Handwriting word recognition using windowed Bernoulli HMMs

Adrià Giménez, Ihab Khoury, Jesús Andrés-Ferrer and Alfons Juan

DSIC/ITI, Universitat Politècnica de València, Camí de Vera s/n, 46022 València, Spain

Abstract

Hidden Markov Models (HMMs) are now widely used for off-line handwriting recognition in many languages. As in speech recognition, they are usually built from shared, embedded HMMs at symbol level, where state-conditional probability density functions in each HMM are modeled with Gaussian mixtures. In contrast to speech recognition, however, it is unclear which kind of features should be used and, indeed, very different features sets are in use today. Among them, we have recently proposed to directly use columns of raw, binary image pixels, which are directly fed into embedded Bernoulli (mixture) HMMs, that is, embedded HMMs in which the emission probabilities are modeled with Bernoulli mixtures. The idea is to by-pass feature extraction and to ensure that no discriminative information is filtered out during feature extraction, which in some sense is integrated into the recognition model. In this work, column bit vectors are extended by means of a sliding window of adequate width to better capture image context at each horizontal position of the word image. Using these *windowed* Bernoulli mixture HMMs,

Email address: `{agimenez,ialkhoury,jandres,ajuan}@dsic.upv.es` (Adrià Giménez, Ihab Khoury, Jesús Andrés-Ferrer and Alfons Juan)

good results are reported on the well-known IAM and RIMES databases of Latin script, and in particular, state-of-the-art results are provided on the IfN/ENIT database of Arabic handwritten words.

Keywords: HTR, Bernoulli HMM, Latin, Arabic, Sliding window

1. Introduction

Hidden Markov Models (HMMs) are now widely used for off-line handwriting recognition in many languages and, in particular, in languages with Latin and Arabic scripts (Dehghan et al., 2001; Günter and Bunke, 2004; Märgner and El Abed, 2007, 2009; Grosicki and El Abed, 2009). Following the conventional approach in speech recognition (Rabiner and Juang, 1993), HMMs at global (line or word) level are built from shared, *embedded* HMMs at character (subword) level, which are usually simple in terms of number of states and topology. In the common case of real-valued feature vectors, state-conditional probability (density) functions are modeled as Gaussian mixtures since, as with finite mixture models in general, their complexity can be easily adjusted to the available training data by simply varying the number of components.

After decades of research in speech recognition, the use of certain real-valued speech features and embedded Gaussian (mixture) HMMs is a de-facto standard (Rabiner and Juang, 1993). However, in the case of handwriting recognition, there is no such a standard and, indeed, very different sets of features are in use today. In Giménez and Juan (2009) we proposed to bypass feature extraction and to directly feed columns of raw, binary pixels into *embedded Bernoulli (mixture) HMMs (BHMMs),* that is, embedded HMMs in which the emission probabilities are modeled with Bernoulli mixtures. The basic idea is to ensure that no discriminative information is filtered out during feature extraction, which in some sense is integrated into the recognition model. In Giménez et al. (2010), we improved our basic approach by using a sliding window of adequate width to better capture image context at each horizontal position of the text image. This improvement, to which we refer as *windowed BHMMs,* achieved very competitive results on the well-known IfN/ENIT database of Arabic town names (Pechwitz et al., 2002).

Although windowed BHMMs achieved good results on IfN/ENIT, it was clear to us that text distortions are more difficult to model with wide windows than with narrow (e.g. one-column) windows. In order to circumvent this difficulty, we have considered new, adaptative window sampling techniques, as opposed to the conventional, direct strategy by which the sampling window center is applied at a constant height of the text image and moved horizontally one pixel at a time. More precisely, these adaptative techniques can be seen as an application of the direct strategy followed by a *repositioning* step by which the sampling window is repositioned to align its center to the center of gravity of the sampled image. This repositioning step can be done horizontally, vertically or in both directions. Although vertical repositioning was expected to have more influence on recognition results than horizontal repositioning, we decided to study both separately, and also in conjunction, so as to confirm this expectation.

In this paper, the repositioning techniques described above are introduced and extensively tested on different, well-known databases for off-line handwriting recognition. In particular, we provide new, state-of-the-art results on the IfN/ENIT database, which clearly outperform our previous results without repositioning (Giménez et al., 2010). Indeed, the first tests on IfN/ENIT of our windowed BHMM system with vertical repositioning were made at the ICFHR 2010 Arabic Handwriting Recognition Competition, where our system ranked first (Märgner and El Abed, 2010). Moreover, the test sets used in this competition were also used in a new competition at the ICDAR 2011 and none of the participants improved the results achieved by our system at the ICFHR 2010 conference (Märgner and El Abed, 2011). Apart from state-of-the-art results on IfN/ENIT, we also provide new empirical results on the IAM database of English words (Marti and Bunke, 2002) and the RIMES database of French words (Grosicki et al., 2009). Our windowed BHMM system with vertical repositioning achieves good results on both databases.

In what follows, we briefly review Bernoulli mixtures (Sec. 2), BHMMs (Sec. 3), maximum likelihood parameter estimation (Sec. 4) and *windowed BHMMs* repositioning techniques (Sec. 5). Empirical results are then reported in Sec. 6 and concluding remarks are given in Sec. 7.

2. Bernoulli Mixture

Let $\mathbf{o}$ be a D-dimensional feature vector. A finite mixture is a probability (density) function of the form:

$$P(\mathbf{o} \mid \mathbf{\Theta}) = \sum_{k=1}^{K} \pi_k \, P(\mathbf{o} \mid k, \mathbf{\Theta}_k) \ , \tag{1}$$

where K is the number of mixture components, π_k is the k-th component coefficient, and $P(\mathbf{o} \mid k, \mathbf{\Theta}_k)$ is the k-th component-conditional probability (density) function. The mixture is controlled by a parameter vector $\mathbf{\Theta}$ comprising the mixture coefficients and a parameter vector for the components, $\mathbf{\Theta}_k$. It can be seen as a generative model that first selects the k-th component with probability π_k and then generates $\mathbf{o}$ in accordance with $P(\mathbf{o} \mid k, \mathbf{\Theta}_k)$.

A Bernoulli mixture model is a particular case of (1) in which each component k has a D-dimensional Bernoulli probability function governed by its own vector of parameters or *prototype* $\mathbf{p}_k = (p_{k1}, \ldots, p_{kD})^t \in [0,1]^D$,

$$P(\mathbf{o} \mid k, \mathbf{\Theta}_k) = \prod_{d=1}^{D} p_{kd}^{o_d} (1 - p_{kd})^{1-o_d} \,, \tag{2}$$

where p_{kd} is the probability for bit d to be 1. Note that this equation is just the product of independent, unidimensional Bernoulli probability functions. Therefore, for a fixed k, it can not capture any kind of dependencies or correlations between individual bits.

3. Bernoulli HMM

Let $O = (\mathbf{o}_1, \ldots, \mathbf{o}_T)$ be a sequence of feature vectors. An HMM is a probability (density) function of the form:

$$P(O \mid \mathbf{\Theta}) = \sum_{q_1,\ldots,q_T} \prod_{t=0}^{T} a_{q_t q_{t+1}} \prod_{t=1}^{T} b_{q_t}(\mathbf{o}_t) \,, \tag{3}$$

where the sum is over all possible *paths* (state sequences) $q_0, \ldots, q_{T+1}$, such that $q_0 = I$ (special *initial* or *start* state), $q_{T+1} = F$ (special *final* or *stop* state), and $q_1, \ldots, q_T \in \{1, \ldots, M\}$, being M the number of regular (non-special) states of the HMM. On the other hand, for any regular states i and j, a_{ij} denotes the *transition* probability from i to j, while b_j is the *observation* probability (density) function at j.

A Bernoulli (mixture) HMM (BHMM) is an HMM in which the probability of observing the binary feature vector $\mathbf{o}_t$, when $q_t = j$, follows a Bernoulli mixture distribution for the state j

$$b_j(\mathbf{o}_t) = \sum_{k=1}^{K} \pi_{jk} \prod_{d=1}^{D} p_{jkd}^{o_{td}} (1 - p_{jkd})^{1-o_{td}} \,, \tag{4}$$

where o_{td} is the d-th bit of $\mathbf{o}_t$, π_{jk} is the prior of the k-th mixture component in state j, and p_{jkd} is the probability that this component assigns to o_{td} to be 1.

Consider the upper part of Fig. 1, where a BHMM example for the number 3 is shown, together with a binary image generated from it. It is a three-state model with single prototypes attached to states 1 and 2, and a two-component mixture assigned to state 3, where Bernoulli prototypes are depicted as a gray image (black=1, white=0, gray=0.5). It is worth noting that prototypes do not account for the whole digit realizations, but only for single columns. This column-by-column emission of feature vectors attempts to better model horizontal distortions at character level and, indeed, it is the usual approach in both speech and handwriting recognition when continuous-density (Gaussian mixture) HMMs are used. The reader can check that, by direct application of (3) and taking into account the existence of two different state sequences, the probability of generating the binary image generated in this example is 0.063.

As discussed in the introduction, BHMMs at global (line or word) level are built from shared, embedded BHMMs at character level. More precisely, let C be the number of different characters (symbols) from which global BHMMs are built, and assume that each character c is modeled with a dif-

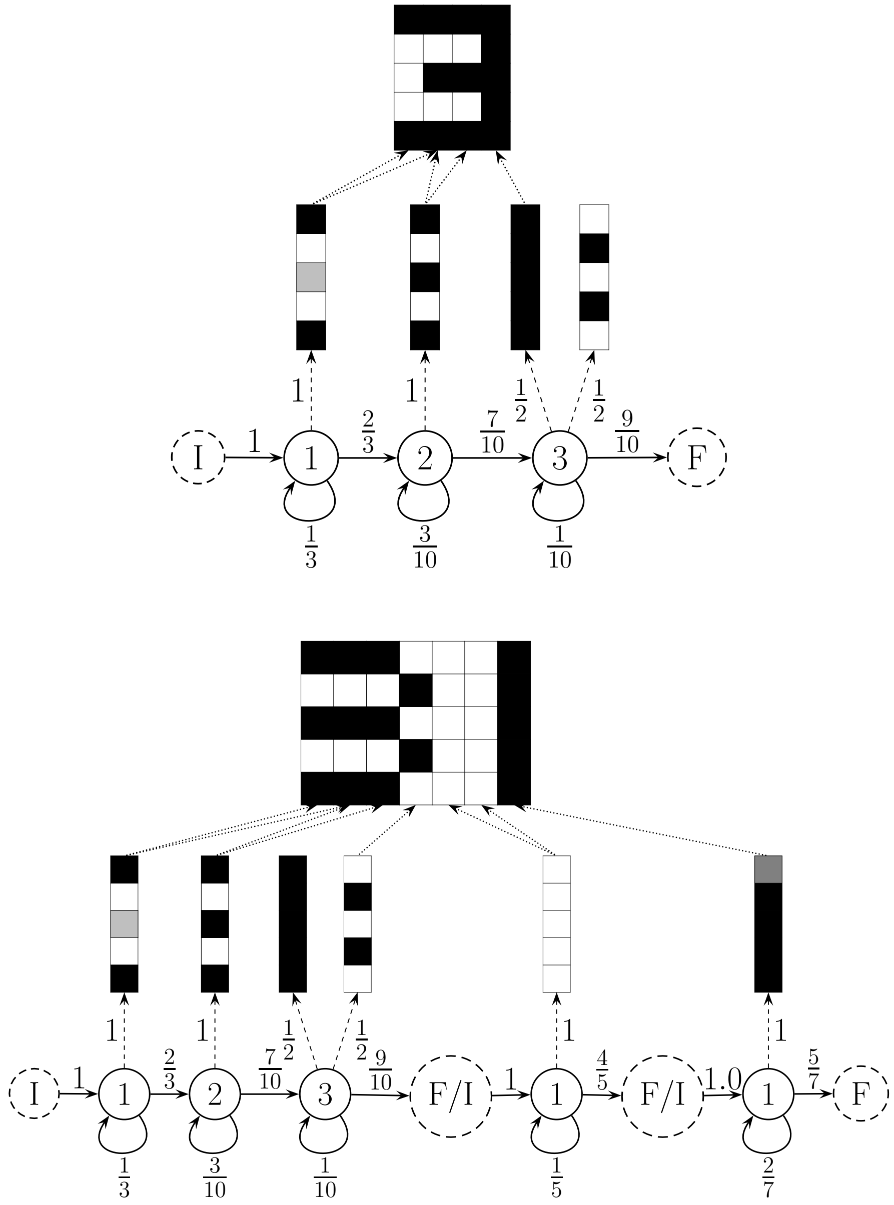


Figure 1: BHMM examples for the numbers 3 (top) and 31 (bottom), together with binary images generated from them. Note that the BHMM example for the number 3 is also embedded into the number 31 example. Bernoulli prototype probabilities are represented using the following color scheme: black=1, white=0,gray=0.5 and light gray=0.1.

ferent BHMM of parameter vector $\mathbf{\Theta}_c$. Let $\mathbf{\Theta} = \{\mathbf{\Theta}_1, \dots, \mathbf{\Theta}_C\}$, and let $O = (\mathbf{o}_1, \dots, \mathbf{o}_T)$ be a sequence of feature vectors generated from a sequence of symbols $S = (s_1, \dots, s_L)$, with $L \leq T$. The probability of O can be calculated, using embedded HMMs for its symbols, as:

$$P(O \mid S, \mathbf{\Theta}) = \sum_{i_1,\dots,i_{L+1}} \prod_{l=1}^{L} P(\mathbf{o}_{i_l}, \dots, \mathbf{o}_{i_{l+1}-1} \mid \mathbf{\Theta}_{s_l}) \,, \tag{5}$$

where the sum is carried out over all possible segmentations of O into L segments, that is, all sequences of indices $i_1, \dots, i_{L+1}$ such that

$$1 = i_1 < \cdots < i_L < i_{L+1} = T + 1;$$

and $P(\mathbf{o}_{i_l}, \dots, \mathbf{o}_{i_{l+1}-1} \mid \mathbf{\Theta}_{s_l})$ refers to the probability (density) of the l-th segment, as given by (3) using the HMM associated with symbol s_l.

Consider now the lower part of Fig. 1. An embedded BHMM for the number 31 is shown, which is the result of concatenating BHMMs for the digit 3, blank space and digit 1, in that order. Note that the BHMMs for blank space and digit 1 are simpler than that for digit 3. Also note that the BHMM for digit 3 is shared between the two embedded BHMMs shown in the figure. The binary image of the number 31 shown above can only be generated from two paths, as indicated by the arrows connecting prototypes to image columns, which only differ in the state generating the second image column (either state 1 or 2 of the BHMM for the first symbol). It is straightforward to check that, according to (5), the probability of generating this image is 0.0004.

4. Maximum Likelihood Parameter Estimation

Maximum likelihood estimation (MLE) of the parameters governing an embedded BHMM does not differ significantly from the conventional Gaussian case, and it is also efficiently performed using the well-known EM (Baum-Welch) re-estimation formulae (Rabiner and Juang, 1993; Young et al., 1995). Let $(O_1, S_1), \ldots, (O_N, S_N)$, be a collection of N training samples in which the n-th observation has length T_n, $O_n = (\mathbf{o}_{n1}, \ldots, \mathbf{o}_{nT_n})$, which corresponds to a sequence of L_n symbols ($L_n \leq T_n$), $S_n = (s_{n1}, \ldots, s_{nL_n})$. At iteration r, the E step requires the computation, for each training sample n, of its corresponding forward (α) and backward (β) recurrences (see Rabiner and Juang (1993)), as well as

$$z^{(r)}_{nltk}(j) = \frac{\pi^{(r)}_{s_{nl}jk} \prod_{d=1}^{D} {p^{(r)}_{s_{nl}jkd}}^{o_{ntd}} \left(1 - p^{(r)}_{s_{nl}jkd}\right)^{1-o_{ntd}}}{b^{(r)}_{s_{nl}j}(\mathbf{o}_{nt})}\,, \qquad (6)$$

for each t, k, j, l. In (6), $z^{(r)}_{nltk}(j)$ is the probability of $\mathbf{o}_{nt}$ to be generated in the k-th mixture component, given that $\mathbf{o}_{nt}$ has been generated in the j-th state of symbol s_l. The conditional probability function $b^{(r)}_{s_{nl}j}(\mathbf{o}_{nt})$ is analogous to that defined in (4).

In the M step, the Bernoulli prototype corresponding to the k-th component of the state j for character c has to be updated as

$$\mathbf{p}^{(r+1)}_{cjk} = \frac{1}{\gamma_{ck}(j)} \sum_{n} \frac{\sum_{l:s_{nl}=c} \sum_{t=1}^{T_n} \xi^{(r)}_{nltk}(j)\mathbf{o}_{nt}}{P(O_n \mid S_n, \mathbf{\Theta}^{(r)})}\,, \qquad (7)$$

where $\gamma_{ck}(j)$ is a normalization factor

$$\gamma_{ck}(j) = \sum_{n} \frac{\sum_{l:s_{nl}=c} \sum_{t=1}^{T_n} \xi^{(r)}_{nltk}(j)}{P(O_n \mid S_n, \mathbf{\Theta}^{(r)})}\,, \qquad (8)$$

and $\xi_{nltk}^{(r)}(j)$ is the probability of O_n when the t-th feature vector of the n-th sample corresponds to symbol s_l and is generated by the k-th component of the state j,

$$\xi_{nltk}^{(r)}(j) = \alpha_{nlt}^{(r)}(j) z_{nltk}^{(r)}(j) \beta_{nlt}^{(r)}(j) \,. \tag{9}$$

Similarly, the k-th component coefficient of the state j in the HMM for character c is updated by

$$\pi_{cjk}^{(r+1)} = \frac{1}{\gamma_c(j)} \sum_n \frac{\sum_{l:s_{nl}=c} \sum_{t=1}^{T_n} \xi_{nltk}^{(r)}(j)}{P(O_n \mid S_n, \mathbf{\Theta}^{(r)})} \,, \tag{10}$$

where $\gamma_c(j)$ is a normalization factor

$$\gamma_c(j) = \sum_n \frac{\sum_{l:s_{nl}=c} \sum_{t=1}^{T_n} \alpha_{nlt}^{(r)}(j) \beta_{nlt}^{(r)}(j)}{P(O_n \mid S_n, \mathbf{\Theta}^{(r)})} \,. \tag{11}$$

Finally, it is well-known that MLE tends to overtrain the models. In order to amend this problem Bernoulli prototypes are smoothed by linear interpolation with a flat (uniform) prototype, $\mathbf{0.5}$,

$$\tilde{\mathbf{p}} = (1 - \delta)\, \mathbf{p} + \delta\, \mathbf{0.5} \,, \tag{12}$$

where δ is usually optimized in a validation set or fixed to a sensible value such as $\delta = 10^{-6}$

5. Windowed BHMMs

Given a binary image normalized in height to H pixels, we may think of a feature vector $\mathbf{o}_t$ as its column at position t or, more generally, as a concatenation of columns in a window of W columns in width, centered at position

t. This generalization has no effect neither on the definition of BHMM nor on its MLE, although it would be very helpful to better capture the image context at each horizontal position of the image. As an example, Fig. 2 shows a binary image of 4 columns and 5 rows, which is transformed into a sequence of four 15-dimensional feature vectors (first row) by application of a sliding window of width 3. For clarity, feature vectors are depicted as 3×5 subimages instead of 15-dimensional column vectors. Note that feature vectors at positions 2 and 4 would be indistinguishable if, as in our previous approach, they were extracted with no context ($W = 1$).

Although one-dimensional, "horizontal" HMMs for image modeling can properly capture non-linear horizontal image distortions, they are somewhat limited when dealing with vertical image distortions, and this limitation might be particularly strong in the case of feature vectors extracted with significant context. To overcome this limitation, we have considered three methods of window *repositioning* after window extraction: *vertical, horizontal,* and *both.* The basic idea is to first compute the center of mass of the extracted window, which is then repositioned (translated) to align its center to the center of mass. This is done in accordance with the chosen method, that is, horizontally, vertically, or in both directions. Obviously, the feature vector actually extracted is that obtained after repositioning. An example of feature extraction is shown in Fig. 2 in which the standard method (no repositioning) is compared with the three methods repositioning methods considered.

It is helpful to observe the effect of the repositioning with real data. Fig. 3 shows the sequence of feature vectors extracted from a real sample of the

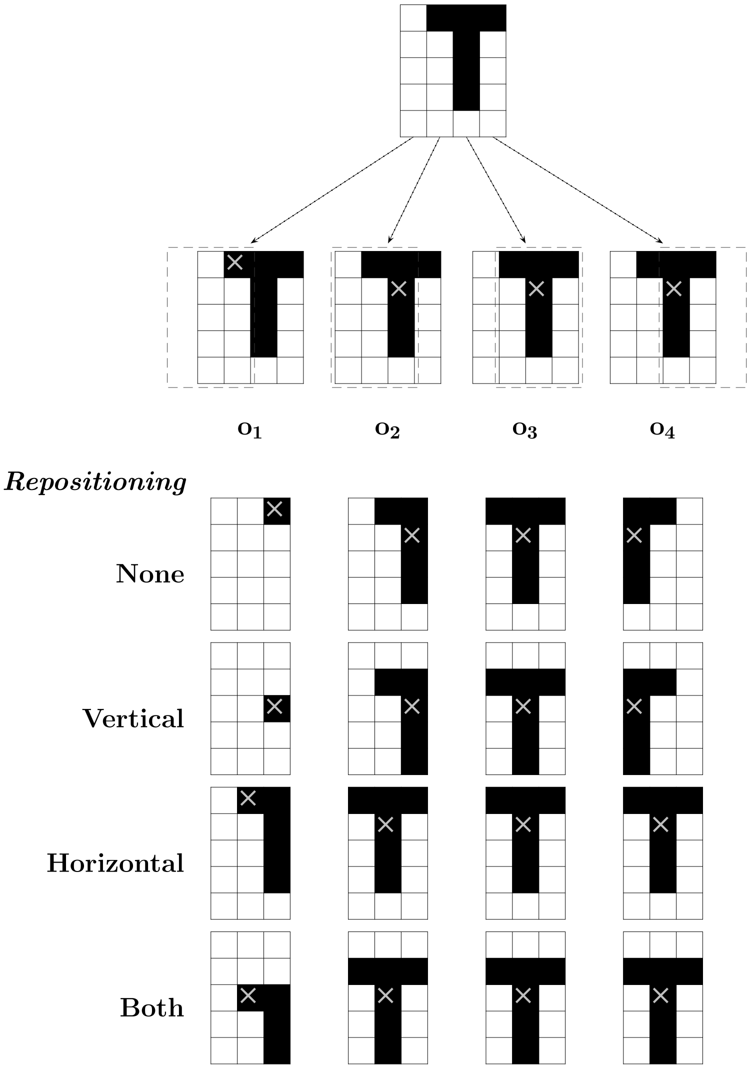


Figure 2: Example of transformation of a 4 × 5 binary image (top) into a sequence of 4 15-dimensional binary feature vectors $O = (\mathbf{o}_1, \mathbf{o}_2, \mathbf{o}_3, \mathbf{o}_4)$ using a window of width 3. After window extraction (illustrated under the original image), the standard method (no repositioning) is compared with the three repositioning methods considered: vertical, horizontal, and both directions. Mass centers of extracted windows are also indicated.

IFN/ENIT database, with and without (both) repositioning. As intended, (vertical or both) repositioning has the effect of normalizing vertical image distortions, especially translations.

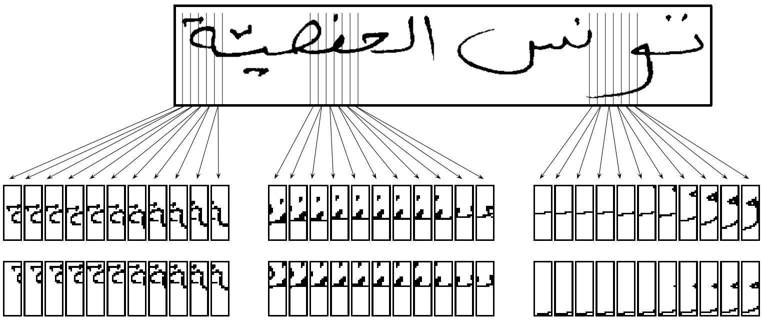


Figure 3: Original sample *pf069_011* from IFN/ENIT database (top) and its sequence of feature vectors produced with and without (both) repositioning (center and bottom, respectively).

6. Experiments

Our windowed BHMMs and the repositioning techniques described above were tested on three well-known databases of handwritten words: the IfN/ENIT database (Pechwitz et al., 2002), IAM words (Marti and Bunke, 2002) and RIMES (Grosicki et al., 2009). In what follows, we describe experiments and results in each database separately.

6.1. IfN/ENIT

The IfN/ENIT database of Arabic handwritten Tunisian town names is a widely used database to compare Arabic handwriting recognition systems (Pechwitz et al., 2002). As in the Arabic handwriting recognition competition held at ICDAR in 2007 (Märgner and El Abed, 2007), we used

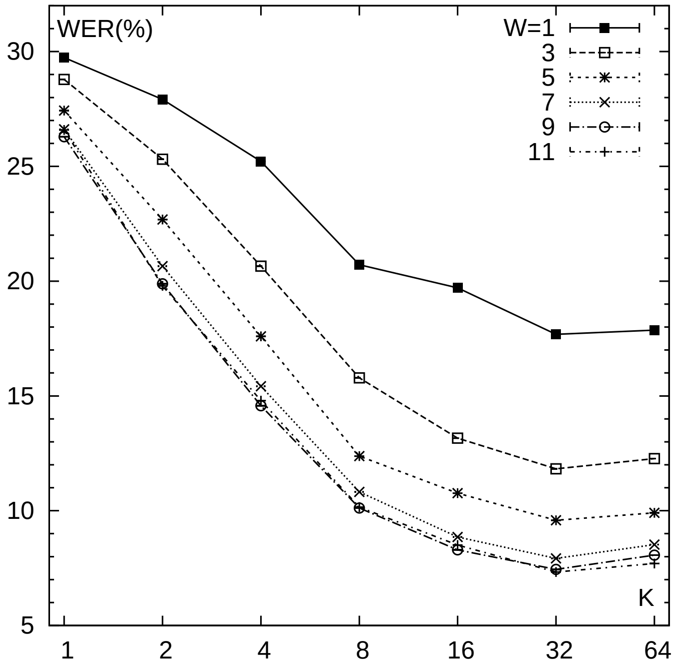
IfN/ENIT version 2.0, patch level 1e (v2.0p1e). It comprises 32492 Arabic word images written by more than 1000 different writers, from a lexicon of 937 Tunisian town/village names. For the experiments reported below, each image was first rescaled in height to $D = 30$ rows, while keeping the original aspect ratio, and then binarized using Otsu's binarization method. The resulting set of binary images was partitioned into five folds labeled as a, b, c, d and e, as defined in (Märgner and El Abed, 2007).

In a first series of experiments, we tried different values for the sliding window width W (1, 3, 5, 7, 9 and 11) and also different values for number of mixture components per state K (1, 2, 4, 8, 16, 32, 64). However, taking into account our previous, preliminary results in Khoury et al. (2010), we only tried BHMMs with 6 states as character models. For $K = 1$, BHMMs were initialized by first segmenting the training set with a "neutral" model analogous to that in Young et al. (1995), and then using the resulting segments to perform a Viterbi initialization. For $K > 1$, BHMMs were initialized by splitting the mixture components of the models trained with $K/2$ mixture components per state. In each case, 4 EM iterations were run after initialization. As usual with conventional HMM systems (Young et al., 1995), the Viterbi algorithm was used in combination with a table of prior probabilities so as to find the most probable transcription (class) of each test image.

Fig. 4 (top) shows the Word Error Rate (WER%) as a function of the number of mixture components, for varying sliding window widths. Each WER estimate (plotted point) was obtained by cross-validation with the first 4 standard folds (abcd). It is clear that the use of sliding windows improves the results to a large extent. Specifically, the best result, 7.4%, is obtained for $W = 9$ and $K = 32$, although very similar results are obtained for $W = 7$ and $W = 11$. It is worth noting that the best result achieved with no sliding windows ($W = 1$) is 17.7%, that is, 10 absolute points above of the best result achieved with sliding windows.

For better understanding of BHMM character models, the model for character خ, trained from folds abc with $W = 9$ and $K = 32$, is (partially) depicted in Fig. 5 (top) together with its Viterbi alignment with a real image of the character خ drawn from sample *de05_007*. As in Fig. 1 (bottom), Bernoulli prototypes are represented as gray images where the gray level of each pixel represents the probability of its corresponding pixel to be black (white = 0 and black = 1). From these prototypes, it can be seen that each state from right to left accounts for a different local part of خ, as if the sliding window was moving smoothly from right to left. Also, note that the main stroke of the character خ appears almost neatly drawn in most prototypes, whereas its upper dot appears blurred, probably due to a comparatively higher variability in window position.

Following previous results in Khoury et al. (2010), in the first series of experiments discussed above we only tried BHMMs with 6 states. However, in a recent work by Dreuw et al. (2009) where conventional (Gaussian) HMMs are tested on IfN/ENIT, the authors claim that Arabic script might be better modeled with character HMMs of variable number of states since Arabic letters are highly variable in length (as opposed to Latin letters). In oorder to check this claim, experiments similar to those described above were repeated with character BHMMs of different number of states. To decide on the number of states of each character BHMM, we first trained BHMMs of 4



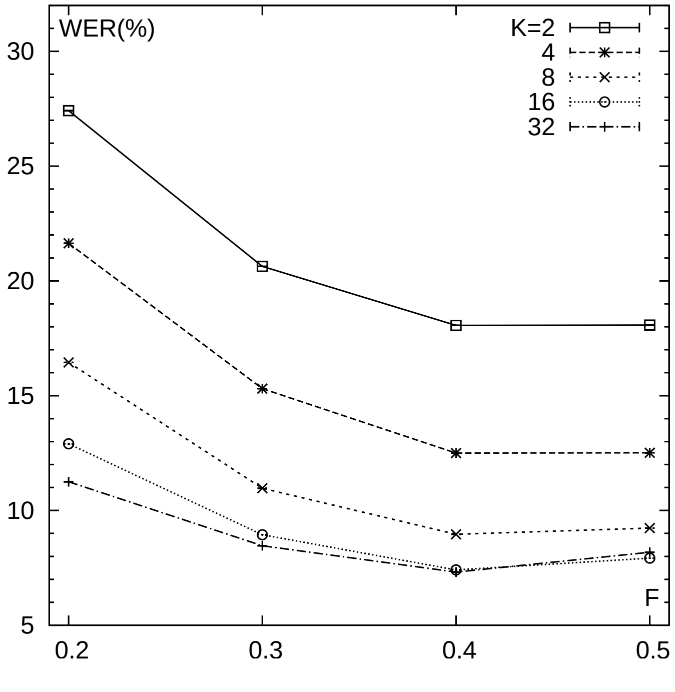


Figure 4: WER(%) on IfN/ENIT as a function of: the number of mixture components (K) for several sliding window widths (W) (top); and the factor F for varying values of the number of mixture components (K) (bottom).

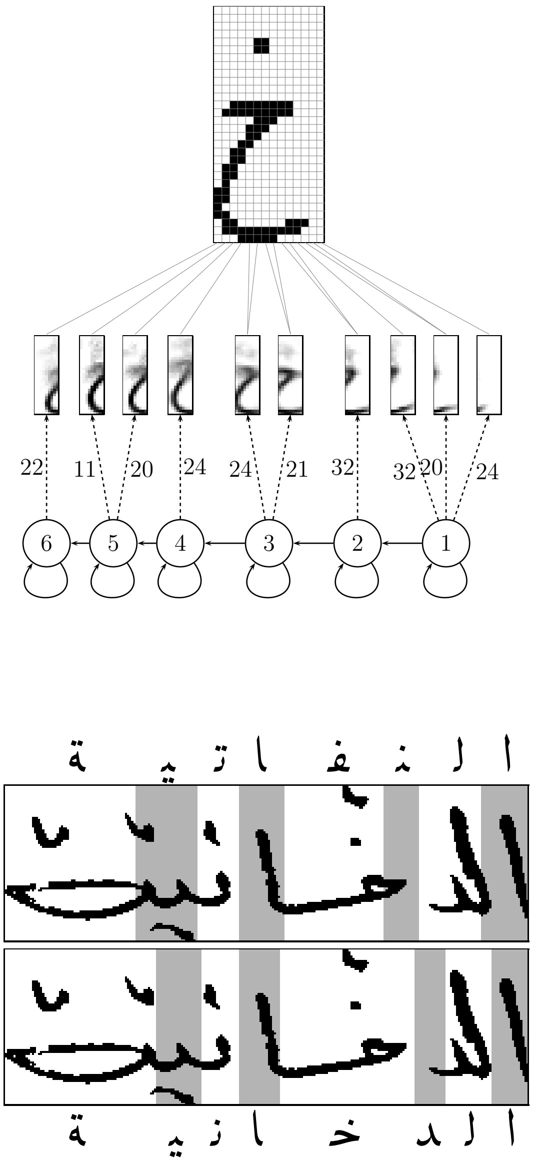


Figure 5: Top: BHMM for character خ, trained from folds abc with $W = 9$ and $K = 32$, together with its Viterbi alignment with a real image of the character خ, drawn from sample *de05_007*. Bottom: the sample *dm33_037* is incorrectly recognized as النفَاتية with BHMMs of 6 states, but correctly recognized as الدخّانية with BHMMs of variable number of states; the background color is used to represent Viterbi alignments at character level.

states which were then used to segment each training sample by the Viterbi algorithm. For each character c, its average length $\bar{T}_c$ was computed over all occurrences of c in the segmented training data. Then, its number of states was set to $F \cdot \bar{T}_c$, where F is a *factor* measuring the average number of states that are required to emit a feature vector. The inverse of F, $\frac{1}{F}$ is easily understood since it can be interpreted as a *state load,* that is, the average number of feature vectors that are emitted in each state. For instance, a factor of $F = 0.2$ implies that only a fraction of 0.2 states is required to emit a feature vector or, alternatively, that $\frac{1}{0.2} = 5$ feature vectors are emitted on average in each state.

Fig. 4 (bottom) shows the WER as a function of the factor F, for different number of mixture components K and a window width of $W = 9$ (with which we obtained the best results in the previous experiments). The best result now, 7.3% (obtained with $F = 0.4$ and $K = 32$), is similar to the 7.4% obtained with 6 states per character. Therefore, in our case, the use of character models of different number of states does not lead to a significant improvement of the results.

Although the results with variable number of states do not lead to significant improvements, it is interesting to see that there are cases in which, as expected, Arabic letters are better modeled with them. An example is shown in Fig. 5 (bottom) using the sample *dm33_037*. This sample was recognized using BHMMs with $W = 9$, $K = 32$ and both, 6 states (top) and variable number of states with $F = 0.4$ (bottom). In both cases, the recognized word is Viterbi-aligned at character level (background color). Although it was incorrectly recognized as النفَاتية with BHMMs of 6 states (top), it was correctly recognized as الدّخانية with BHMMs of variable number of states (bottom). Note that there are two letters, 'ل' and 'د', that are written at the same vertical position (or column) and thus it is very difficult for our BHMMs to recognize them as two different letters. Anyhow, the incorrectly recognized word (top) is actually not very different in shape from the correct one; e.g. the characters 'ﻨ' and 'ﺘ' are very similar.

As indicated in the introduction, this work is largely motivated by the development of window repositioning techniques to deal with text distortions that are difficult to model with our windowed BHMMs. To test these techniques on IfN/ENIT, we used the best settings found above, that is, $W = 9$, $K = 32$ and BHMMs of variable number of states with $F = 0.4$. We compared the standard technique (no repositioning) with the three repositioning techniques introduced in this work: vertical, horizontal and both directions (see Sec. 5). Results are given in Table 1 for each of the four partitions considered above (abc-d, abd-c, acd-b, and bcd-a) and the partition abcd-e, which is also often used by other authors.

Table 1: WER% on five IfN/ENIT partitions of four repositioning techniques: none (no repositioning), vertical, horizontal and both. We used $W = 9$ and BHMMs of variable number of states ($F = 0.4$) and $K = 32$.

Training	Test	None	Vertical	Horizontal	Both
abc	d	7.5	4.7	8.4	4.8
abd	c	6.9	3.6	7.7	3.8
acd	b	7.7	4.5	8.1	4.4
bcd	a	7.6	4.4	8.2	4.6
abcd	**e**	**12.3**	**6.1**	**12.4**	**6.1**

From the figures in Table 1 it is clear that vertical window repositioning significantly improves the results obtained with the standard method or horizontal repositioning alone. To our knowledge, the result obtained for the abcd-e partition with vertical (or both) repositioning, **6.1**%, is the best result reported on this partition to date. Indeed, it represents a 50% relative error reduction with respect to the 12.3% of WER obtained without repositioning which, to our knowledge, was the best result until now. As said in the introduction, our windowed BHMM system with vertical repositioning ranked first at the ICFHR 2010 Arabic Handwriting Recognition Competition. In Table 2 we provide the best results on the test sets f and s (only known by the organization) from the last four competition editions (Märgner and El Abed, 2011).

Table 2: Best results from last four editions of the Arabic Handwriting Recognition Competition. Systems are based on HMM, NN (Neural Networks) or a combination of both.

System	Technology	Conference	ACC%	
			set *f*	set *s*
Siemens	HMM	ICDAR 2007	87.22	73.94
MDLSTM	NN	ICDAR 2009	**93.37**	81.06
UPV PRHLT (**This work**)	HMM	ICFHR 2010	92.20	**84.62**
RWTH-OCR	HMM+NN	ICDAR 2011	92.20	84.55

6.2. IAM Words

The IAM database comprises forms of unconstrained handwritten English text drawn from the LOB corpus and written by a total of 657 writers. This dataset was semi-automatically annotated to isolate text line images and individual handwritten words in them, from which two main versions of the dataset were built: IAM words and IAM lines. For the results reported below, we have used IAM words on the basis of a standard protocol for IAM lines, which is a writer independent protocol comprising 6 161 lines for training, 920 for validation and 2 781 for testing. Only words annotated as correctly segmented were used, which resulted in 46 956 words for training, 7 358 for validation and 19 907 words for testing. We used a closed vocabulary of 10 208 words for recognition, that is, the vocabulary of all words occurring in the training, validation and test sets. Class priors were computed as a smoothed unigram language model.

A first series of experiments was conducted on the training and validation data so as to determine appropriate preprocessing and feature extraction options. We tested different preprocessing alternatives, from no preprocessing at all to a full preprocessing method consisting of three conventional steps: gray level normalization, deslanting, and size normalization (Pastor, 2007). It is worth noting that, in this context, size normalization refers to a procedure for vertical size normalization of three different areas in the text line image (ascenders, text body and descenders), which of course might not be correctly located in all cases. On the other hand, feature extraction comprised three steps: rescaling of the preprocessed image to a given height D, binarization by Otsu's method, and final feature extraction by application of a window of a given width W and a particular repositioning technique. We tested different values of D (30 and 40) and W (9 and 11), and also each of the four repositioning techniques discussed above.

The best results in our first series of experiments were obtained with a two-step preprocessing including gray level normalization and deslanting, followed by feature extraction with $D = 40$, $W = 9$ and vertical repositioning. Using these settings, a second series of experiments was conducted on the training and validation data in which we tested different values for the number of states Q (4, 6, 8, 10 and 12) and the number of mixture components per state K (1, 4, 16 and 64). BHMMs were trained as described in Sec. 5 for the IfN/ENIT database. The results are shown in Fig. 6. Note that our best result in it, 24.8%, was obtained with $K = 64$ and $Q = 8$.

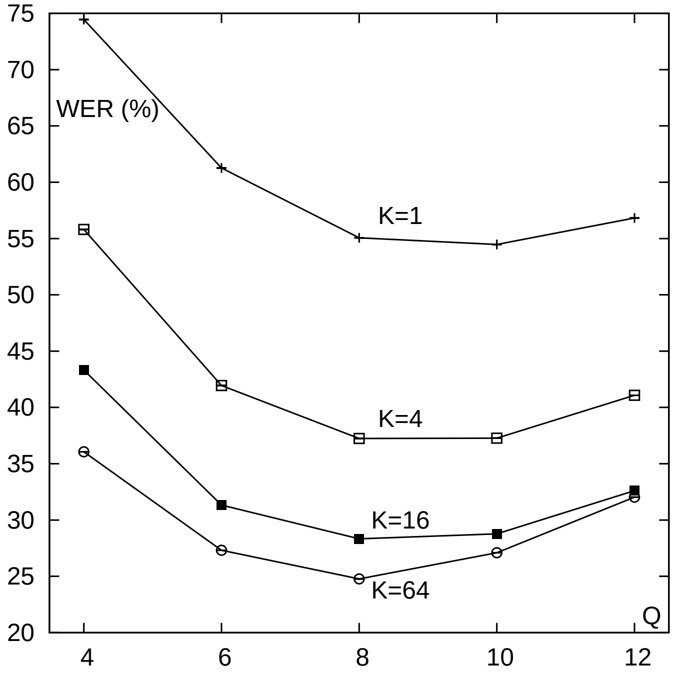


Figure 6: WER(%) on IAM words as a function of the number of states (Q) for several number of mixture components (K).

As usual in recognition of handwritten text lines, we may fine-tune system performance by adequately weighting the importance of class priors with

Table 3: Test-set WER on IAM words obtained with BHMMs and other techniques reported in (Bianne-Bernard et al., 2011).

System	WER %
BHMM (this work)	25.8
Context-independent HMM (CI)	35.4
Context-dependent HMM (CD)	32.7
Combination (CI+CD+Hybrid)	21.9

respect to class-conditional likelihoods. This is done by introducing a *grammar scale factor* G to scale class priors. We tested several values of G on the validation set using a system trained in accordance with the best results obtained in the previous series of experiments. A WER of 22.4% was achieved with $G = 90$.

In our final experiment on the IAM words dataset, we trained a system on the training and validation sets, using the best settings found above for preprocessing, feature extraction and recognition. It achieved a WER of 25.8% on the test set, which is quite good in comparison with other recent results on IAM words using the protocol described here (Bianne-Bernard et al., 2011). In particular, as it can be seen in Table 3, BHMMs are much better than the two systems based on HMM technology alone, though the combination of these two systems with a third, hybrid system (combining HMMs and Neural Networks) achieves even better results.

It must be noted that we had already tested conventional BHMMs (with one-column windows and no repositioning) on IAM words (Giménez and Juan, 2009), but we used the experimental protocol followed by Günter and

Bunke (2004), which is quite different to that used by Bianne-Bernard et al. (2011) and also here. Although the results are not directly comparable, our previous result with BHMMs, 29.6%, was not as good as the 25.8% of WER reported here.

6.3. RIMES

The *Reconnaissance et Indexation de données Manuscrites et de fac-similÉS* (RIMES) database of handwritten French letters was designed to evaluate automatic recognition and indexing systems of handwritten letters. Also, it has been used in several international competitions on handwritten words and line recognition (Grosicki and El Abed, 2009, 2011). For the experiments reported below, we have adopted the WR2 protocol used in the handwritten word recognition competition held at ICDAR 2009. It comprises 44 196 samples for training, 7 542 for validation and 7 464 for testing. The lexicon to be used during recognition is that of the set to be recognized (1 636 words for validation and 1 612 for testing), and the alphabet consists of 81 characters. As above, class priors were computed as a smoothed unigram language model.

As with IAM words, a first series of experiments was conducted on the training and validation data to decide on adequate options and parameter values for preprocessing, feature extraction and recognition. In particular, we tried three preprocessing alternatives, two repositioning techniques and different number of states ($Q = 4$, 6, 8, 10) and mixture components ($K = 1$, 4, 16 and 64). Other parameter values used were $D = 30$ and $W = 9$. The best WER, 21.7%, was obtained with a two-step preprocessing including deslanting and size normalization, followed by feature extraction with

$D = 30$, $W = 9$ and vertical repositioning; and then BHMM trained with $Q = 8$ and $K = 64$. Also as with IAM words, the performance of this system was fine-tuned by trying several values of the grammar scale factor G on the validation data. We achieved a WER of 18.7% with $G = 120$.

The best options and parameter values found on the validation set were used to train a system from the training and validation data, which was finally evaluated on the test set. We obtained a WER of 16.8%. In Table 4, this result is compared with those reported at the ICDAR 2009 competition (using the WR2 protocol) (Grosicki and El Abed, 2009). From these results, it becomes clear that our windowed BHMM system with vertical repositioning achieves comparatively good results.

Table 4: Test-set WER on RIMES obtained with BHMMs and different systems participating at the ICDAR 2009 competition (using the WR2 protocol). NN and MRF refer, respectively, to Neural Networks and Markov Random Fields.

System	Technology	WER %
TUM	NN	6.8
UPV	NN+HMM	13.9
BHMM (this work)	HMM	16.8
SIEMENS	HMM	18.7
ParisTech (1)	NN+HMM	19.8
IRISA	HMM	20.4
LITIS	HMM	25.9
ParisTech (2)	HMM	27.6
ParisTech (3)	HMM	36.2
ITESOFT	MRF+HMM	40.6

7. Concluding Remarks

Windowed Bernoulli mixture HMMs (BHMMs) for handwriting word recognition have been described and improved by the introduction of window *repositioning* techniques. In particular, we have considered three techniques of window *repositioning* after window extraction: *vertical, horizontal,* and *both.* They only differ in the way in which extracted windows are shifted to align mass and window centers (only in the vertical direction, horizontally or in both directions). In this work, these repositioning techniques have been carefully described and extensively tested on three well-known databases for off-line handwriting recognition. In all cases, the best results were obtained with vertical repositioning. We have reported state-of-the-art results in the IfN/ENIT database, and also good results on IAM words and RIMES.

Our current work is focused on the application of BHMMs to handwritten text line images and the use of different training techniques. We are also studying the application of repositioning techniques to other models, particularly conventional (Gaussian) HMMs. In the mid-term, we plan to try systems combining our BHMM technology with other technologies such as Neural Networks.

Acknowledgment

Work supported by the EC (FEDER/FSE), the Spanish MICINN (MIPRCV "Consolider Ingenio 2010", iTrans2 TIN2009-14511, MITTRAL TIN2009-14633-C03-01, erudito.com TSI-020110-2009-439, and a AECID 2010/11 grant).

References

Bianne-Bernard, A.L., Menasri, F., Al-Hajj Mohamad, R., Mokbel, C., Kermorvant, C., Likforman-Sulem, L., 2011. Dynamic and Contextual Information in HMM Modeling for Handwritten Word Recognition. IEEE Transactions on Pattern Analysis and Machine Intelligence 33, 2066–2080.

Dehghan, M., Faez, K., Ahmadi, M., Shridhar, M., 2001. Handwritten Farsi (Arabic) word recognition: a holistic approach using discrete HMM. Pattern Recognition 34, 1057–1065.

Dreuw, P., Heigold, G., Ney, H., 2009. Confidence-Based Discriminative Training for Model Adaptation in Offline Arabic Handwriting Recognition, in: ICDAR '09, Barcelona (Spain). pp. 596–600.

Giménez, A., Juan, A., 2009. Embedded Bernoulli Mixture HMMs for Handwritten Word Recognition, in: ICDAR '09, Barcelona (Spain). pp. 896–900.

Giménez, A., Khoury, I., Juan, A., 2010. Windowed Bernoulli Mixture HMMs for Arabic Handwritten Word Recognition, in: ICFHR' 10, Kolkata (India). pp. 533–538.

Grosicki, E., Carré, M., Brodin, J.M., Geoffrois, E., 2009. Results of the RIMES Evaluation Campaign for Handwritten Mail Processing, in: ICDAR '09, Barcelona (Spain). pp. 941 –945.

Grosicki, E., El Abed, H., 2009. ICDAR 2009 Handwriting Recognition Competition, in: ICDAR '09, Barcelona (Spain). pp. 1398 – 1402.

Grosicki, E., El Abed, H., 2011. ICDAR 2011 - French Handwriting Recognition Competition, in: ICDAR '11, Beijing (China). pp. 1459 – 1463.

Günter, S., Bunke, H., 2004. HMM-based handwritten word recognition: on the optimization of the number of states, training iterations and Gaussian components. Pattern Recognition 37, 2069–2079.

Khoury, I., Giménez, A., Juan, A., 2010. Arabic Handwritten Word Recognition Using Bernoulli Mixture HMMs, in: PICCIT '10, Hebron (Palestine).

Märgner, V., El Abed, H., 2007. ICDAR 2007 - Arabic Handwriting Recognition Competition, in: ICDAR '07, Curitiba (Brazil). pp. 1274–1278.

Märgner, V., El Abed, H., 2009. ICDAR 2009 Arabic Handwriting Recognition Competition, in: ICDAR '09, Barcelona (Spain). pp. 1383–1387.

Märgner, V., El Abed, H., 2010. ICFHR 2010 - Arabic Handwriting Recognition Competition, in: ICFHR '10, Kolkata (India). pp. 709–714.

Märgner, V., El Abed, H., 2011. ICDAR 2011 - Arabic Handwriting Recognition Competition, in: ICDAR '11, Beijing (China). pp. 1444 – 1448.

Marti, U.V., Bunke, H., 2002. The IAM-database: an English sentence database for offline handwriting recognition. IJDAR 5, 39–46.

Pastor, M., 2007. Aportaciones al reconocimiento automático de texto manuscrito. Ph.D. thesis. Dep. de Sistemes Informàtics i Computació. València, Spain.

Pechwitz, M., Maddouri, S.S., Märgner, V., Ellouze, N., Amiri, H., 2002. IFN/ENIT - DATABASE OF HANDWRITTEN ARABIC WORDS, in: CIFED '02, Hammamet (Tunis). pp. 21–23.

Rabiner, L., Juang, B.H., 1993. Fundamentals of speech recognition. Prentice-Hall.

Young, S., et al., 1995. The HTK Book. Cambridge University Engineering Department.