We have shown that by using an efficient LR approach to dynamically reduce the size of the database, shape matching approaches can perform subword recognition in single-writer historical documents with relatively low computation time, while maintaining high accuracy. For the sake of completeness, our article explaining the proposed contour alignment algorithm is presented in Appendix I.

6.4 Feature evaluation for handwriting recognition

Chapter 5 covered feature evaluation for handwriting recognition systems. A large body of features exist in the literature. They are inspired from different fields, so a straightforward comparison is difficult. Moreover, no method allows assessing the complementary nature of features. Therefore, we developed a framework for feature evaluation where each feature is assigned a score which represents its strength and complementarity with other features. This constitutes the main contribution of this work, and it is useful for the design of the next generation of features. In particular, our results showed that distribution features are the most efficient, and are complementary with visual-descriptor and automatically learned features. Moreover, the proposed scheme has a low computational overhead because the combination is at the decision level; it just requires WRS trained with a single feature. In particular, this is far less costly than the combination at the feature level because of the combinatorial number of combinations and the long convergence time required for the training of WRS. In addition, the resulting combination system using a weighted vote is competitive with other combination rules at the decision level. Nevertheless, the proposed framework also has some limitations. First, the weights are not regularized during optimization, for example a sparseness constraint would give a weight of 0 to non-relevant features and therefore highlight the important ones. Another limitation is that the method does not provide score for individual components of a feature. For example, the first 3 components of a feature vector are relevant while the 3 last are not. The proposed method will attribute a global score for the feature and will not go to the individual component level. Finally, the evaluation procedure is still dependent on the WRS architecture. This is because the most efficient and elegant approach for handwriting recognition is based on analytical recognition (character level recognition), and therefore requires a WRS to get any decent results from features. The downside of this dependency is that the feature evaluation is dependent on the actual WRS model used. It would be possible to think that the use of a different model with different strengths and weaknesses would provide different evaluation results.

6.5 Benchmarking of popular features for handwriting recognition

In Chapter 5 several features have been tested using the same benchmark, which comprise the same recognition engine, the same experimental protocol, and the same databases. Therefore, the performances of different features are directly comparable. This is not possible usually because experiments are conducted by different parties and the benchmark is always somehow different. We chose as WRS the recently proposed RNN which has outperformed the classic HMM on several sequence recognition tasks. Therefore, the proposed results are based on the state of art recognition engine and are aimed to improve it. Finally, using two databases based on Latin and Arabic scripts shows that our benchmark is not biased for any specific script, and it allows to identify the best features for cursive handwriting recognition in the broad sense.

GENERAL CONCLUSION

In this thesis, we have addressed the feature aspect and its impact on handwriting recognition systems. Features are the ground on which the recognition is built. Without efficient features, even state-of-the-art word recognition systems would produce modest results at best. Two distinct aspects of features for handwriting recognition have been studied in this thesis. We have introduced these aspects in a particular sequence to emphasize a proper methodology for feature design. First, when no features or very few features are available for a particular task, such as lexicon reduction for Arabic script, the expert of the field knowledge is important to build good features. Then, when enough features exist, and that it becomes difficult for an expert to comprehend their strength and complementarity, it is important to rely on automatic tool to assist the expert. This is typically the case of features for handwriting recognition.

For Arabic word recognition, the contribution of this thesis in term of holistic shape descriptor opens, or reopens, the direction for holistic Arabic word recognition. The last trend for Arabic word recognition relies on analytical recognition models with convincing results. Nevertheless, it totally ignores the specificity of Arabic script which integrates the subword unit. Therefore, direct recognition at the subword level is complementary to pure analytical model and a combination of both approaches has the potential to improve the overall recognition. Although the number of classes at the subword level is high, recognition is still feasible because the subwords frequency is usually high, as they constitute the building block of Arabic words.

Automatic feature evaluation for handwriting recognition system is an important tool for feature design. It can guide experts for the design of the next generation of features, which would be more efficient. Nevertheless, the level of automation can be pushed farther. Similarly to this tool replacing expert judgment, it is possible to think of a feature design algorithm replacing human experts. Such algorithm would consider the evaluation provided by our method to design new features. The new features would be evaluated themselves, providing the design algorithm with feedback to improve feature design.

Summary of contributions

In this section, we briefly highlight the major contribution of this thesis.

1. A new graph-indexing method for weighted graph has been introduced. It is grounded on a solid theoretical framework. It has provided a new framework for Arabic LR, based on subword shape matching and indexing. This is the first approach to consider the shape of subwords for LR, and it provides significant improvement at the subword level over existing methods, which are based on subword counts and diacritics.

2. The Arabic word descriptor (AWD), which is a novel descriptor of Arabic word have been designed for LR. It integrates subword shapes as well as symbolic information (subword counts and diacritics) into a single feature vector. This provides algorithmic efficiency as well as improved performances, with state-of-the-art results on two publicly available Arabic databases.

3. A framework for feature evaluation in handwriting recognition has been introduced. To the best of our knowledge, it is the first method to quantify feature performance and complementarity. The evaluation assigns a score to each feature, which is easy to interpret. This approach provided great insight on the strength of existing features, which will be useful for the design on future features.

Articles in peer reviewed journals

1. Youssouf Chherawala, Partha Pratim Roy and Mohamed Cheriet: Feature evaluation for offline handwriting recognition using supervised system weighting. *Submitted* to the IEEE Transactions on Pattern Analysis and Machine Intelligence (October 2013).

2. Youssouf Chherawala, Mohamed Cheriet: Arabic word descriptor for handwritten word indexing and lexicon reduction. *Submitted* to Pattern Recognition (May 2013).

3. Youssouf Chherawala, Mohamed Cheriet: W-TSV: Weighted topological signature vector for lexicon reduction in handwritten Arabic documents. Pattern Recognition Volume

45, Issue 9, September 2012, Pages 3277-3287. http://dx.doi.org/10.1016/j.patcog.2012.02.030

Articles in peer reviewed conference proceedings

1. Youssouf Chherawala, Partha Pratim Roy and Mohamed Cheriet (2013): Feature design for offline Arabic handwriting recognition: handcrafted vs automated? In Proceedings of the 12th International Conference on Document Analysis and Recognition (ICDAR '13), Washington DC, USA, pp. 290-294. http://dx.doi.org/10.1109/ICDAR.2013.65

2. Guoqiang Zhong, Youssouf Chherawala and Mohamed Cheriet (2013): An Empirical Evaluation of Supervised Dimensionality Reduction for Recognition. In Proceedings of the 12th International Conference on Document Analysis and Recognition (ICDAR '13), Washington DC, USA, pp. 1315-1319. http://dx.doi.org/10.1109/ICDAR.2013.266

3. Youssouf Chherawala, Robert Wisnovsky and Mohamed Cheriet (2012): Sparse descriptor for lexicon reduction in handwritten Arabic documents. In Proceedings of the 21th International Conference on Pattern Recognition (ICPR '12), Tsukuba Science City, Japan, pp. 3729-3732. http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=6460975

4. Youssouf Chherawala, Mohamed Cheriet (2012): Shape recognition on a Riemannian manifold. In Proceedings of the 11th International Conference on Information Sciences, Signal Processing and their Applications: Special Sessions (ISSPA2012: Special Sessions), Montreal, Canada, pp. 1205-1210. http://dx.doi.org/10.1109/ISSPA.2012.6310475

5. Youssouf Chherawala, Robert Wisnovsky and Mohamed Cheriet (2011): topological signature vector-based lexicon reduction for fast recognition of pre-modern Arabic subwords, in: Proceedings of the 1st Workshop on Historical Document Imaging and Processing (HIP '11), Beijing, China, pp. 6-13. http://dx.doi.org/10.1145/2037342.2037345

6. Yoshua Bengio, Frédéric Bastien, Arnaud Bergeron, Nicolas Boulanger-Lewandowski, Thomas M. Breuel, Youssouf Chherawala, Moustapha Cisse, Myriam Côté, Dumitru Erhan, Jeremy Eustache, Xavier Glorot, Xavier Muller, Sylvain Pannetier Lebeuf, Razvan Pascanu, Salah Rifai, François Savard, Guillaume Sicard (2011): Deep Learners Benefit More from Out-of-Distribution Examples. Journal of Machine Learning Research - Proceedings Track 15 (AISTATS '11 Proceedings), Fort Lauderdale, FL, USA, 2011 , pp. 164-172. http://www.jmlr.org/proceedings/papers/v15/bengio11b/bengio11b.pdf

Awards

École de Technologie Supérieure (ÉTS), Internal Scholarship (2012).

Conference organization

IEEE 11th International Conference on Information Sciences, Signal Processing and their Applications (ISSPA '12), volunteer.

Paper reviewing

- 9th International Workshop on Systems, Signal Processing and their applications (WOSSPA '13) (4 papers)
- 10th IAPR International Workshop on Document Analysis Systems (DAS '12) (2 papers)
- IEEE Transactions on Pattern Analysis and Machine Intelligence (1 paper)

APPENDIX I

SHAPE RECOGNITION ON A RIEMANNIAN MANIFOLD

Youssouf Chherawala, Mohamed Cheriet

Synchromedia Laboratory, École de Technologie Supérieure
1100 Notre-Dame Ouest, Montréal, QC, Canada

Published in the Proceeding of ISSPA '12
July 2-5 2012, Montreal, Canada, Pages 1205-1210

Abstract

In this paper, we propose to perform shape recognition on a Riemannian manifold. Shape representation on a manifold has the advantage to be intrinsically invariant to shape preserving transformation, such as scaling and translation. Also, shape distance can be naturally computed because Riemannian manifolds are metric spaces. We propose to use the square-root velocity manifold (SRV), which model the shape external contour as a unit-length curve. We detail a dynamic programming algorithm for curve alignment w.r.t. parameterization, which respects the unit-length constraint. Then, we increase the robustness of the SRV representation to shape deformations with additional features. In order to be resilient to occlusion, the distance between two curves is performed in two steps. First, the curves are aligned and the less matching parts are removed; then the resulting curves are aligned and the distance is evaluated. Finally, a support vector machine classifier is trained based on the pairwise shape distance for a robust recognition. Promising results are obtained using state-of-the-art benchmarks.

Keywords

Shape recognition, Manifold, Dynamic programming, SVM.

1 Introduction

Shape recognition is an important problem in computer vision. It aims to recognize objects present in a natural scene. Several features are representative of an object, such as color, texture or shape. Most of the approaches focus on the latter, as it is a powerful cue for recog-

nition. From 2D images of a scene, it is not possible to obtain the complete shape of a 3D object because of the projection process involved. In fact, only the silhouette on an object can be extracted after the segmentation of the scene, and it is used as an approximation of the object shape. The silhouette of an object may suffer for partial occlusion and change of scale, depending on the scene configuration, but it is also subject to great variation if the object is articulated. In order to solve these problems, shape representations are made invariant to certain transformations, such as scaling, rotation and translation. Also, the shape recognition process must be tolerant to shape deformations and articulations.

Shape recognition is based on shape matching. Given two shapes, the correspondence between similar parts is found, and a distance is derived based on the quality of the matching. The recognition process uses the pairwise distance between shapes. Its performance can be further increased by using machine learning methods.

Several representations have been proposed for natural and man-made object shapes. They are based on the silhouette contour, which is a simple (non-self-intersecting) closed curve, as the silhouette doesn't have holes. Shock graph representations (Siddiqi *et al.*, 1999; Sebastian *et al.*, 2004) are based on the shape medial axis, which can be seen as the singularities formed during the contour curve evolution under 'motion by curvature'. The medial axis is decomposed into segments and represented as shock graph, which encodes the structure and the geometry of the shape. Gorelick *et al.* (2006) combine contour-based features with the notion of random walks. They assign to each point of the shape the mean time required by a walk starting at that point to reach the contour. The shape recognition is performed by the extraction of weighted moments from the shape. The shape context feature (Belongie *et al.*, 2002) samples points from the shape contour. A context is assigned to each sampled point, represented by the distribution of the relative position of the other points. Given two shapes, the points correspondence is found using a bipartite graph matching algorithm, and their distance is found based on the quality of the shape context alignment. The shape context can be made invariant to scale and rotation. Ling and Jacobs (2007) proposed the inner-distance descriptor which provides resilience to articulation and capture parts structure. The inner-distance is

defined as the length of the shortest path between landmark points, constrained to pass within the shape silhouette. Daliri and Torre (2008) combine shape context features with dynamic programming to recover shape correspondence. Shapes are then aligned and transformed into string of symbols to evaluate their similarity. Finally, they used a kernel-edit distance in order to improve their recognition results with an SVM classifier (Daliri and Torre, 2010).

In this work, we use the square-root velocity (SRV) representation, which is invariant to scaling and rotation (Srivastava *et al.*, 2011). It is based on the silhouette curve. The SRV representation has been augmented with additional features based on the silhouette center of mass, in order to be resilient to articulation-based deformations. A dynamic programming (DP) algorithm respecting the scaling invariance property has been devised for shape alignment. Matching is made robust to occlusions by removing in a first step the less matching parts. Finally, a support vector machine classifier has been used to improve the recognition performance. The outline of this work is presented in Figure I-1. The first contribution of this work is to use the SRV representation in the context of natural shape recognition. It will be shown that with slight modification, the SRV provides a powerful and unified framework for shape recognition. The second contribution is to provide a DP algorithm for curve alignment, respecting the SRV representation unit-length constraint.

The organization of this paper is as follows. First, we introduce the SRV representation in section 2. Then, we present our algorithm for shape alignment in section 3, followed by our shape recognition framework in section 4. Finally, the details of our experiments are given in section 5 and the conclusions in section 6.

2 Square-root velocity representation (SRV)

The square-root velocity (SRV) representation is a manifold for shapes. Each shape is represented by its external contour, which is a simple (non-self-intersecting) closed curve. For simplicity, the SRV is detailed here for the case of open curves; for more detail the reader is referred to Srivastava *et al.* (2011). The SRV allows shape matching, while being invariant to translation and scaling by embedding the contour curve of the shapes on an appropriate man-

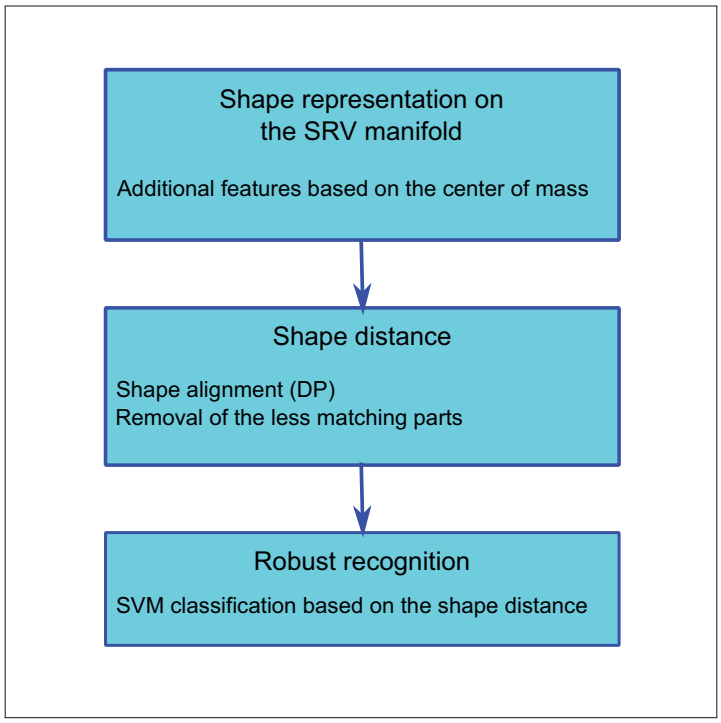


Figure-A I-1 Outline of our framework.

ifold. The curve is defined on the $\mathbb{L}^2$ Hilbert space, and has value in the $\mathbb{R}^n$ Euclidean space, where $n = 2$ for 2D curves. It is parameterized by t over the domain $D = [0, 1]$. First, the contour curve f is normalized to a unit length, in order to remove the effect of scale. The curve is then represented using the SRV representation:

$$q(t) = \dot{f}(t) / \sqrt{\left\| \dot{f}(t) \right\|} \tag{A I-1}$$

This representation is invariant to translation as it is based on the derivation of f. It also preserves the unit-length constraint on f:

$$\int_D \|q(t)\|^2 dt = \int_D \left\| \dot{f}(t) \right\| dt = 1 \tag{A I-2}$$

Therefore, the set of all curves under the SRV representation forms a unit hypersphere in $\mathbb{L}^2$.

The geodesic distance between two curves q_1 and q_2 is simply defined as $d(q_1, q_2) = \arccos(\langle q_1, q_2 \rangle)$. As the SRV representation forms a Riemannian manifold, the geodesic distance is a metric. In order to build the shape space, the metric must be invariant to rotation and it should accommodate to elastic deformation. Elastic deformations are modeled by the re-parameterization of the original curve f, i.e. $f \circ \gamma(t)$ is the re-parameterization of f by $\gamma \in \Gamma$, the set of all orientation-preserving diffeomorphisms of D. As the actions of the rotation group $SO(n)$ and re-parameterization group Γ act by isometry on the SRV representation, a quotient space is built such that an orbit of a curve q is given by $[q] = O(q \circ \gamma)\sqrt{\dot{\gamma}} | (\gamma, O) \in \Gamma \times SO(n)$. Therefore, the geodesic distance between two curves in the shape space is given by:

$$d([q_1], [q_2]) = \inf_{(\gamma, O) \in \Gamma \times SO(n)} d\left(q_1, O(q_2 \circ \gamma)\sqrt{\dot{\gamma}}\right)$$

3 SRV curves alignment

In this section we present our approach for SRV curve alignment. In particular, we detail an algorithm for optimal curve parameterization under unit-length constraint. Given two SRV representations, the best curve alignment is sought, in order to decrease the influence of shape deformation on the recognition process. The best rotation and re-parameterization must be simultaneously found in order to minimize the geodesic distance. Nevertheless, no closed form solutions exist for this problem so far. A gradient descent algorithm has been proposed by Srivastava *et al.* (2011), but its alignment performance was inferior to that of dynamic programming (DP). Unlike the DP algorithm proposed by Mio *et al.* (2007), our algorithm is directly applicable to the SRV representation, without decomposing it into 'speed' and 'orientation' functions; also it provides a more symmetric treatment to both curves. As DP procedure is only applicable on open curves, the closed curve will be considered as an open curve.

Therefore, the alignment is broken into two steps: first the curves are aligned with respect to rotation and parameterization origin, then they are opened at the best origin and re-parameterize with DP.

For the first step, an arbitrary point is chosen on the first curve and the best origin is exhaustively searched on the second curve. The best origin is the one that minimizes the geodesic distance between the two curves, after removing the action of the rotation group. Given two open curves, the optimal rotation can be efficiently found using Procrustes analysis (Dryden and Mardia, 1998).

For the second step, DP is used to align both curves by re-parameterization. For a parameterization γ, if $\dot{\gamma}(t) > 1$ for a given $t \in D$, the curve is locally 'compressed' w.r.t. t, while if $\dot{\gamma}(t) < 1$ it is locally 'stretched'. In practice, this is done by insertion or deletion of curve points, if we make analogy with the string edit distance. Deletion corresponds to 'compression' while insertion corresponds to 'stretching'. A completely symmetric formulation of curve alignment allows the mapping of a segment of a given curve with a single point of the other curve. This is usually achieved by the deletion of curves points or segments (Sebastian *et al.*, 2003). As the manifold is a hypersphere, minimizing the geodesic distance is equivalent to minimizing the $\mathbb{L}^2$ distance $d_{\mathbb{L}^2} = \|q_1 - q_2\|$. For convenience the square of $d_{\mathbb{L}^2}$ will be minimized with DP. In the discrete setting, each curve q_m ($m \in \{1, 2\}$) is represented by n points $q_{m,i}$ such that $1 \leq i \leq n$ and the first point match the last point ($q_{m,0} = q_{m,n}$). Assuming the trapezoidal rule for integration, the squared $\mathbb{L}^2$ distance is defined as:

$$d^2_{\mathbb{L}^2}(q_1, q_2) = \sum_{i=1}^{n-1} \langle d_i, d_i \rangle \Big/ (n-1) \tag{A I-3}$$

Where $d_i = q_{1,i} - q_{2,i}$. Nevertheless, two problems appear during the SRV curves alignment with DP. First, the action of re-parameterization on the SRV curve doesn't act by isometry, and thus doesn't preserve its norm. Second, during the re-parameterization, curve points are inserted or removed, which changes the denominator of the squared distance numerical calculation. As this number is not known beforehand, DP algorithm can't be applied in general because of the latter normalization problem (Marzal and Vidal, 1993).

Nevertheless, if we consider a special case, where only stationary points (i.e. $\|q_{m,i}\| = 0$ or $\dot{\gamma}(t) = 0$) are inserted to a curve q_m, both of these problems compensate each other's. This

operation corresponds to map a segment of a curve to a single point of the other curve and is, therefore, completely symmetric. After the insertion of stationary points, γ is not anymore invertible and thus doesn't belong to Γ, but this is not a problem as both curves are simultaneously re-parameterized. We start by giving the details of the algorithm before proving its validity in Proposition 1. An $n \times n$ grid is built where the axes correspond to the sampled points of the curve. The cost associated with the curves 'editions' are as follows:

- substitution of a point $q_{1,i}$ with $q_{2,j}$ is
  $\langle q_{1,i} - q_{2,j}, q_{1,i} - q_{2,j} \rangle$,
- insertion of a stationary point at the location of a point $q_{m,i}$ is $\langle q_{m,i}, q_{m,i} \rangle$ for $m \in \{1, 2\}$.

If we assume q_1 is represented along the rows of the grid and q_2 along the columns, the substitution of points corresponds to a diagonal displacement, the insertion of a stationary point to q_1 to a displacement to the right and the insertion of a stationary point to q_2 to a displacement to the bottom (Figure I-2). We restrict all the displacement to be between neighboring cells of the grid. During DP, the first and the last points of each curve are matched. The insertion of k stationary points to one curve means the insertion of k stationary points to the other curves, in order to maintain equal number of points. We now prove the validity of this algorithm for SRV curves alignment:

Proposition 1. *The optimal path on the DP grid corresponds to the optimal alignment between curves q_1 and q_2 by insertion of stationary points.*

Proof. The squared norm of a SRV curve q_m is equal to 1. The re-parameterization of this curve into q'_m by the addition of k stationary points has a scaling effect: $\|q'_m\| = \sqrt{(n-1)/(n+k-1)}$. For optimal alignment of the curve, we want to minimize the squared distance between the curves q'_1 and q'_2 normalized to unit norm: $\sum_{i=1}^{n+k-1} \langle d'_i, d'_i \rangle \Big/ (n-1)$ where $d'_i = q'_{1,i} - q'_{2,i}$. We can observe that the normalization of the integral is independent of k. Hence, the normalization of the curves compensates the normalization of the integral. Also, the numerator of the last equation corresponds to the cost of a path on the DP grid. Therefore, the optimal path

on the DP grid corresponds to the optimal alignment between curves q_1 and q_2 by insertion of stationary points. □

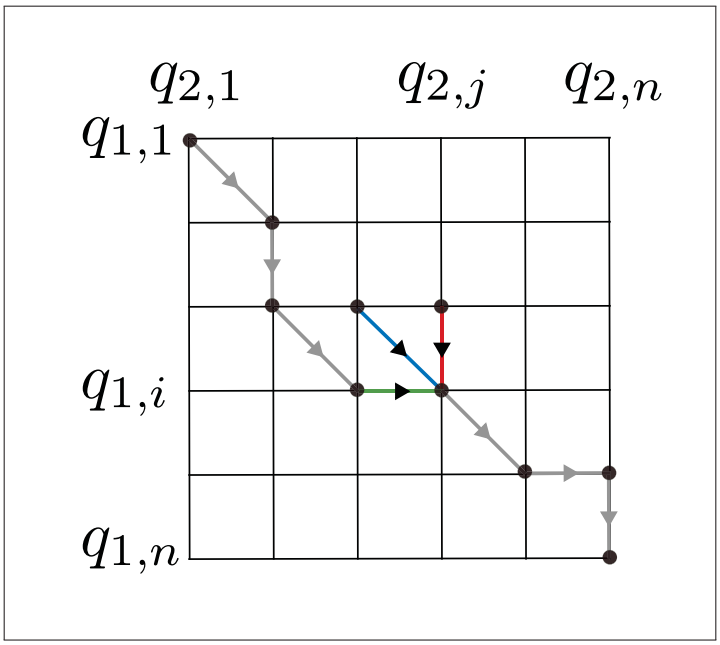


Figure-A I-2 SRV curves alignment with dynamic programming. Gray: optimal path on the grid; blue: substitution of $q_{1,i}$ with $q_{2,i}$; green: insertion of a stationary point to q_1 after $q_{1,i}$ at the location of $q_{2,j}$; red: insertion of a stationary point to q_2 after $q_{2,j}$ at the location of $q_{1,i}$.

4 Shape recognition

In this section we detail the features we use to build the SRV curves. Then, we show how the distance between two SRV curves is obtained for the case of natural object silhouette. Finally, we present our robust recognition strategy using an SVM classifier.

4.1 Features for the SRV representation

The SRV representation tolerates elastic deformation, nevertheless it remains sensitive to articulation, which often occurs in natural and man-made objects. In order to alleviate this limitation, we propose two additional SRV representations. The first one, the $\mathrm{SRV}_{\mathrm{Euclid}}$ is based

on the Euclidean distance of the silhouette curve points to the silhouette curve center of mass; this feature tends to preserve the shape external boundaries. The second one, the $\mathrm{SRV}_{\mathrm{inner}}$ is based on the inner distance of the silhouette curve points to the silhouette curve center of mass; this feature tends to be insensitive to shape articulation. These two SRVs are based on 1D curves unlike the classical SRV, therefore, they don't require rotational alignment. Also, they are defined with respect to a fixed reference point (the center of mass), while the classical SRV is represented relatively to the previous points on the silhouette. The reference point must provide additional stability. During shape matching, each of these 3 representations, namely SRV, $\mathrm{SRV}_{\mathrm{Euclid}}$ and $\mathrm{SRV}_{\mathrm{inner}}$ will be aligned separately and they will provide a distance, respectively $\mathrm{dist}_{\mathrm{SRV}}$, $\mathrm{dist}_{\mathrm{Euclid}}$ and $\mathrm{dist}_{\mathrm{inner}}$. Also, the combination of these distances is considered as follows:

$$\mathrm{dist}_{\mathrm{combined}} = \mathrm{dist}_{\mathrm{SRV}} + \mathrm{dist}_{\mathrm{Euclid}} + \mathrm{dist}_{\mathrm{inner}}$$

These concepts are illustrated in Figure I-3, where the 3 features for the shape silhouette are shown, namely the silhouette contour, the Euclidean distance curve and the inner distance curve. The center of mass may seem at a low position with respect to the shape, but as already mentioned, it represents the center of mass of the silhouette contour and not of the silhouette 'body'.

4.2 Pairwise shape distance computation

The distance between two SRV representations is found as follows. First, the two shapes are aligned, then a given amount of less matching points in the least-squares sense are removed from both curves. The removal of these points can be interpreted as re-parameterization of the curves. The resulting curves are projected back on the manifold and aligned once again. The latter curves are used to compute the pairwise geodesic distance.

4.3 Robust classification with SVM

The support vector machine (SVM) (Burges, 1998) is an algorithm for binary classification, i.e. for problems where there are only two classes. The SVM is a linear classifier, in which

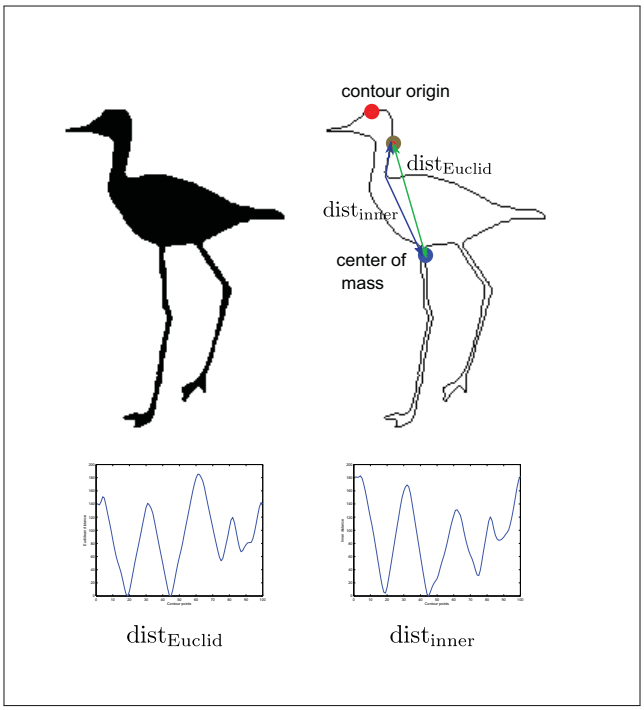


Figure-A I-3 Features for the SRV representation. From top-left to bottom right: Shape silhouette, silhouette contour, Euclidean distance from the center of mass and inner distance from the center of mass.

Algorithm 3 Computation of the Shape distance

Input: SRV representation of two shapes
Output: pairwise shape distance
Align the SRV curves
Remove the less matching curve points
Project the curves back on the manifold
Align the resulting curves
Compute the geodesic distance

the two classes are separated by a hyperplane. The SVM is extended to multi-class problems by converting them into multiple binary classification problems. The optimal hyperplane is defined as the one with the largest distance from the nearest training point. To solve problems which are not linearly separable, the input vector are first mapped in a higher dimensional space (possibly infinite dimensional) in which the classes can be easily separated. Here, we

will embed the SRV representation using the Gaussian kernel with the geodesic distance:

$$k\left(q_1, q_2\right) = \exp\left(-\gamma \cdot d([q_1], [q_2])^2\right)$$

Where here γ is a free parameter ($\gamma > 0$). The distance d can be any of the defined SRV distances. This kernel is semi-definite positive because the geodesic distance is a metric.

5 Experiments

We have evaluated the 3 SRV representations, namely SRV, $\mathrm{SRV}_{\mathrm{Euclid}}$ and $\mathrm{SRV}_{\mathrm{inner}}$ separately and combined together. The SRVs are computed from the uniform sampling of 100 points of the silhouette contour. If the center of mass of the contour points fall outside of the silhouette, it is approximated by the nearest contour point for the computation of the inner distance. Hence, in some cases the center of mass for the computation of $\mathrm{dist}_{\mathrm{Euclid}}$ is different from that of $\mathrm{dist}_{\mathrm{inner}}$. The best parameterization origin is sought every 3 points for rotation and origin alignment. During the DP optimization, we limit the grid search to the addition of 10% of stationary points. Also, during the computation of the SRV distance, we remove up to 20% of the less matching points.

The natural silhouettes database[1] has been used for evaluation. It is composed of 490 silhouettes of natural and man-made objects (Figure I-4), divided into 12 classes. We use the 1-NN and the SVM classifier for the recognition. The database is randomly divided into training and testing sets, respectively of size 396 and 94. For the SVM classifier experiment, the best values for the soft margin parameter C and γ are found by grid search, using a 5-fold cross validation. The search interval for C and γ are respectively $\left[2^{-2}, 2^{7}\right]$ and $\left[2^{-5}, 2^{5}\right]$. This process is repeated 100 times and the recognition rates of all the repetitions are averaged. The results are shown in Table I-1. First, we compare the 3 SRV representations using the 1-NN classifier. We notice the average error rate of the SRV (2.6%) is better than that of $\mathrm{SRV}_{\mathrm{Euclid}}$ and $\mathrm{SRV}_{\mathrm{inner}}$. However, if we combine the distance of these 3 representations, the average error rate decreases to 1.8%. The SVM classifier, which use label information during the training phase for robust

[1] http://www.csd.uwo.ca/~ygorelic/downloads.html

recognition, decreases the average error rate for almost all representations compared to the 1-NN. In particular, for the combination of the 3 SRV distances, the error rate decreases to 1.3%. The comparison of this last result with that of Daliri and Torre (2010) (Table I-2) shows they are comparable. Under the t test, the difference between these two results is considered to be not statistically significant.

Our framework has also been evaluated on the MPEG-7 shape database. It is composed of 1400 shapes (Figure I-5), equally divided into 70 classes. For this database, the evaluation is done using a leave-one-out strategy, where alternatively each shape constitutes the test set and the remaining shapes the training set. The classification results for all the shapes are then averaged (Table I-3). Similar observation holds for this database too, that is the combination of the 3 SRV distances performs the best. The performance of our approach is encouraging, even if it is inferior to that of Daliri and Torre (2010) (Table I-4). Examples of mismatched shapes with the 1-NN and the combined distance are shown in Figure I-6.

The method of Daliri and Torre (2010) is very similar to ours (see description in Section 1). The correspondence between two shapes contour points is found by dynamic programming, based on their shape context distance. Then, selected contour points are aligned by Procrustes analysis. Finally, the contour points are transformed into string of 'symbols', and the classification is performed by an SVM based on the string-edit distance. The differences with our approach are the following, they allow reflection (mirroring) during shapes alignment, and they use a multi-resolution scheme. The pairwise shape distances from multiple resolutions are averaged to provide the final distance. Such refinements can be incorporated in our approach and would improve our results. The main advantage of our method is to use a single framework, the SRV manifold, while Daliri and Torre (2010) resort to multiple concepts. Therefore, our implementation is simpler and has fewer parameters.

6 Conclusions

In this work, we proposed a method to perform robust shape recognition on a Riemannian manifold. For this purpose, we used the SRV representation, and we derived two more features

Figure-A I-4 Natural silhouettes database. One example from each class is shown.

Tableau-A I-1 Comparison of the average recognition error rate (%) on the Natural Silhouettes database

Meth.	SRV	$\text{SRV}_{\text{Euclid}}$	$\text{SRV}_{\text{inner}}$	Combined
1-NN	2.6 ± 1.6	7.0 ± 3.0	6.1 ± 2.5	1.8 ± 1.4
SVM	1.9 ± 1.6	6.2 ± 2.6	6.2 ± 2.4	1.3 ± 1.3

for it. Furthermore, we proposed a dynamic programming algorithm for optimal alignment of SRV curves through re-parameterization. Robust recognition is obtained using an SVM classifier. The result achieved by our approach is comparable to that of state-of-the-art methods. The main advantage of the SRV representation is to provide a unified framework for all shape

Tableau-A I-2 Comparison of the average recognition error rate (%) on the Natural Silhouettes database with another approach

Method	Average error rate
Kernel edit distance (Daliri and Torre, 2010)	$1.29 \pm 1.24\%$
proposed method	$1.32 \pm 1.28\%$

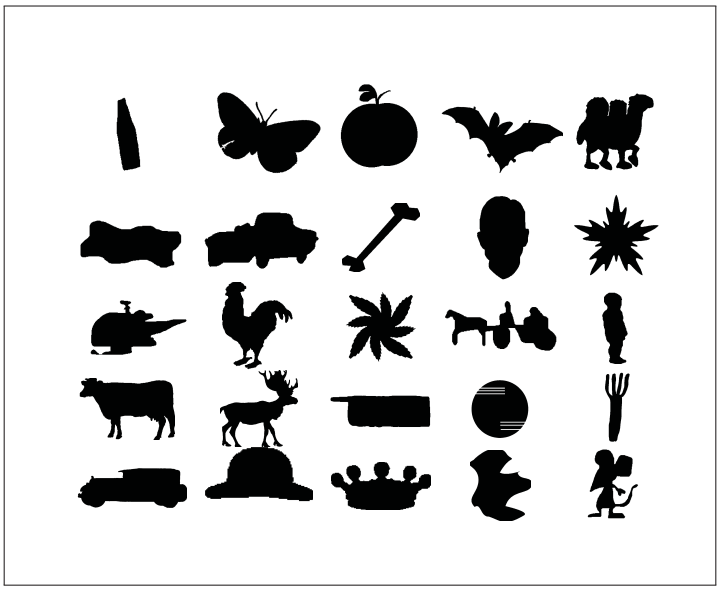


Figure-A I-5 MPEG-7 shape database, 25 sample shapes are shown.

Tableau-A I-3 Comparison of the recognition error rate (%) on the MPEG-7 shape database

Meth.	SRV	$\mathrm{SRV}_{\mathrm{Euclid}}$	$\mathrm{SRV}_{\mathrm{inner}}$	Combined
1-NN	4.64	6.21	10.86	3.43
SVM	4.57	7.14	9.50	2.50

recognition steps. In future work, we will improve the robustness of our method by allowing shape reflection during alignment, and by using a multi-resolution approach.

7 Acknowledgments

Tableau-A I-4 Comparison of the recognition error rate (%) on the MPEG-7 shape database with another approach

Method	Error rate
Kernel edit distance (Daliri and Torre, 2010)	1.07%
proposed method	2.50%

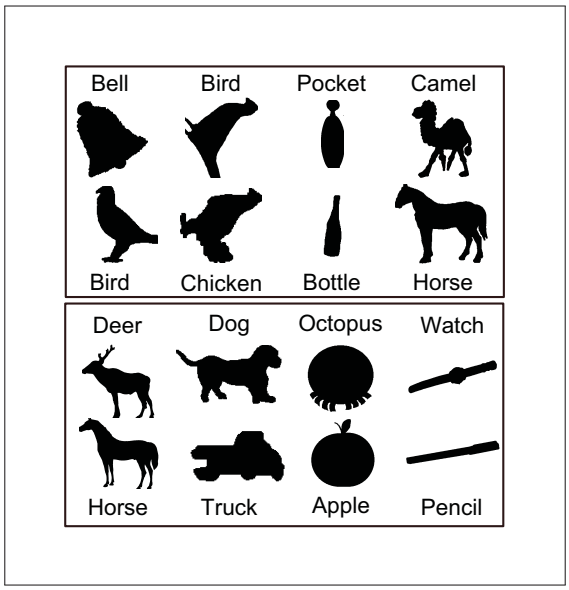


Figure-A I-6 Mismatched shapes from the MPEG-7 shape database, using a 1-NN with the combined distance.

The authors would like to thank NSERC of Canada for their financial support.

BIBLIOGRAPHY

Abuhaiba, Ibrahim S. I., Sabri A. Mahmoud, and Roger J. Green. 1994. "Recognition of handwritten cursive Arabic characters". *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 16, n° 6, p. 664–672.

Adamek, Tomasz, Noel E. O'Connor, and Alan F. Smeaton. 2007. "Word matching using single closed contours for indexing handwritten historical documents". *International Journal of Document Analysis and Recognition*, vol. 9, n° 2–4, p. 153–165.

Al-Hajj Mohamad, Ramy, Laurence Likforman-Sulem, and Chafic Mokbel. July 2009. "Combining Slanted-Frame Classifiers for Improved HMM-Based Arabic Handwriting Recognition". *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 31, n° 7, p. 1165–1177.

Arthur, David and Sergei Vassilvitskii. 2007. "k-means++: the advantages of careful seeding". In *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms (SODA '07)*. (Philadelphia, PA, USA 2007), p. 1027–1035. Society for Industrial and Applied Mathematics.

Asi, Abedelkadir, Jihad El-Sana, and Volker Märgner. July 2012. "Hierarchical Scheme for Arabic Text Recognition". In *Proceedings of the 11th International Conference on Information Sciences, Signal Processing and their Applications: Special Sessions (ISSPA2012: Special Sessions)*. p. 1299–1304.

Azeem, Sherif Abdel and Hany Ahmed. November 2012. "Off-Line Arabic Handwriting Recognition System Based on Concavity Features and HMM Classifier". In *Proceedings of the 21th International Conference on Pattern Recognition (ICPR '12)*. p. 705–708.

Baum, Leonard E., Ted Petrie, George Soules, and Norman Weiss. 1970. "A Maximization Technique Occurring in the Statistical Analysis of Probabilistic Functions of Markov Chains". *The Annals of Mathematical Statistics*, vol. 41, n° 1, p. 164–171.

Bay, Herbert, Andreas Ess, Tinne Tuytelaars, and Luc Van Gool. June 2008. "Speeded-Up Robust Features (SURF)". *Computer Vision and Image Understanding*, vol. 110, n° 3, p. 346–359.

Belongie, Serge, Jitendra Malik, and Jan Puzicha. 2002. "Shape matching and object recognition using shape contexts". *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, n° 4, p. 509–522.

Bertolami, Roman, Christoph Gutmann, Horst Bunke, and A. Lawrence Spitz. September 2008. "Shape Code Based Lexicon Reduction for Offline Handwritten Word Recognition". In *Proceedings of the Eighth IAPR International Workshop on Document Analysis Systems (DAS '08)*. p. 158–163.

Bianne-Bernard, Anne-Laure, Farès Menasri, Ramy Al-Hajj Mohamad, Chafic Mokbel, Christopher Kermorvant, and Laurence Likforman-Sulem. 2011. “Dynamic and Contextual Information in HMM Modeling for Handwritten Word Recognition”. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, n° 10, p. 2066–2080.

Burges, Christopher J. C. June 1998. “A Tutorial on Support Vector Machines for Pattern Recognition”. *Data Mining and Knowledge Discovery*, vol. 2, p. 121–167.

Carbonnel, Sabine and Eric Anquetil. 2004. “Lexicon Organization and String Edit Distance Learning for Lexical Post-Processing in Handwriting Recognition”. In *Proceedings of the Ninth International Workshop on Frontiers in Handwriting Recognition (IWFHR ’04)*. (Washington, DC, USA 2004), p. 462–467. IEEE Computer Society.

Cheriet, Mohamed and Ching Y. Suen. 1993. “Extraction of key letters for cursive script recognition”. *Pattern Recognition Letters*, vol. 14, n° 12, p. 1009–1017.

Chherawala, Youssouf and Mohamed Cheriet. 2012a. “W-TSV: Weighted topological signature vector for lexicon reduction in handwritten Arabic documents”. *Pattern Recognition*, vol. 45, n° 9, p. 3277–3287.

Chherawala, Youssouf and Mohamed Cheriet. July 2012b. “Shape Recognition on a Riemannian Manifold”. In *Proceedings of the 11th International Conference on Information Sciences, Signal Processing and their Applications: Special Sessions (ISSPA2012: Special Sessions)*. p. 1205–1210.

Chherawala, Youssouf, Robert Wisnovsky, and Mohamed Cheriet. 2011. “TSV-LR: topological signature vector-based lexicon reduction for fast recognition of pre-modern Arabic subwords”. In *Proceedings of the 1st Workshop on Historical Document Imaging and Processing (HIP ’11)*. p. 6–13.

Chherawala, Youssouf, Robert Wisnovsky, and Mohamed Cheriet. November 2012. “Sparse Descriptor for Lexicon Reduction in Handwritten Arabic Documents”. In *Proceedings of the 21th International Conference on Pattern Recognition (ICPR ’12)*. p. 3729–3732.

Chherawala, Youssouf, Partha Pratim Roy, and Mohamed Cheriet. 2013. “Feature design for offline Arabic handwriting recognition: handcrafted vs automated?”. In *Proceedings of the 12th International Conference on Document Analysis and Recognition (ICDAR ’13)*. (Washington, DC, USA 2013), p. 290–294.

Côté, Myriam, Eric Lecolinet, Mohamed Cheriet, and Ching Y. Suen. 1998. “Automatic reading of cursive scripts using a reading model and perceptual concepts”. *International Journal on Document Analysis and Recognition*, vol. 1, n° 1, p. 3–17.

Dalal, Navneet and Bill Triggs. 2005. “Histograms of oriented gradients for human detection”. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR ’05)*. p. 886–893.

Daliri, Mohammad Reza and Vincent Torre. 2008. “Robust symbolic representation for shape recognition and retrieval”. *Pattern Recognition*, vol. 41, n° 5, p. 1782–1798.

Daliri, Mohammad Reza and Vincent Torre. 2010. “Shape recognition based on Kernel-edit distance”. *Computer Vision and Image Understanding*, vol. 114, n° 10, p. 1097–1103.

De Oliveira,José J., Jr., João M. de Carvalho, Cinthia O. de A. Freitas, and Robert Sabourin. 2002. “Feature sets evaluation for handwritten word recognition”. In *Proceedings of the 8th International Workshop on Frontiers in Handwriting Recognition (IWFHR ’02)*. p. 446–450.

Dimitrov, Pavel, Carlos Phillips, and Kaleem Siddiqi. 2000. “Robust and efficient skeletal graphs”. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR ’00)*. p. 417–423.

Doetsch, Patrick, Mahdi Hamdani, Hermann Ney, Adrià Giménez, Jesús Andrés-Ferrer, and Alfons Juan. 2012. “Comparison of Bernoulli and Gaussian HMMs Using a Vertical Repositioning Technique for Off-Line Handwriting Recognition”. In *Proceedings of the 3rd International Conference on Frontiers in Handwriting Recognition (ICFHR ’12)*. p. 3–7.

Dreuw, Philippe, Patrick Doetsch, Christian Plahl, and Hermann Ney. 2011. “Hierarchical hybrid MLP/HMM or rather MLP features for a discriminatively trained Gaussian HMM: A comparison for offline handwriting recognition”. In *Proceedings of the 18th IEEE International Conference on Image Processing (ICIP ’11)*. p. 3541–3544.

Dreuw, Philippe, David Rybach, Georg Heigold, and Hermann Ney, July 2012. *RWTH OCR: A Large Vocabulary Optical Character Recognition System for Arabic Scripts*, chapter Part II: Recognition, p. 215–254. Springer, London, UK. ISBN 978-1-4471-4071-9.

Dryden, Ian L. and Kanti V. Mardia, 1998. *Statistical shape analysis*. Wiley series in probability and statistics. Chichester [u.a.] : Wiley.

Eraqi, Hesham M. and Sherif Abdelazeem. 2012. “HMM-based Offline Arabic Handwriting Recognition: Using New Feature Extraction and Lexicon Ranking Techniques”. In *Proceedings of the 3rd International Conference on Frontiers in Handwriting Recognition (ICFHR ’12)*. p. 554–559.

Farooq, Faisal, Anurag Bhardwaj, and Venu Govindaraju. 2009. “Using topic models for OCR correction”. *International Journal on Document Analysis and Recognition*, vol. 12, n° 3, p. 153–164.

Farrahi Moghaddam, Reza and Mohamed Cheriet. July 2009. “Application of Multi-Level Classifiers and Clustering for Automatic Word Spotting in Historical Document Images”. In *Proceedings of the 10th International Conference on Document Analysis and Recognition (ICDAR ’09)*. p. 511–515.

Farrahi Moghaddam, Reza and Mohamed Cheriet. 2010. “ A multi-scale framework for adaptive binarization of degraded document images ”. *Pattern Recognition*, vol. 43, n° 6, p. 2186–2198.

Farrahi Moghaddam, Reza, Mohamed Cheriet, Mathias M. Adankon, Kostyantyn Filonenko, and Robert Wisnovsky. 2010. “ Ibn Sina: A database for research on processing and understanding of Arabic manuscripts images ”. In *Proceedings of the 9th IAPR International Workshop on Document Analysis Systems (DAS ’10)*. (New York, NY, USA 2010), p. 11–18. ACM.

Feng, Shaolei, R. Manmatha, and Andrew McCallum. April 2006. “ Exploring the use of conditional random field models and HMMs for historical handwritten document recognition ”. In *Proceedings of the Second International Conference on Document Image Analysis for Libraries (DIAL ’06)*. p. 8–37.

Fink, Gernot A., 2008. *Markov Models for Pattern Recognition—From Theory to Applications*. Berlin Heidelberg : Springer-Verlag.

Fischer, Andreas, Kaspar Riesen, and Horst Bunke. November 2010. “ Graph Similarity Features for HMM-Based Handwriting Recognition in Historical Documents ”. In *Proceedings of the 12th International Conference on Frontiers in Handwriting Recognition (ICFHR ’10)*. p. 253–258.

Gers, Felix A., Nicol N. Schraudolph, and Jürgen Schmidhuber. March 2003. “ Learning precise timing with lstm recurrent networks ”. *The Journal of Machine Learning Research*, vol. 3, p. 115–143.

Giménez, Adrià and Alfons Juan. July 2009. “ Embedded Bernoulli Mixture HMMs for Handwritten Word Recognition ”. In *Proceedings of the 10th International Conference on Document Analysis and Recognition (ICDAR ’09)*. p. 896–900.

Gorelick, Lena, Meirav Galun, Eitan Sharon, Ronen Basri, and Achi Brandt. December 2006. “ Shape Representation and Classification Using the Poisson Equation ”. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28, n° 12, p. 1991–2005.

Graves, Alex. “ RNNLIB: A recurrent neural network library for sequence learning problems ”. http://sourceforge.net/projects/rnnl/.

Graves, Alex and Jürgen Schmidhuber. 2009. “ Offline handwriting recognition with multidimensional recurrent neural networks ”. In *Advances in Neural Information Processing Systems 21*. p. 545–552.

Graves, Alex, Marcus Liwicki, Santiago Fernández, Roman Bertolami, Horst Bunke, and Jürgen Schmidhuber. May 2009. “ A Novel Connectionist System for Unconstrained Handwriting Recognition ”. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 31, n° 5, p. 855–868.

Grosicki, Emmanuèle and Haikal El Abed. 2009. “ICDAR 2009 Handwriting Recognition Competition”. In *Proceedings of the 10th International Conference on Document Analysis and Recognition (ICDAR ’09).* p. 1398–1402.

Grosicki, Emmanuèle, Matthieu Carré, Jean-Marie Brodin, and Edouard Geoffrois. 2009. “Results of the RIMES Evaluation Campaign for Handwritten Mail Processing”. In *Proceedings of the 10th International Conference on Document Analysis and Recognition (ICDAR ’09).* (Washington, DC, USA 2009), p. 941–945. IEEE Computer Society.

He, Xiao Chen and N.H.C. Yung. May 2008. “Corner detector based on global and local curvature properties”. *Optical Engineering*, vol. 47, n° 5, p. 057008-1-12.

Hedjam, Rachid, Reza Farrahi Moghaddam, and Mohamed Cheriet. 2011. “A spatially adaptive statistical method for the binarization of historical manuscripts and degraded document images”. *Pattern Recognition*, vol. 44, n° 9, p. 2184–2196.

Hermansky, Hynek, Daniel P.W. Ellis, and Sangita Sharma. 2000. “Tandem connectionist feature extraction for conventional HMM systems”. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP ’00).* p. 1635–1638 vol.3.

Joshi, Shantanu H., Eric Klassen, Anuj Srivastava, and Ian Jermyn. 2007. “A Novel Representation for Riemannian Analysis of Elastic Curves in Rn”. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR ’07).* p. 1–7.

Kaufmann, Guido and Horst Bunke. 2000. “Automated Reading of Cheque Amounts”. *Pattern Analysis & Applications*, vol. 3, p. 132–141.

Kaufmann, Guido, Horst Bunke, and M. Hadorn. August 1997. “Lexicon reduction in an framework based on quantized feature vectors”. In *Proceedings of the 4th International Conference on Document Analysis and Recognition (ICDAR ’97).* p. 1097–1101.

Kim, Kye Kyung, Jin Ho Kim, Yun Koo Chung, and Ching Y. Suen. 2001. “Legal amount recognition based on the segmentation hypotheses for bank check processing”. In *Proceedings of the 6th International Conference on Document Analysis and Recognition (ICDAR ’01).* p. 964–967.

Koerich, Alessandro L., Robert Sabourin, and Ching Y. Suen. 2003. “Large vocabulary off-line handwriting recognition: A survey”. *Pattern Analysis & Applications*, vol. 6, p. 97–121.

Koerich, Alessandro L., Robert Sabourin, and Ching Y. Suen. 2005. “Recognition and Verification of Unconstrained Handwritten Words”. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, p. 1509–1522.

Lavrenko, Victor, Toni M. Rath, and R. Manmatha. 2004. “Holistic word recognition for handwritten historical documents”. In *Proceedings of the 1st International Workshop on Document Image Analysis for Libraries (DIAL ’04).* p. 278–287.

Lazebnik, Svetlana, Cordelia Schmid, and Jean Ponce. 2006. "Beyond Bags of Features: Spatial Pyramid Matching for Recognizing Natural Scene Categories". In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR '06)*. p. 2169–2178.

LeCun, Yann, Léon Bottou, Yoshua Bengio, and Patrick Haffner. November 1998. "Gradient-based learning applied to document recognition". *Proceedings of the IEEE*, vol. 86, n° 11, p. 2278–2324.

Leydier, Yann, Asma Ouji, Frank LeBourgeois, and Hubert Emptoz. 2009. "Towards an omnilingual word retrieval system for ancient manuscripts". *Pattern Recognition*, vol. 42, n° 9, p. 2089–2105.

Li, Ning, Xudong Xie, Wentao Liu, and Kin-Man Lam. 2012. "Combination of global and local baseline-independent features for offline Arabic handwriting recognition". In *Proceedings of the 21st International Conference on Pattern Recognition (ICPR '12)*. p. 713–716.

Ling, Haibin and D.W. Jacobs. February 2007. "Shape Classification Using the Inner-Distance". *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 29, n° 2, p. 286–299.

Liu, Cheng-Lin, M. Koga, and H. Fujisawa. November 2002. "Lexicon-driven segmentation and recognition of handwritten character strings for Japanese address reading". *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, n° 11, p. 1425–1437.

Lladós, Josep, Marçal Rusi nol, Alicia Fornés, David Fernández, and Anjan Dutta. 2012. "On the Influence of Word Representations for Handwritten Word Spotting in Historical Documents". *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 26, n° 5.

Lorigo, Liana M. and Venu Govindaraju. 2006. "Offline Arabic handwriting recognition: a survey". *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28, n° 5, p. 712–724.

Lowe, David G. 2004. "Distinctive Image Features from Scale-Invariant Keypoints". *International Journal of Computer Vision*, vol. 60, p. 91–110.

Madhvanath, Sriganesh, V. Krpasundar, and Venu Govindaraju. 2001. "Syntactic methodology of pruning large lexicons in cursive script recognition". *Pattern Recognition*, vol. 34, n° 1, p. 37–46.

Märgner, Volker and Haikal El Abed. July 2009. "ICDAR 2009 Arabic Handwriting Recognition Competition". In *Proceedings of the 10th International Conference on Document Analysis and Recognition (ICDAR '09)*. p. 1383–1387.

Märgner, Volker and Haikal El Abed. November 2010. "ICFHR 2010 - Arabic Handwriting Recognition Competition". In *Proceedings of the 12th International Conference on Frontiers in Handwriting Recognition (ICFHR '10)*. p. 709–714.

Märgner, Volker and Haikal El Abed. September 2011. "ICDAR 2011 - Arabic Handwriting Recognition Competition". In *Proceedings of the 11th International Conference on Document Analysis and Recognition (ICDAR '11)*. p. 1444–1448.

Marti, Urs-Viktor and Horst Bunke. 2001. "Using a statistical language model to improve the performance of an HMM-based cursive handwriting recognition system". *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 15, n° 01, p. 65–90.

Marzal, Andrés and Enrique Vidal. September 1993. "Computation of normalized edit distance and applications". *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 15, n° 9, p. 926–932.

Menasri, Farès, Jérôme Louradour, Anne-Laure Bianne-Bernard, and Christopher Kermorvant. 2012. "The A2iA French handwriting recognition system at the Rimes-ICDAR2011 competition". <http://dx.doi.org/10.1117/12.911981>.

Milewski, Robert and Venu Govindaraju. June 2004. "Handwriting analysis of pre-hospital care reports". In *Proceedings of the 17th IEEE Symposium on Computer-Based Medical Systems (CBMS '04)*. p. 428–433.

Mio, Washington, Anuj Srivastava, and Shantanu Joshi. July 2007. "On Shape of Plane Elastic Curves". *International Journal of Computer Vision*, vol. 73, p. 307–324.

Morgan, Nelson and Hervé Bourlard. 1995. "Continuous speech recognition". *IEEE Signal Processing Magazine*, vol. 12, n° 3, p. 24–42.

Mori, Greg, Serge Belongie, and Jitendra Malik. November 2005. "Efficient shape matching using shape contexts". *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, n° 11, p. 1832–1837.

Morita, Marisa, Robert Sabourin, Flávio Bortolozzi, and Ching Y. Suen. 2002. "Segmentation and recognition of handwritten dates". In *Proceedings of the 8th International Workshop on Frontiers in Handwriting Recognition (IWFHR '02)*. p. 105–110.

Mozaffari, Saeed, Karim Faez, Volker Märgner, and Haikal El Abed. September 2007. "Strategies for Large Handwritten Farsi/Arabic Lexicon Reduction". In *Proceedings of the 9th International Conference on Document Analysis and Recognition (ICDAR '07)*. p. 98–102.

Mozaffari, Saeed, Karim Faez, Volker Märgner, and Haikal El Abed. 2008a. "Two-Stage Lexicon Reduction for Offline Arabic Handwritten Word Recognition". *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 22, p. 1323–1341.

Mozaffari, Saeed, Karim Faez, Volker Märgner, and Haikal El-Abed. 2008b. "Lexicon reduction using dots for off-line Farsi/Arabic handwritten word recognition". *Pattern Recognition Letters*, vol. 29, n° 6, p. 724–734.

Neumaier, Arnold. 1982. "The second largest eigenvalue of a tree". *Linear Algebra and its Applications*, vol. 46, n° 0, p. 9–25.

Niu, Xiao-Xiao and Ching Y. Suen. 2012. "A novel hybrid CNN–SVM classifier for recognizing handwritten digits". *Pattern Recognition*, vol. 45, n° 4, p. 1318–1325.

Oh, Il-Seok, Jin-Seon Lee, and Ching Y. Suen. 1999. "Analysis of class separation and combination of class-dependent features for handwriting recognition". *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 21, n° 10, p. 1089–1094.

Otsu, Nobuyuki. 1979. "A Threshold Selection Method from Gray-Level Histograms". *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 9, p. 62–66.

Palla, Srinivas, Hansheng Lei, and Venu Govindaraju. 2004. "Signature and Lexicon Pruning Techniques". In *Proceedings of the 9th International Workshop on Frontiers in Handwriting Recognition (IWFHR '04)*. (Washington, DC, USA 2004), p. 474–478. IEEE Computer Society.

Park, Jaehwa. July 2002. "An adaptive approach to offline handwritten word recognition". *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, n° 7, p. 920–931.

Pechwitz, Mario and Volker Märgner. 2002. "Baseline estimation for Arabic handwritten words". In *Proceedings of the 8th International Workshop on Frontiers in Handwriting Recognition (IWFHR '02)*. p. 479–484.

Pechwitz, Mario, Samia Snoussi Maddouri, Volker Märgner, Noureddine Ellouze, and Hamid Amiri. 2002. "IFN/ENIT-Database of Handwritten Arabic words". In *Proceedings of the 7th Colloque International Francophone sur l'Ecrit et le Document (CIFED '02)*. (Hammamet, Tunisia 2002), p. 129–136.

Plamondon, Réjean and Sargur N. Srihari. 2000. "Online and off-line handwriting recognition: a comprehensive survey". *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, n° 1, p. 63–84.

Quelhas, Pedro, Florent Monay, Jean-Marc Odobez, Daniel Gatica-Perez, and Tinne Tuytelaars. September 2007. "A Thousand Words in a Scene". *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 29, n° 9, p. 1575–1589.

Rabiner, Lawrence R. and Biing-Hwang Juang. 1986. "An introduction to hidden Markov models". *IEEE ASSP Magazine*, vol. 3, n° 1, p. 4–16.

Rath, Toni M. and R. Manmatha. June 2003a. "Word image matching using dynamic time warping". In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR '03)*. p. II-521 - II-527 vol.2.

Rath, Toni M. and R. Manmatha. August 2003b. "Features for word spotting in historical manuscripts". In *Proceedings of the 7th International Conference on Document Analysis and Recognition (DAS '03)*. p. 218–222 vol.1.

Rodríguez-Serrano, José A. and Florent Perronnin. 2009. "Handwritten word-spotting using hidden Markov models and universal vocabularies". *Pattern Recognition*, vol. 42, n° 9, p. 2106–2116.

Rodríguez-Serrano, José A. and Florent Perronnin. November 2012. "A Model-Based Sequence Similarity with Application to Handwritten Word Spotting". *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, n° 11, p. 2108–2120.

Rothacker, Leonard, Szilárd Vajda, and Gernot A. Fink. 2012. "Bag-of-Features Representations for Offline Handwriting Recognition Applied to Arabic Script". In *Proceedings of the 3rd International Conference on Frontiers in Handwriting Recognition (ICFHR '12)*. p. 149–154.

Roweis, Sam T. and Lawrence K. Saul. 2000. "Nonlinear dimensionality reduction by locally linear embedding". *Science*, vol. 290, p. 2323–2326.

Sayre, Kenneth M. 1973. "Machine recognition of handwritten words: A project report". *Pattern Recognition*, vol. 5, n° 3, p. 213–228.

Scholkopf, Bernhard, Alexander Smola, and Klaus-Robert Müller. 1999. "Kernel principal component analysis". In *Advances in kernel methods - Support vector learning*. p. 327–352. MIT Press.

Schomaker, Lambert. 1998. "From handwriting analysis to pen-computer applications". *Electronics Communication Engineering Journal*, vol. 10, n° 3, p. 93–102.

Sebastian, Thomas B., Philip N. Klein, and Benjamin B. Kimia. January 2003. "On aligning curves". *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 25, n° 1, p. 116–125.

Sebastian, Thomas B., Philip N. Klein, and Benjamin B. Kimia. 2004. "Recognition of Shapes by Editing Their Shock Graphs". *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 26, n° 5, p. 550–571.

Shokoufandeh, Ali, Diego Macrini, Sven Dickinson, Kaleem Siddiqi, and Steven W. Zucker. 2005. "Indexing hierarchical structures using graph spectra". *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, n° 7, p. 1125–1140.

Siddiqi, Kaleem, Ali Shokoufandeh, Sven J. Dickinson, and Steven W. Zucker. 1999. "Shock Graphs and Shape Matching". *International Journal of Computer Vision*, vol. 35, n° 1, p. 13–32.

Slimane, Fouad, Slim Kanoun, Haikal El Abed, Adel M. Alimi, Rolf Ingold, and Jean Hennebert. September 2011. "ICDAR 2011 - Arabic Recognition Competition: Multi-font

Multi-size Digitally Represented Text ". In *Proceedings of the 11th International Conference on Document Analysis and Recognition (ICDAR '11)*. p. 1449–1453.

Slimane, Fouad, Oussama Zayene, Slim Kanoun, Adel M. Alimi, Jean Hennebert, and Rolf Ingold. November 2012. " New Features for Complex Arabic Fonts in Cascading Recognition System ". In *Proceedings of the 21th International Conference on Pattern Recognition (ICPR '12)*. p. 738–741.

Srihari, Sargur N. 1993. " Recognition of handwritten and machine-printed text for postal address interpretation ". *Pattern Recognition Letters*, vol. 14, n° 4, p. 291–302.

Srivastava, Anuj, Eric Klassen, Shantanu H. Joshi, and Ian H. Jermyn. July 2011. " Shape Analysis of Elastic Curves in Euclidean Spaces ". *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, n° 7, p. 1415–1428.

Terasawa, Kengo and Yuzuru Tanaka. July 2009. " Slit Style HOG Feature for Document Image Word Spotting ". In *Proceedings of the 10th International Conference on Document Analysis and Recognition (ICDAR '09)*. p. 116–120.

Tomai, Catalin I., Bin Zhang, and Venu Govindaraju. 2002. " Transcript mapping for historic handwritten document images ". In *Proceedings of the 8th International Workshop on Frontiers in Handwriting Recognition (IWFHR '02)*. p. 413–418.

Vamvakas, Georgios, Basilios Gatos, Nikolaos Stamatopoulos, and Stavros J. Perantonis. September 2008. " A Complete Optical Character Recognition Methodology for Historical Documents ". In *Proceedings of the 8th IAPR International Workshop on Document Analysis Systems (DAS '08)*. p. 525–532.

van der Zant, Tijn, Lambert Schomaker, and Koen Haak. 2008. " Handwritten-Word Spotting Using Biologically Inspired Features ". *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 30, n° 11, p. 1945–1957.

van Erp, Merijn, Louis Vuurpijl, and Lambert Schomaker. 2002. " An overview and comparison of voting methods for pattern recognition ". In *Proceedings of the 8th International Workshop on Frontiers in Handwriting Recognition (IWFHR '02)*. p. 195–200.

Vincent, Pascal, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. 2008. " Extracting and composing robust features with denoising autoencoders ". In *Proceedings of the 25th international conference on Machine learning (ICML '08)*. (New York, NY, USA 2008), p. 1096–1103. ACM.

Vinciarelli, Alessandro. 2002. " A survey on off-line Cursive Word Recognition ". *Pattern Recognition*, vol. 35, n° 7, p. 1433–1446.

Vinciarelli, Alessandro, Samy Bengio, and Horst Bunke. 2004. " Offline recognition of unconstrained handwritten texts using HMMs and statistical language models ". *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 26, n° 6, p. 709–720.

Viola, Paul and Michael J. Jones. 2004. “Robust Real-Time Face Detection”. *International Journal of Computer Vision*, vol. 57, p. 137–154.

Wshah, Safwan, Venu Govindaraju, Yanfen Cheng, and Huiping Li. August 2010. “A Novel Lexicon Reduction Method for Arabic Handwriting Recognition”. In *Proceedings of the 20th International Conference on Pattern Recognition (ICPR ’10)*. p. 2865–2868.

Wu, Lei and Steven C.H. Hoi. January 2011. “Enhancing Bag-of-Words Models with Semantics-Preserving Metric Learning”. *IEEE MultiMedia*, vol. 18, n° 1, p. 24–37.

Wüthrich, Markus, Marcus Liwicki, Andreas Fischer, Emanuel Indermühle, Horst Bunke, Gabriel Viehhauser, and Michael Stolz. July 2009. “Language Model Integration for the Recognition of Handwritten Medieval Documents”. In *Proceedings of the 10th International Conference on Document Analysis and Recognition (ICDAR ’09)*. p. 211–215.

Yang, Jun, Yu-Gang Jiang, Alexander G. Hauptmann, and Chong-Wah Ngo. 2007. “Evaluating bag-of-visual-words representations in scene classification”. In *Proceedings of the 9th international Workshop on multimedia information retrieval (MIR ’07)*. (New York, NY, USA 2007), p. 197–206. ACM.

Young, Steve J., Gunnar Evermann, Mark J. F. Gales, Thomas Hain, Dan Kershaw, Xunying Liu, Gareth Moore, Julian Odell, Dave Ollason, Dan Povey, Valtcho Valtchev, and Phil C. Woodland, 2006. *The HTK Book, version 3.4*. Cambridge, UK : Cambridge University Engineering Department.

Zhou, Li, Zongtan Zhou, and Dewen Hu. 2013. “Scene classification using a multi-resolution bag-of-features model”. *Pattern Recognition*, vol. 46, n° 1, p. 424–433.

Zidouri, Abdelmalek. October 2004. “ORAN: a basis for an Arabic OCR system”. In *Proceedings of 2004 International Symposium on Intelligent Multimedia, Video and Speech Processing (ISIMP ’04)*. p. 703–706.

Zimmermann, Matthias and Jianchang Mao. 1999. “Lexicon reduction using key characters in cursive handwritten words”. *Pattern Recognition Letters*, vol. 20, n° 11-13, p. 1297–1304.