Hindawi Publishing Corporation
EURASIP Journal on Advances in Signal Processing
Volume 2008, Article ID 247354, 13 pages
doi:10.1155/2008/247354

Research Article

Arabic Handwritten Word Recognition Using HMMs with Explicit State Duration

A. Benouareth,[1] A. Ennaji,[2] and M. Sellami[1]

[1] *Laboratoire de Recherche en Informatique, Département d'Informatique, Université Badji Mokhtar, Annaba, BP 12- 23000 Sidi Amar, Algeria*
[2] *Laboratoire LITIS (FRE 2645), Université de Rouen, 76800 Madrillet, France*

Correspondence should be addressed to A. Benouareth, benouareth@lri-annaba.net

Received 09 March 2007; Revised 20 June 2007; Accepted 28 October 2007

Recommended by C.-C. Kuo

We describe an offline unconstrained Arabic handwritten word recognition system based on segmentation-free approach and discrete hidden Markov models (HMMs) with explicit state duration. Character durations play a significant part in the recognition of cursive handwriting. The duration information is still mostly disregarded in HMM-based automatic cursive handwriting recognizers due to the fact that HMMs are deficient in modeling character durations properly. We will show experimentally that explicit state duration modeling in the HMM framework can significantly improve the discriminating capacity of the HMMs to deal with very difficult pattern recognition tasks such as unconstrained Arabic handwriting recognition. In order to carry out the letter and word model training and recognition more efficiently, we propose a new version of the Viterbi algorithm taking into account explicit state duration modeling. Three distributions (Gamma, Gauss, and Poisson) for the explicit state duration modeling have been used, and a comparison between them has been reported. To perform word recognition, the described system uses an original sliding window approach based on vertical projection histogram analysis of the word and extracts a new pertinent set of statistical and structural features from the word image. Several experiments have been performed using the IFN/ENIT benchmark database and the best recognition performances achieved by our system outperform those reported recently on the same database.

1. INTRODUCTION

The term handwriting recognition (HWR) refers to the process of transforming a language, which is presented in its spatial form of graphical marks, into its symbolic representation. The problem of handwriting recognition can be classified into two main groups, namely *offline* and *online* recognition, according to the format of handwriting inputs. In offline recognition, only the image of the handwriting is available, while in the online case temporal information such as pentip coordinates as a function of time is also available. Typical data acquisition devices for offline and online recognition are scanners and digitizing tablets, respectively. Due to the lack of temporal information, offline handwriting recognition is considered more difficult than online. Furthermore, it is also clear that the offline case is the one that corresponds to the conventional reading task performed by humans.

Many applications require offline HWR capabilities such as bank processing, mail sorting, document archiving, commercial form-reading, and office automation. So far, offline HWR remains a very challenging task in spite of dramatic boost of research [1–3] in this field and the latest improvement in recognition methodologies [4–7].

Studies on Arabic handwriting recognition, although not as advanced as those devoted to other scripts (e.g., Latin), have recently shown a renewed interest [8–10]. We point out that the techniques developed for Latin HWR are not appropriate for Arabic handwriting because Arabic script is based on alphabet and rules different from those of Latin. Arabic writing, both handwritten and printed, is semicursive (i.e., the word is a sequence of disjoint connected components called pseudowords and each pseudoword is a sequence of completely cursive characters and is written from right to left). The character shape is context sensitive, that is, depending on its position within a word. For instance, a letter

as غ has 4 different shapes: isolated "غ" as in "بلوغ," beginning as "غروب", middle as "نغم", and end as "تبغ". Arabic writing is very rich in diacritic marks (e.g., dots, Hamza, etc.) because some Arabic characters may have exactly the same main shape, and are distinguished from each other only by the presence or the absence of these diacritics and their number and their position with respect to the main shape. The main characteristics of Arabic writing are summarized by Figure 1 [11].

One can classify the field of offline handwriting cursive word recognition into four categories according to the size and nature of the lexicon involved: very large; large; limited but dynamic; and small and specific. Small lexicons do not include more than 100 words, while limited lexicons may go up to 1000. Large lexicons may contain thousands of words, and very large lexicons refer to any lexicon beyond that. When a dynamic lexicon (in contrast with specific or constant) is used, it means that the words that will be relevant during a recognition task are not available during training because they belong to an unknown subset of a much larger lexicon.

The lexicon is a key point to the success of any HWR system, because it is a source of linguistic knowledge that helps to disambiguate single characters by looking at the entire context. As the number of words in the lexicon grows, the more difficult the recognition task becomes, because more similar words are more likely to be present in the lexicon. The computational complexity is also related to the lexicon, and it increases according to its size [1].

The word is the most natural unit of handwriting, and its recognition process can be done either by an *analytic* approach of recognizing individual characters in the word or *holistic* approach of dealing with the entire word image as a whole.

Analytical approaches (e.g., [13]) basically have two steps, segmentation and combination. First the input image is segmented into units no bigger than characters, then segments are combined to match character models using dynamic programming. Based on the granularity of segmentation and combination, analytical approaches can be further divided into three subcategories: (i) character-based approaches [14] that recognize each character in the word and combine the character recognition results using either explicit or implicit segmentation and requiring high-performance character recognizer; (ii) grapheme-based approaches [4, 13] that use graphemes (i.e., structural parts in characters, e.g., the loop part in "ق", arcs, etc.) instead of characters as the minimal unit being matched; and (iii) pixel-based approaches [15–18] that use features extracted from pixel columns in sliding window to form words models for word recognition.

Holistic approaches [19] deal with the entire input image. Holistic features, like translation/rotation invariant quantities, word length, connected components, ascenders, descenders, dots, and so forth, are usually used to eliminate less likely choices in the lexicon. Since holistic models must be trained for every word in the lexicon, compared against analytical models that need only be trained for every character, their application is limited to those with small and constant lexicons, such as reading the courtesy amount on bank checks [20, 21].

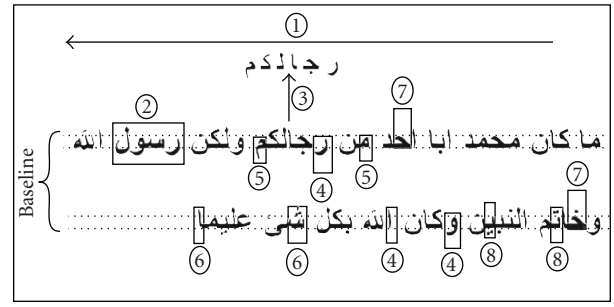

Figure 1: An Arabic sentence demonstrating the main characteristics of Arabic text [12]. (1) Written from right to left. (2) One Arabic word includes three cursive subwords. (3) A word consisting of six characters. (4) Some characters are not connectable from the left side with the succeeding character. (5) The same character with different shapes depends on its position in the word. (6) Different characters with different sizes. (7) Different characters with a different number of dots. (8) Different characters have the same number of dots but different positions of dots.

The analytical approach is theoretically more efficient in handling a large vocabulary. Indeed with a constant number of classification classes (e.g., the number of letters in the alphabet), it can handle any string of characters and therefore an unlimited number of words. However, the Sayere's paradox (a word cannot be segmented before being recognised and cannot be recognized before being segmented [22]) was shown to be a significant limit of any analytical approach. The holistic approach on the other hand must generally rely on an established vocabulary of acceptable words. Its number of classification classes increases with the size of the lexicon. The "whole word" scheme is potentially faster when considering a relatively small lexicon. It is also more accurate having to consider only the legitimate word possibilities. One disadvantage of a whole word recognizer is its inability to identify a word not contained in the vocabulary. On the other hand, it has greater tolerance in the presence of noise, spelling mistakes, missing characters, unreadable part of the word, and so forth.

Stochastic models, especially hidden Markov models (HMMs) [23], have been successfully applied to offline HWR in recent years [4, 6, 7]. This success can be attributed to the probabilistic nature of HMM models, which can perform a robust modeling of the handwriting signal with huge variability and sometimes corrupted by noise. Moreover, HMMs can efficiently integrate the contextual information at different levels of the recognition process (morphological, lexical, syntactical, etc.).

Character durations play a significant part in the recognition of cursive handwriting. The duration information is still mostly disregarded in HMMs-based automatic cursive handwriting recognizers due to the fact that HMMs are deficient in modeling character durations properly. We will show experimentally that explicit state duration modeling

in the HMM framework can significantly improve the discriminating capacity of the HMMs to deal with very difficult pattern recognition tasks such as unconstrained Arabic handwriting recognition on a large lexicon. In order to carry out the letter and word model training and recognition more efficiently, we propose a new version of the Viterbi algorithm taking into account explicit state duration modeling.

This paper describes an extended version of an offline unconstrained Arabic handwritten word recognition system based on segmentation-free approach and discrete HMMs with explicit state duration [24]. Three distributions (Gamma, Gauss, and Poisson) for the explicit state duration modeling have been used and a comparison between them has been reported. To the best of our knowledge, this is the first work that uses explicit state duration of discrete and continuous distribution for the offline Arabic handwriting recognition problem. After preprocessing intended to simplify the later stages of the recognition process, the word image is first divided according to two different schemes (uniform and nonuniform) from right to left into frames using a sliding window. We have introduced the nonuniform segmentation in order to tackle the morphological complexity of Arabic handwriting characters. Then each frame is analyzed and characterized by a vector having 42 components and combining a new set of relevant statistical and structural features. The output of this stage is a sequence of feature vectors which will be transformed by vector quantization into a sequence of discrete observations. This latter sequence is submitted to an HMM classifier to carry out word discrimination by a modified version of the Viterbi algorithm [15, 25]. The HMMs relating to the word recognition lexicon are built during a training stage, according to two different methods. In the first method, each word model is created separately from its training samples. The second method associates a distinct HMM for each basic shape of Arabic character, and thus, each word model is generated by linking its character models. This efficiently allows character model sharing between word models using a tree-structured lexicon.

Significant experiments have been performed on the IFN/ENIT benchmark database [26]. They have shown on the one hand a substantial improvement in the recognition rate when HMMs with explicit state duration of either discrete or continuous distribution is used instead of classical HMMs (i.e., with implicit state duration, cf. Section 3.2). On the other hand, the system has achieved best performances with the Gamma distribution for state duration. Our best recognition performances outperform those recently reported on the same database. The HMM parameter selection is discussed and the resulting performances are presented with respect to the state duration distribution type, as well as to the word segmentation scheme into frames and the word model training method.

The rest of this paper is organized as follows. Section 2 sketches some related studies in HWR using HMMs. Section 3 briefly introduces the classical HMMs and details HMMs with different explicit state duration types and their parameter estimation. A modified version of the Viterbi algorithm used in the training and recognition of letter and word models is also presented in this section. Section 4 summarizes the developed system architecture in a block diagram. Section 5 explains the preprocessing applied to the word image. Section 6 describes the features extraction stage. Section 7 is devoted to the training and the classification process. Section 8 illustrates and outlines the results achieved by the experiments performed on the IFN/ENIT benchmark database, and makes a comparison between our best recognition performances and those recently reported on the same database. Finally, a conclusion is drawn with some outlooks in Section 9.

2. RELATED WORKS

Since the end of 1980s, the very successful use of HMMs in speech recognition has led many researchers to apply them to various problems in the field of handwriting recognition such as character recognition [27], offline word recognition [28], and signature verification and identification [12]. These HMM frameworks can be distinguished from each other by the state meaning, the modeled units (stroke, character, word, etc.), the unit model topology (ergodic or left-to-right), the HMM type (discrete or continuous), the HMM dimensionality (one-dimensional, planar, bidimensional, or random fields), the state duration modeling type (implicit or explicit), and the modeling level (morphological, lexical, syntactical, etc.).

Gillies [29] has used an implicit segmentation-based HMM for cursive word recognition. First, a label is given to each pixel in the image according to its membership in strokes, holes, and concavities. Then, the image is transformed into a sequence of symbols by vector quantization of each pixel column. Each letter is modeled by a different discrete HMM whose parameters are estimated from hand-segmented data. The Viterbi algorithm [25] is used for recognition and it allows an implicit segmentation of the word into letters by a by-product of the word matching.

Mohamed and Gader [30] used continuous HMMs to segmentation-free modeling of handwritten words in which the observations are based on the location of black-white and white-black transitions on each image column. They designed a 12-state left-to-right HMM for each character.

Chen et al. [28] used HMMs with explicit state duration named continuous density variable duration HMM. After explicit segmentation of the word into subcharacters, the observations used are based on geometrical and topological features (pixel distribution, etc.). Each letter is identified with a state which can account for up to four segments per letter. The parameters of the HMM are estimated using the lexicon and the manually labeled data. A modified Viterbi algorithm is applied to provide the N best paths, which are postprocessed using a general string edit distance method.

Vinciarelli and Bengio [31] employed continuous density HMM to recognize offline cursive words written by a single writer. Their system is based on a sliding window approach to avoid the need of independent explicit segmentation stage. As the sliding window blindly isolates the pattern frames from which the feature vectors are extracted, the

used features are computed by partitioning each frame into cells regularly arranged in 4×4 grids and by locally averaging the pixel distribution in each cell. The HMM parameter number is reduced by using diagonal covariance matrices in the emission probabilities. These matrices are derived from the decorrelated feature vectors that result from applying principal component analysis (PCA) and independent component analysis (ICA) to the basic features. A different HMM is created for each letter in which the number of states and the number of Gaussian in the mixtures are selected through the cross-validation method. The word models are established as concatenations of letter models.

Bengio et al. [32] have proposed an online word recognition system using convolutional neural networks and HMMs. After word normalization by fitting a geometrical model to the word structure using the expectation maximization (EM) algorithm, an annotated image representation (i.e., a low-resolution image in which each pixel contains information about the local properties of the handwritten strokes) is derived from the pen trajectory. Then, character spotting and recognition is done by convolutional neural network, and its outputs are interpreted by HMM that takes into account word-level constraints to produce word scores. A three-state HMM for each character with a left and right state to model transitions and a center state for the character itself are used to form an observation graph by connecting these character models, allowing any character to follow any other character. The word level constraints are the constraints that are independent of observations (i.e., grammar graph) and can embody lexical constraints. The recognition finds the best path in the observation graph that is compatible with the grammar graph.

El-Yacoubi et al. [4] have designed an explicit segmentation-based HMM approach to recognize offline unconstrained handwritten words for a large but dynamically limited vocabulary. Three sets of features have been used: the first two are related to the shape of the segmented units (letters or subletters) while the features of the third set describe the segmentation points between these units. The first set is based on global features, such as loops, ascenders, and descenders; and the second set is based on features obtained by the analysis of the bidimensional contour transition histogram of each segment. Finally, segmentation features correspond to either spaces, possibly occurring between letters or words, or to the vertical position of segmentation points that split connected letters. The two shape-feature sets are separately extracted from the segmented image; this allows representing each word by two feature sequences of equal length, each consisting of an alternating sequence of segment shape symbols and associated segmentation points symbols. Since the basic unit in the model is the letter, then the word (or word sequence) model is dynamically made up of the concatenation of appropriate letter models consisting of elementary HMMs, and an interpolation technique is used to optimally combine the shape symbols and the segmentation symbols. Character model is related to the behavior of the segmentation process. This process can produce either a correct segmentation of a letter, a letter omission, or an oversegmentation of a letter into two or three segments. As a result, an eight-state HMM having three paths, in order to take into account these configurations, is built for each letter. Observations are then emitted along transitions. Besides, a special model is designed for interword space, in the case in which the input image contains more than one word. It consists of two states linked by two transitions, modeling a space or no space between a pair of words.

Koerich et al. [13] have improved the system of El-Yacoubi et al. [4] to deal with a large vocabulary of 30,000 words. The recognition is carried out with a tree-structured lexicon, and the characters are modeled by multiple HMMs that are concatenated to build the word models. The tree structure of lexicon allows, during the recognition stage, words to share the same computation steps. To avoid an explosion of the search space due to presence of multiple character models, a lexicon-driven level building algorithm (LDLBA) has been developed to decode the lexicon tree and to choose the more likely models at each level. Bigram probabilities related to the variation of writing styles within the word are inserted between the levels of the LDLBA to improve the recognition accuracy. To further speed up the recognition process, some constraints on the number of levels and on the number of observations aligned at each level are added to limit the search scope to more likely parts of the search space.

Amara and Belaid [33] used planar HMMs [34] with a holistic approach for offline-printed Arabic pseudowords recognition. The adopted pseudoword model topology, in which the main model (i.e., HMM with superstates) is vertical, allows modeling of the different variations of the Arabic writing such as elongation of the horizontal ligatures and the presence of vertical ligatures. Firstly, the pseudoword image is vertically segmented into strips according to the considered pattern. These strips reflect the morphological features of different characters forming the pseudoword such as ascenders, the upper diacritic dots, holes and/or vertical ligature position, the lower diacritic dots and/or vertical ligature position, and descenders. Then, each strip is modeled by a left-to-right horizontal secondary model (1D HMM) whose parameters are tightly related to the strip topology. In the horizontal model, the observations are computed on the different segments (runs) of the pseudoword image, and they consist of the segment color (black or white) together with its length and its position with respect to the segment situated above it. In the vertical model, the duration (assimilated to the lines number in each strip) in each superstate is explicitly modeled by a specific function, in order to take into account the height of each strip.

Khorsheed [35] has presented a method for offline-handwritten script recognition, using a single HMM with structural features extracted from the manuscript words. The single HMM is composed of multiple character models where each model is left-to-right HMM, and represents one letter from the Arabic alphabet. After preprocessing, the skeleton graph of the word is decomposed into a sequence of links in the order in which the word is written. Then, each link is further broken into several line segments using a line approximation technique. The line segment sequence

is transformed into discrete symbols by vector quantization. The symbol sequence is presented to the single HMM which outputs an order list of letter sequence associated with the input pattern by applying a modified version of the Viterbi algorithm.

Pechwitz and Maergner [17] have described an HMM-based approach for offline-handwritten Arabic word recognition using the IFN/ENIT benchmark database [26]. Preprocessing is applied to normalize the height, length, and the baseline of the word, and followed by a feature extraction stage based on a sliding window approach. The features used are collected directly from the gray values of the normalized word image, and reduced by a Loeve-Karhunen transformation. Due to the fact that Arabic characters might have several shapes depending on their position in a word, a semicontinuous HMM (SCHMM) is generated for each character shape. This SCHMM has 7 states, in which each state has 3 transitions: a self-transition, a transition to the next state, and one allowing skipping a single state. The training process is performed by a *k-mean* algorithm where a model parameter initialization is done by a dynamic programming clustering approach. The recognition is carried out by applying a frame synchronous network Viterbi search algorithm together with a tree-structured lexicon representing the valid words.

From this quick survey, we can conclude that HMMs dominate the field of cursive handwriting recognition, but there are few works in this field in which HMMs with explicit state duration have been employed.

3. HIDDEN MARKOV MODELS (HMMS) AND STATE DURATION MODELING

Before introducing the notion of explicit state modeling in HMMs, we will shortly recall the definition of one-dimensional discrete HMMs.

3.1. Hidden Markov models (HMMs)

A hidden Markov model (HMM) [23] is a type of stochastic model appropriate for nonstationary stochastic sequences with statistical properties that undergo distinct random transitions among a set of different stationary processes. In other words, the HMM allows to model a sequence of observations as a piecewise stationary process. More formally, an HMM is defined by N: the number of states, M: the number of possible observation symbols, T: the length of the observation sequence, $Q = \{q_t\}$: the set of possible states, $q_t \in \{1, 2, \ldots, N\}$, $1 \le t \le T$, $V = \{v_k\}$: the codebook or the discrete set of possible observation symbols, $1 \le k \le M$. $A = \{a_{ij}\}$: the state transition probability: $a_{ij} = P(q_{t+1} = j \mid q_t = i)$, $1 \le i, j \le N$, $B = \{b_j(v_k)\}$: the observation symbol probability distribution:

$$b_j(v_k) = P(v_k \text{ at } t \mid q_t = j), \quad 1 \le i \le N, \ 1 \le k \le M, \tag{1}$$

$\pi = \{\pi_i\}$: the initial state probability, $\pi_i = P(q_1 = i)$, $1 \le i \le N$. More compactly, an HMM can be represented by the parameter $\lambda(\pi, A, B)$.

To suitably use HMMs in handwriting recognition, three problems must be solved. The first problem is concerned with the probability evaluation of an observation sequence given the model λ (i.e., the observation matching). The second problem is that we attempt to determine the state sequence (i.e., state decoding) that "best" explains the input sequence of observations. The third problem consists of determining a method to optimize the model parameters (i.e., the parameter re-estimation) to satisfy a certain optimization criterion.

The evaluation probability problem can be efficiently solved by the forward-backward procedure [23]. A solution to the state decoding problem, based on dynamic programming, has been designed, namely, the Viterbi algorithm [25]. The model parameter determination is usually done by the Baum-Welch procedure based on the expectation maximization (EM) algorithm [23], and consists in iteratively maximizing the observation likelihood given the model, and often converges to a local maximum.

3.2. Duration modeling in the HMM framework

We clearly distinguish between two discrete HMM types: HMM with implicit state duration (i.e., classical HMM) and HMM with explicit state duration. Classical HMMs do not allow explicit duration modeling (i.e., duration that the model can spend in some state). Indeed, the probability distribution of staying for a duration d in the state i (i.e., probability of consecutively observing d symbols in state i), noted $P_i(d)$, is always considered as a geometric one with parameter a_{ii}:

$$P(d/q_i) = a_{ii}^{d-1}(1 - a_{ii}). \tag{2}$$

The form of this distribution is exponentially decreasing (i.e., it gets to its maximal value at the minimal duration $d = 1$, and decays exponentially as d increases). Described with one parameter, the distribution can effectively depict only the mean duration. Beyond that, it is unable to model any variation in the duration distributions, and hence, its use is not appropriate when the states have some explicit significance. For example, in handwriting they represent the letters or letter fragments, because, in this case, narrow letters (e.g., "ب") are modeled as being more probable than wide letters (e.g., "س"). As a result, it is suitable to explicitly model the duration spent in each state.

An HMM λ with explicit state duration probability distribution is mainly defined by the following parameters: A, B, N, $p(d)$, and π that are, respectively, state transition probability matrix, output probability matrix, a total number of HMM states, a state duration probability vector, and initial state probability vector.

In HMM with explicit state duration, the sequence of observations is generated along the following steps.

(1) Generate q_1 from the initial state distribution π.

(2) Set $t = 1$.

(3) Calculate the duration of the state q_t, d, by sampling from $P_{q_t}(d)$ (i.e., a duration d is chosen according to the state duration density $P_{q_t}(d)$).

(4) Generate d observations according to the joint observation density, $b_{q_t}(O_t, O_{t+1}, \ldots, O_{t+d})$.

(5) Set $t = t + d$.

(6) If $t \leq T$, draw the next state q_t from the transition probabilities $a_{q_{t-1}q_t}$, where $q_{t-1} \neq q_t$, and go to step (3); otherwise, terminate the procedure.

The probability $P(O/\lambda)$ of an HMM λ with explicit state duration, for a discrete observation sequence O, can be computed by a generalized forward-backward algorithm [34], as follows:

$$P(O/\lambda) = \sum_{i=1}^{N} \sum_{j=1, i\neq j}^{N} \sum_{d=1}^{t} \alpha_{t-d}(i) a_{ij} p_j(d) \prod_{s=t-d+1}^{t} b_j(o_s) \beta_t(j), \tag{3}$$

where α_t and β_t are, respectively, the partial forward and backward likelihoods that are recursively computed as

$$\begin{aligned}
&\alpha_0(j) = \pi_j, \quad 1 \leq j \leq N,\\
&\alpha_t(j) = \sum_{d=1}^{t} \sum_{\substack{i=1\\ i\neq j}}^{N} \alpha_{t-d}(i) a_{ij} p_j(d) \prod_{s=t-d+1}^{t} b_j(o_s),\\
&\qquad 1 \leq j \leq N, \quad 1 \leq t \leq T,\\
&\beta_T(i) = 1, \quad 1 \leq i \leq N,\\
&\beta_t(i) = \sum_{d=1}^{T-t} \sum_{\substack{j=1\\ j\neq i}}^{N} a_{ij} p_j(d) \prod_{s=t+1}^{t+d} b_j(o_s) \beta_{t+d}(j),\\
&\qquad 1 \leq i \leq N, \quad 1 \leq t \leq T.
\end{aligned} \tag{4}$$

To be useful, the HMMs with explicit state duration require an efficient parameter reestimation algorithm for the state duration probability (i.e., $p(d)$).

In the developed system, we have used one analytical discrete distribution (i.e., Poisson [36]) and two other continuous distributions (i.e., Normal and Gamma [37]) for the state duration probability. This choice is justified by the availability of the estimation formulas, which are derived with respect to the likelihood criterion for the parameter set of these distributions. Moreover, the number of parameters to be estimated for these distributions is tiny. According to the performed experiments on the IFN/ENIT benchmark database, the Gamma distribution seems to be the best approximation of the real distribution that remains very hard to be determined.

3.2.1. Discrete distribution

For the speech recognition purpose, Russell and Moore [36] have used a Poisson distribution for the state duration probability in the HMM. This distribution is defined as follows:

$$p_j(d) = \exp(-\mu_j) \cdot \frac{(\mu_j)^d}{d!}. \tag{5}$$

The random variable d, which denotes the time spent in state j and follows this distribution, has an expected value μ_j representing one parameter of the Poisson density. This parameter is reestimated by (6), and it is considered as the expected spent duration in state j divided by the expected occurrence of this state:

$$\overline{\mu}_j = \frac{\sum_{t_0=1}^{T} \sum_{t_1=t_0}^{T} \chi_{t_0,t_1}(j) \cdot (t_1 - t_0 + 1)}{\sum_{t_0=1}^{T} \sum_{t_1=t_0}^{T} \chi_{t_0,t_1}(j)}, \tag{6}$$

where

$$\chi_{t_0,t_1}(j) = \frac{\sum_{i=1, i\neq j}^{N} \alpha_{t_0-1}(i) a_{ij} \prod_{s=t_0}^{t_1} b_j(o_s) p_j(t_1 - t_0 + 1) \beta_{t_1}(j)}{P(O/\lambda)}. \tag{7}$$

3.2.2. Continuous distribution

Levinson [37] has proposed, in the HMM-based speech recognition framework, two continuous distributions for the state duration based on the Gamma and Gaussian probability density.

Gaussian distribution

With this distribution, the state duration probability distribution is defined as follows:

$$p_j(d) = \frac{1}{\sigma_j (2\pi)^{1/2}} \exp\left(\frac{-(d - m_j)^2}{2\sigma^2}\right), \tag{8}$$

where m_j and σ_j are the mean and variance of the Gaussian distribution.

Gamma distribution

In this case, the state duration density is defined by

$$p_j(d) = \frac{\eta_j^{\nu_j} d^{\nu_j - 1} \exp(-\eta_j d)}{\Gamma(\nu_j)}, \tag{9}$$

where the η_j and ν_j are the parameters of the Gamma distribution having a mean $\mu_j = \nu_j \eta_j^{-1}$ and a variance $\sigma_j = \nu_j \eta_j^{-2}$. Here, $\Gamma(\nu_j)$ is the Gamma function on ν_j.

The parameters of these continuous distributions are estimated by applying (6) and (10):

$$\overline{\sigma}_j = \frac{\sum_{t_0=1}^{T} \sum_{t_1=t_0}^{T} \chi_{t_0,t_1}(j) \cdot (t_1 - t_0 + 1)^2}{\sum_{t_0=1}^{T} \sum_{t_1=t_0}^{T} \chi_{t_0,t_1}(j)} - (\overline{\mu}_j)^2, \tag{10}$$

where $\overline{\mu}_j$ is defined by (6).

3.3. The modified Viterbi algorithm

We propose an extended Viterbi algorithm for sequence decoding in HMMs with explicit state duration [15]. We need to define two quantities: (1) $\delta_t(i)$ which is the probability of the best state sequence ending in state S_i at time t,

but which can be in another state at time $t+1$; (2) ψ_t which is a 2D vector used to memorize the state sequence of the optimal path and the duration of each state in this sequence, that is, $\psi_t(i,1)$: the time spent in state i; and $\psi_t(i,2)$: the state leading to state i.

The modified Viterbi algorithm is stated as follows.

(1) *Initialisation* $1 \le i \le N$

$$\begin{aligned} \delta_1(i) &= \pi_i b_i(O_1)\, p_i(1) \\ \psi_1(i) &= (0,0) \end{aligned} \tag{11}$$

(2) *Recursion* $1 \le i \le N, 2 \le t \le T$

$$\begin{aligned} \delta_t(i) &= \max_{1\le\tau\le t-1} \left\{ \max_{\substack{1\le j\le N \\ i\ne j}} \{\delta_\tau(j) a_{ji}\}\, p_i(t-\tau) \prod_{k=\tau+1}^{t} b_i(O_\tau) \right\} \\ \psi_t(i) &= \arg\max_{1\le\tau\le t-1} \left\{ \arg\max_{\substack{1\le j\le N \\ i\ne j}} \{\delta_\tau(j) a_{ji}\}\, p_i(t-\tau) \prod_{k=\tau+1}^{t} b_i(O_\tau) \right\} \end{aligned} \tag{12}$$

(3) *Termination*

$$\begin{aligned} P^* &= \max_{1\le j\le N} [\delta_T(j)] \\ q_T^* &= \arg\max_{1\le j\le N} [\delta_T(j)] \\ \tau^* &= T - \psi_T(q_T^*, 1) \\ q_{T-\tau}^* &= q_T^*, \quad 1 \le \tau < \tau^* \end{aligned} \tag{13}$$

(4) *Path backtracking* $T - \tau^* \ge t \ge 1$

$$\begin{aligned} q_t^* &= \psi_{t+\tau^*}(q_{t+\tau^*}^*, 2) \\ \tau^* &= t - \psi_t(q_t^*, 1) \\ q_{t-\tau}^* &= q_t^*, \quad 1 \le \tau < \tau^* \\ t &= t - \tau^* \end{aligned} \tag{14}$$

4. SYSTEM ARCHITECTURE

The system architecture that we have developed for Arabic handwritten word recognition is illustrated by Figure 2. The input image goes through the steps of preprocessing, feature extraction, vector quantization and classification.The classification stage uses a discrete observation sequence derived from the input image according to a sliding window approach, a tree-structured lexicon, and a database of HMMs with explicit state duration where each of them is related to a lexicon entry. These steps are detailed in the subsequent sections. The system output is a ranked list of the words producing the best likelihood on the input image.

5. PREPROCESSING

The aim of preprocessing is the removal of all elements in the word image that are not useful for recognition process. Usually, preprocessing consists of some operations such as binarization, smoothing, baseline estimation, and thinning. Due to the fact that we use the cropped binary word images coming from IFN/ENIT database [26], binarization is not needed. A smoothing process was taken to perform noise reduction by using the spatial filter proposed by Amin et al. [8]. The extraction of some features (i.e., diacritic points) requires baseline (i.e., writing line) estimation in the word image. The method described in [17], based on projection after transforming image into Hough parameter space, gives a good estimation of the baseline. Thinning is used to reduce handwriting style variability and to make straightforward extraction of some features such as cusp points, loops, and so forth. This operation is generally time-consuming, and sometimes its application to Arabic handwriting can remove diacritic points which are relevant primitives for word discrimination. Pavlidis's algorithm [38] has a lower complexity and its application preserves the diacritic points. Figure 3(b) shows the result of applying this algorithm to Figure 3(a). The skeleton can be distorted (i.e., having spurious branches and false feature points). To remedy this, we apply the technique used in [35] that is based on using the original and the thinned word image. Here, the maximum circle technique is adopted to modify the thinning result.

6. FEATURE EXTRACTION AND VECTOR QUANTIZATION

The straightforward recognition of a handwritten word from its bitmap representation is almost impossible due to the huge variability of the handwriting style and to noise affecting the data. Hence, the need to a feature extraction method that allows extracting a feature set from the word image which is relevant for classification in the most general sense of minimizing the intraclass pattern variability while maximizing the interclass pattern variability. Moreover, these features must be reliable, independent, small in number, and reduce redundancy in the word image.

The feature extraction process is tightly related to the adopted segmentation approach. Segmentation is a well-known problem in handwritten word recognition due to its high variability, especially when dealing with a large lexicon for semicursive scripts as Arabic. In order to build a feature vector sequence to describe each word, we use implicit word segmentation where the image is divided from right to left into many vertical windows or frames. We have adopted two segmentation schemes into frames. The first one is uniform where all frames have the same width as illustrated in Figure 4(a). This uniform segmentation approach is similar to those reported in [16–18], and the best frame width has been empirically fixed to 20 pixels. The second segmentation scheme that we have introduced to deal with the morphological complexity of Arabic handwritten characters is nonuniform as illustrated by Figure 4(b). In this last scheme, the frames do not necessarily have the same width and the boundaries of each frame are based on minima and maxima analysis of the vertical projection histogram (see Figure 4(c)). This analysis consists in defining the frame boundaries to be the midpoints between adjacent

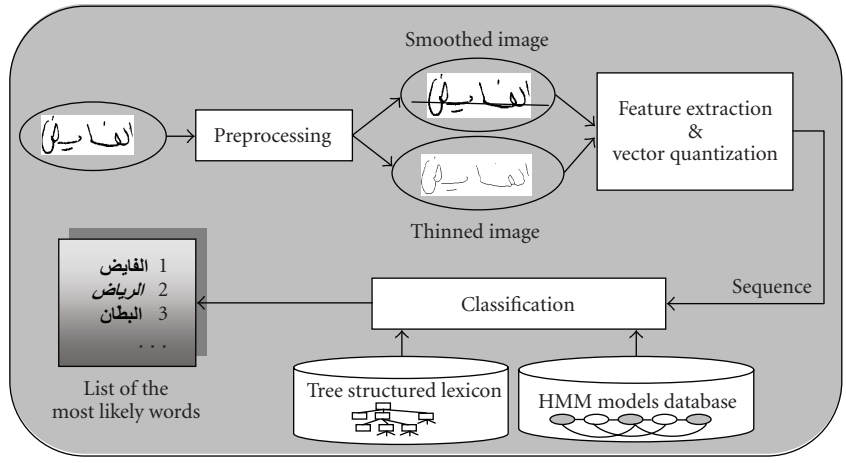

Figure 2: Recognition system architecture showing the main stages which must be carried out to identify the word image.

(a) Original image (b) Thinned image

Figure 3: Result of the thinning algorithm.

(a) (b) (c)

Figure 4: Word segmentation into frames: (a) uniform segmentation; (b) non-uniform segmentation obtained from vertical projection histogram (c).

minimum/maximum pairs. These midpoints must verify some heuristic rules related to the distance between the corresponding adjacent minimum/maximum pairs. Both these segmentation schemes have been tested and the resulted performances are reported in the validation section (cf. Section 8).

After word segmentation into frames, each frame is described by a parameter vector that is a combination of 42 relevant statistical and structural features. 33 statistical features have been computed from the histograms of the projection and transition related to 4 directions: vertical, horizontal, diagonal 45°, and diagonal 135°. The 9 structural features are computed from the thinned image. These features are detailed below. Word description is then performed from right to left as a sequence of feature vectors gathered from each frame.

6.1. Statistical features

These features consist of the mean μ, variance σ^2, and the mode (i.e., the most frequently occurring value) for the projection histogram: the minimum and maximum value for the white-to-black transition histogram. Therefore, 12 features are extracted from the projection histograms and 20 features from the transition histograms, in addition to the frame aspect ratio (i.e., width/height ratio in a frame).

6.2. Structural features

The word skeleton representation allows getting some features which are hard to extract from the bitmap representation. Works on handwriting recognition have shown that the recognition system performance may be markedly improved by using statistical and structural feature combination. The features which are computed from the thinned image correspond to the following.

(i) *Feature points*: represent the black pixels in the word skeleton having a neighbor number different from 0 and 2 (see Figures 5(a)–5(c)). There are two types: end points and junction points. End points correspond to a segment beginning/ending. The junction points connect three or more branches in the word skeleton, and are split into cross and branch points.

(ii) *Inflection points*: correspond to a curvature sign change in the word skeleton (see Figure 5(d)).

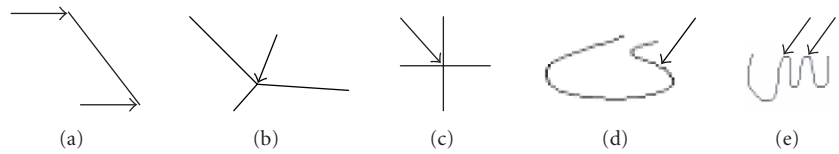

Figure 5: Some structural features: (a) end points, (b) branch point, (c) cross point, (d) inflection point, (e) cusp points.

(iii) *Cusp points*: correspond to sharp changes in direction and occur when two segments from a sharp angle in th word skeleton (see Figure 5(e)). These points are computed by Algorithm 1 [39].

The *smoothed global curvature* is defined as

$$\delta_{is} = \theta_{(i+1,S)} - \theta_{(i-1,S)}, \tag{15}$$

such that

$$\theta_{ik} = \text{Arctg}\left[\frac{(y_i - y_{(i-k)})}{(x_i - x_{(i-k)})}\right], \tag{16}$$

where, (x_i, y_i) are the point coordinates p_i of the analyzed curve (i.e., a point sequence in the skeleton), and S is a smoothing factor (i.e., optimum interval for which quantization noise is attenuated, and meaningful details are conserved in each point of the curve). To get a well-smoothed curve, S must be in the range $5 \leq S \leq 15$. After many attempts, the best value for S was fixed to 7.

(iv) *Diacritic points*: are the black pixels having 0 foreground neighbour with their location (above or below the baseline). This type of point characterizes characters having a secondary part such as ("ب" and "ن").

(v) *Loops*: represent the skeleton inner contours with the information reflecting their partial or complete including inside the frame.

6.3. Vector quantization

Because we use discrete HMMs, we have to map each continuous feature vector representing a frame to a discrete symbol. This mapping is done by a procedure called vector quantization that implements the *LBG* [40] variant of the *K-means* algorithm. The *LBG* algorithm partitions the feature vectors representing the training samples into several classes, where each class is represented by its centroid which is a 42-dimensional vector. Then, it considers the index of each centroid as a codebook symbol. The best codebook size has been empirically fixed to 84.

7. WORD MODEL TRAINING AND CLASSIFICATION

Word model training is carried out to build up an HMM with explicit state duration for each word in the lexicon. This task can be done by two methods. In the first method (*whole model training*) a different HMM is created for each word from the samples labeled by the word identity. With this method we must cope with the problem of insufficient training data. In the IFN/ENIT database [26] some words are relatively well represented through a few hundreds of samples, whereas other words are poorly represented with solely three samples. To overcome the problem of insufficient training data, the second method performs character model training (*analytical model training*) and the word model is built up by character model concatenation. This makes the system flexible with respect to a change of lexicon because any word is a string of characters. In this way, it is sufficient to have, in the training set, samples of characters composing the words to be modeled rather than samples of the words themselves. Furthermore, the number of parameters is kept lower because the word models share the parameters belonging to the same characters. This can improve the training quality given the same amount of training data. In our case, we have no letter samples, however we have word samples. As a result, we do not apply the training algorithm directly to letter models, but to their concatenations corresponding to the words in the training set. This is called *embedded training* and has two important advantages: the first one is that the characters are modeled when being part of a word (that is the actual shape of the characters in the cursive handwriting), the second one is that it is not necessary to segment the words into characters to perform the training.

Both methods of training have been used in experimental tests, and the system performances were reported according to each training method (cf. Section 8).

In our word modeling based on HMMs with explicit state duration, the state meaning is associated to a logical notion that is either the letter when performing whole model training or the subletter (i.e., grapheme) when performing analytical model training. As a result, the state number by HMM model is varied with respect to the modeled word length. For instance, when the state represents a letter, the HMM model of the word ("الشرايع") has 7 states (see Figure 6) while the HMM model related to the word ("القلعة الصغرَى") has 12 states. In the analytical model training, each character shape is modeled by HMM having 4 states. Also, characters with additional marks (Hamza, Chedda, etc.) and ligatures are labeled and modeled separately. Subsequently, we have up to 160 different HMM models related to 28 Arabic basic characters.

The recognition lexicon is structured as tree, this allows efficient sharing of the character HMM models between the words, and hence reduces the storage space and processing time.

For each skeleton point p_i between two feature points, and having 2 black neighbours ***do***
{
1- Compute the smoothed global curvature sum *SGCS1* of the points sequence prior to p_i.
2- Compute the smoothed global curvature sum *SGCS2* of the points sequence following p_i.
3- ***If*** (*SGCS*1 > 0 and *SGCS*2 > 0) or (*SGCS*1 < 0 and *SGCS*1 < 0) ***then*** p_i is a ***cusp point***.
}

Algorithm 1: Cusp points detection.

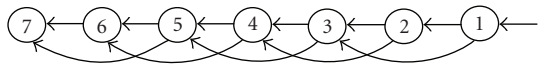

Figure 6: A right-to-left HMM with explicit state duration and interstate skips for the word "الشرايع".

The HMM topology is right to left with sole transitions to the next state or the one allowing for skipping of a single state. The state self-transitions are substituted by the explicit state duration.

Training and classification are basically done by the aforementioned modified version of the Viterbi algorithm. In the training stage, a segmental k-mean algorithm [23] is performed. In each iteration, only the state-vector assignments resulting from the best path obtained from applying the Viterbi algorithm are used to reestimate the model parameters. Moreover, we use formulas (6) and (10) to readjust parameters of state duration probability distributions.

8. RESULTS AND DISCUSSIONS

To test our system, we have carried out several experiments on the IFN/ENIT [26] benchmark database. This database consists of 26459 Arabic words written by 411 different writers, related to a lexicon of 946 Tunisian town/village names. Four distinct datasets (a, b, c, d) are predefined in the database, and the ground truth of the character shape level is available for each database sample. Therefore, character model building is practical. As it is recommended in [26], three datasets were used for training and one set for testing. Several experiments were carried out in order to measure the effect of the following issues on the recognition performance of the system: (1) the distribution of the explicit state duration; (2) the segmentation procedure into frames; (3) the word model training method. They were performed by selecting each time three datasets for training and one dataset for testing (the total number of possible combinations is four). Tables 1 and 2 summarize the mean results of these tests. The best results are graphically illustrated by Figure 7.

The above results show that HMMs with explicit state duration are more efficient for modeling unconstrained Arabic handwriting, compared to classical HMMs. The average performance gain is 11.07% (resp., 5.72) in top 1 with Gamma distribution and a *whole* (resp., *analytical*) *word model training method* with the best recognition rate in top 1 of 89.57% (resp., 90.02%) when using the datasets (a, b, d)

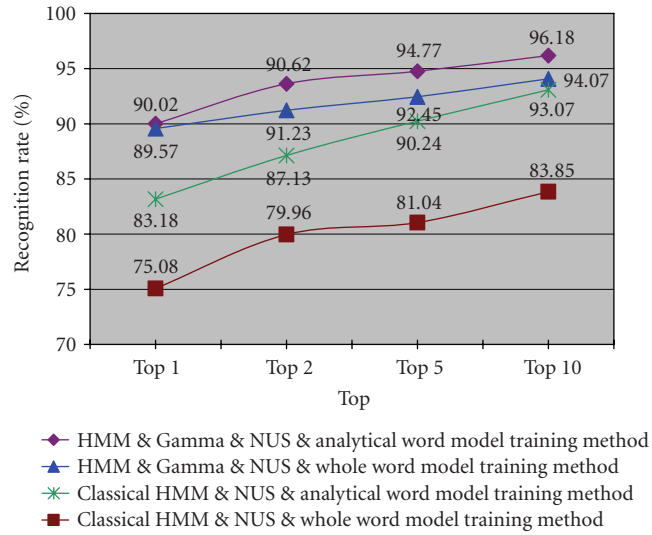

Figure 7: The best recognition performances of each training method which are obtained with the dataset c (6477 images) for test; NUS: nonuniform segmentation

for training and the data set (c) for generalization. Figure 8 shows some errors which can be avoided when using HMMs with explicit state duration. With classical HMMs: the word (see Figure 8(a)) "المنزه 8" was recognized as "المنزه 9" by confusing "8" with "9"; the word (see Figure 8(b)) 'شغّال' was recognized as 'نحّال' by confusing "ش" with "ن", and "ع" with "ح"; and the word (see Figure 8(c)) 'دقّة' was recognized as "دقة" by confusing "ز" with "د".

Gamma distribution seems to be more efficient for state duration modeling. Such behaviors can be attributed to its statistical proprieties and to the appropriateness of the data used for estimating its parameters. The discrete Poisson distribution results are less accurate than those of Gauss and Gamma. This fact can be explained by insufficient training data for some words which are needed to estimate the one parameter of Poisson distribution sufficiently well. On the other hand, the nonuniform segmentation scheme is more suitable than the uniform one because the nonuniform segmentation almost gives rise to a frame whose shape represents a complete character or a subcharacter. By contrast, the uniform segmentation can always produce a frame representing a partial combination of 2 characters.

(a) (b) (c)

Figure 8: Example of samples that are misrecognized by classical HMMs but they are correctly recognized by HMMs with explicit state duration since the character duration allows accurate character identification.

Table 1: Mean recognition results using each time 3 datasets for training and one data set for testing, and a *whole word model training method*.

State duration distribution	Segmentation Type	Top 1	Top 2	Top 5	Top 10
Classical HMMs	Uniform	69.56	74.02	75.76	79.84
	Nonuniform	72.94	76.76	79.57	81.29
HMMs and Poisson	Uniform	74.58	77.57	78.57	82.50
	Nonuniform	77.52	82.09	83.04	86.12
HMMs and Gauss	Uniform	79.11	82.91	84.15	86.94
	Nonuniform	81.75	84.84	85.75	88.36
HMMs and Gamma	Uniform	84.99	87.88	88.89	91.03
	Nonuniform	88.12	89.92	91.28	93.19

Table 2: Mean recognition results using each time 3 datasets for training and one data set for testing, and an *analytical word model training method*.

State Duration Distribution	Segmentation Type	Top 1	Top 2	Top 5	Top 10
Classical HMMs	Uniform	79.71	84.04	87.66	90.74
	Nonuniform	81.79	85.16	88.83	91.63
HMMs and Poisson	Uniform	80.78	85.18	88.49	91.74
	Nonuniform	82.79	85.65	89.69	92.35
HMMs and Gauss	Uniform	81.97	86.04	89.33	92.68
	Nonuniform	83.44	86.90	90.06	93.22
HMMs and Gamma	Uniform	85.88	88.78	90.11	93.50
	Nonuniform	88.79	91.25	92.93	95.02

Table 3: Comparison with other word recognition systems which are presented in [41]: recognition results in % with the dataset *d* (6735 images).

System	Top 1	Top 5	Top 10
ICRA	88.95	94.22	95.01
TH-OCR	30.13	41.95	46.59
UOB	85.00	91.88	93.56
REAM*	89.06	99.15	99.62
ARAB-IFN	87.94	91.42	95.62
Proposed system	**89.08**	**93.27**	**95.98**

* The system is trained on a reduced set with 1000 names.

In all performed tests, the analytical word model training method has performed better than the holistic one. This is due to the fact that the latter is shackled by the problem of insufficient training data. We point out that our best results outperform those reported recently on the same database in the international competition in Arabic handwriting recognition systems at ICDAR 2005 [41] (see Table 3).

Most of the recognition errors of the proposed system can be attributed to failure in baseline detection method and to the poor quality of some data samples. Also, using discrete HMMs with extracted features that are naturally continuous and that need to be quantized is problematic because the discriminating power of these features is altered by the vector quantization procedure.

9. CONCLUSION

We have proposed a segmentation-free method for offline unconstrained Arabic handwritten word recognition using HMMs with different explicit distribution for state duration, and combining statistical and structural features extracted by a sliding window approach that operates according to two segmentation (uniform and nonuniform) schemes into frames. The word model training has been done by two methods. In the first method, the unit directly targeted by the training process is the word as a whole, whereas the second method performs the training of the character shapes based on the word samples without explicit segmentation into characters (*embedded training*), and the word model is built up by concatenating its character models. The results obtained are very promising and have shown that the explicit state duration modeling within HMM framework can improve the recognition rate significantly. Moreover, continuous distributions (i.e., Gamma and Gauss) of state duration are more suitable than discrete ones (i.e., Poisson) for Arabic handwriting modeling, and the nonuniform segmentation scheme is more recommended. The main drawback of discrete HMMs is the imperfect observation probability estimation. Hence, our foreseen perspective to surpass this problem is searching an appropriate integration method of artificial neural networks (ANNs) to discrete HMMs with explicit state duration for best estimation of observation probability. Also, one may improve the system performance by using continuous HMMs that will allow a straightforward modeling of the handwriting features without the critical vector quantization required for discrete HMMs.

ACKNOWLEDGMENTS

The authors would like to express their gratitude to the Agence Universitaire de la Francophonie (AUF) for the financial support of this research. They are also indebted to Dr. M.T. Khadir, member of LRI Laboratory, Annaba University and Dr. M. Ramdani, associate professor at Electronic Department, Annaba University for helping to improve the paper quality.

REFERENCES

[1] A. L. Koerich, R. Sabourin, and C. Y. Suen, "Large vocabulary off-line handwriting recognition: a survey," *Pattern Analysis and Applications*, vol. 6, no. 2, pp. 97–121, 2003.

[2] R. Plamondon and S. N. Srihari, "On-line and off-line handwriting recognition: a comprehensive survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 1, pp. 63–84, 2000.

[3] A. Vinciarelli, "A survey on off-line cursive word recognition," *Pattern Recognition*, vol. 35, no. 7, pp. 1433–1446, 2002.

[4] M. A. El-Yacoubi, M. Gilloux, R. Sabourin, and C. Y. Suen, "An HMM-based approach for off-line unconstrained handwritten word modeling and recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 21, no. 8, pp. 752–760, 1999.

[5] M. A. El-Yacoubi, M. Gilloux, and J. M. Bertille, "A statistical approach for phrase location and recognition within a text line: an application to street name recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 2, pp. 172–188, 2002.

[6] A. L. Koerich, R. Sabourin, and C. Y. Suen, "Recognition and verification of unconstrained handwritten words," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, no. 10, pp. 1509–1522, 2005.

[7] A. Vinciarelli, S. Bengio, and H. Bunke, "Off-line recognition of unconstrained handwritten texts using HMMs and statistical language models," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 26, no. 6, pp. 709–720, 2004.

[8] A. Amin, H. Al-Sadoun, and S. Fischer, "Hand-printed Arabic character recognition system using an artificial network," *Pattern Recognition*, vol. 29, no. 4, pp. 663–675, 1996.

[9] E. N. B. Amara and F. Bouslama, "Classification of Arabic script using multiple sources of information: state of the art and perspectives," *International Journal on Document Analysis and Recognition*, vol. 5, no. 1, pp. 195–212, 2003.

[10] L. M. Lorigo and V. Govindaraju, "Offline Arabic handwriting recognition: a survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28, no. 5, pp. 712–724, 2006.

[11] L. Zheng, A. H. Hassin, and X. Tang, "A new algorithm for machine printed Arabic character segmentation," *Pattern Recognition Letters*, vol. 25, no. 15, pp. 1723–1729, 2004.

[12] L. Yang, B. K. Widjaja, and R. Prasad, "Application of hidden Markov models for signature verification," *Pattern Recognition*, vol. 28, no. 2, pp. 161–170, 1995.

[13] A. L. Koerich, R. Sabourin, and C. Y. Suen, "Lexicon-driven HMM decoding for large vocabulary handwriting recognition with multiple character models," *International Journal on Document Analysis and Recognition*, vol. 6, no. 2, pp. 126–144, 2003.

[14] G. Kirn and V. Govindaraju, "A lexicon driven approach to handwritten word recognition for real-time applications," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 19, no. 4, pp. 366–379, 1997.

[15] A. Benouareth, A. Ennaji, and M. Sellami, "Semi-continuous HMMs with explicit state duration applied to Arabic handwritten word recognition," in *Proceedings of the International Workshop on Frontiers in Handwriting Recognition (IWFHR '06)*, pp. 97–101, La Baule, France, October 2006.

[16] R. El-Hajj, L. Likforman-Sulem, and C. Mokbel, "Arabic handwriting recognition using baseline dependant features and hidden Markov modeling," in *Proceedings of the International Conference on Document Analysis and Recognition (ICDAR '05)*, vol. 2005, pp. 893–897, Seoul, Korea, August 2005.

[17] M. Pechwitz and V. Maergner, "HMM based approach for handwritten Arabic word recognition using the IFN/ENIT-database," in *Proceedings of the International Conference on Document Analysis and Recognition (ICDAR '03)*, pp. 890–894, Edinburgh, Scotland, August 2003.

[18] M. Pechwitz, V. Maergner, and H. El Abed, "Comparison of two different feature sets for offline recognition of handwritten Arabic words," in *Proceedings of the International Workshop on Frontiers in Handwriting Recognition (IWFHR '06)*, pp. 109–114, La Baule, France, October 2006.

[19] S. Madhvanath and V. Govindaraju, "The role of holistic paradigms in handwritten word recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 23, no. 2, pp. 149–164, 2001.

[20] N. Farah, L. Souici, and M. Sellami, "Classifiers combination and syntax analysis for Arabic literal amount recognition," *Engineering Applications of Artificial Intelligence*, vol. 19, no. 1, pp. 29–39, 2006.

[21] L. Souici and M. Sellami, "A hybrid neuro-symbolic approach for Arabic handwritten word recognition," *Journal of Advanced Computational Intelligence and Intelligent Informatics*, vol. 10, no. 1, 2006.

[22] K. M. Sayre, "Machine recognition of handwritten words: a project report," *Pattern Recognition*, vol. 5, no. 3, pp. 213–228, 1973.

[23] L. R. Rabiner, "A tutorial on hidden Markov models and select applications in speech recognition," *Proceedings of the IEEE*, vol. 77, no. 2, pp. 257–286, 1989.

[24] A. Benouareth, A. Ennaji, and M. Sellami, "HMMs with explicit state duration applied to handwritten Arabic word recognition," in *Proceedings of the International Conference on Pattern Recognition*, vol. 2, pp. 897–900, 2006.

[25] G. D. Forney, "The viterbi algorithm," *Proceedings of the IEEE*, vol. 61, no. 3, pp. 268–278, 1973.

[26] M. Pechwitz, S. S. Maddouri, V. Maergner, N. Ellouze, and H. Amiri, "IFN/ENIT—database of handwritten Arabic words," in *Proceedings of the Colloque International Francophone sur l'Écrit et le Document (CIFED '02)*, pp. 129–136, Hammamet, Tunisia, October 2002.

[27] S. R. Veltman and R. Prasad, "Hidden Markov models applied to on-line handwritten isolated character recognition," *IEEE Transactions on Image Processing*, vol. 3, no. 3, pp. 314–318, 1994.

[28] M.-Y. Chen, A. Kundu, and S. N. Srihari, "Variable duration hidden Markov model and morphological segmentation for handwritten word recognition," *IEEE Transactions on Image Processing*, vol. 4, no. 12, pp. 1675–1688, 1995.

[29] A. M. Gillies, "Cursive word recognition using hidden Markov models," in *Proceedings of the 5th U.S. Postal Service Advanced Technology Conference*, pp. 557–562, Washington, DC, USA, November 1992.

[30] M. Mohamed and P. Gader, "Handwritten word recognition using segmentation-free hidden Markov modeling and segmentation-based dynamic programming techniques," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 18, no. 5, pp. 548–554, 1996.

[31] A. Vinciarelli and S. Bengio, "Off-line cursive word recognition using continuous density HMMs trained with PCA or ICA features," in *Proceedings of the International Conference on Pattern Recognition (ICPR '02)*, pp. 493–498, Quebec City, Canada, August 2002.

[32] Y. Bengio, Y. LeCun, and D. Henderson, "Globally trained handwritten word recognizer using spatial representation, space displacement neural networks and hidden Markov models," in *Advances in Neural Information Processing Systems*, J. D. Cowan, G. Tesauro, and J. Alspector, Eds., vol. 6, pp. 937–944, Morgan Kaufmann, San Mateo, Calif, USA, 1994.

[33] N. B. Amara and A. Belaid, "Printed PAW recognition based on planar hidden Markov models," in *Proceedings of the International Conference on Pattern Recognition (ICPR '98)*, pp. 25–29, Sydney, Australia, December 1998.

[34] E. Levin and R. Pieraccini, "Dynamic planar warping for optical character recognition," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP '92)*, vol. 3, pp. 149–152, 1992.

[35] M. S. Khorsheed, "Recognising handwritten Arabic manuscripts using a single hidden Markov model," *Pattern Recognition Letters*, vol. 24, no. 14, pp. 2235–2242, 2003.

[36] M. J. Russell and R. K. Moore, "Explicit modelling of state occupancy in hidden Markov models for automatic speech recognition," in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP '85)*, pp. 5–8, Tampa, Fla, USA, March 1985.

[37] S. E. Levinson, "Continuously variable duration hidden Markov models for automatic speech recognition," *Computer Speech and Language*, vol. 1, no. 1, pp. 29–45, 1986.

[38] T. Pavlidis, *Algorithms for Graphic and Image Processing*, Computer science press, Rockville, Md, USA, 1982.

[39] H. Freeman and L. S. Davis, "Corner-finding algorithm for chain-coded curves," *IEEE Transactions on Computers*, vol. C-26, no. 3, pp. 297–303, 1977.

[40] Y. Linde, A. Buzo, and R. M. Gray, "An algorithm for vector quantizer design," *IEEE Transactions on Computers*, vol. 36, pp. 85–95, 1980.

[41] V. Märgner, M. Pechwitz, and H. El Abed, "ICDAR 2005 Arabic handwriting recognition competition," in *Proceedings of the International Conference on Document Analysis and Recognition, (ICDAR '05)*, vol. 2005, pp. 70–74, 2005.