A Multi-Stream HMM-based Approach for Off-line Multi-Script Handwritten Word Recognition

Yousri KESSENTINI
LITIS-MIRACL Laboratories
University of Rouen, France
University of Sfax, Tunisia
yousri.kessentini@univ-rouen.fr

Thierry PAQUET
LITIS Laboratory,
University of Rouen, France
thierry.paquet@univ-rouen.fr

AbdelMajid BENHAMADOU
MIRACL Laboratory,
University of Sfax, Tunisia
abdelmajid.benhamadou@
isimsf.rnu.tn

Abstract

In this paper we present a unified approach for multi-script handwritten word recognition. The proposed approach is based on multi-stream HMM to combine 2 low level feature streams: density based features extracted from 2 different sliding window widths, and contours based features extracted from upper and lower contours. The multi-stream paradigm provides an interesting framework for the integration of multiple sources of information. Significant experiments have been carried out on two publicly available word databases: IFN/ENIT benchmark database (Arabic script) and IRONOFF database (Latin script). The proposed framework improves the recognition performance.

Keywords: handwriting recognition, HMM, multi-script, multi-stream.

1. Introduction

In the last years the field of handwriting recognition has been the topic of intensive research. The main development of the field took place in the last decade and some commercial products, based on Off-Line Cursive Word Recognition (CWR), are yet running in real world applications [3, 4]. However, this success is limited to some domains like postal address reading and bank check legal amount recognition where numerical information (zip code and digits amount) plays an important role to improve the recognition task. Many issues are then still open and the general problem of CWR is still far from being solved.

In today's business world Latin script is mostly used. But with the increasing communication between the different world communities more scripts are getting integrated into information systems.

In recent years various handwriting recognition methods have been developed for the recognition of different scripts (Latin, Arabic, Indian, Chinese …). In the case of multi-lingual documents, where handwritten scripts are present in several different languages on the same document, the combination of multiple CWR systems has been envisaged. Generally, document content are first classified according to the type of script before a specific script dependant CWR system is applied to recognize textual information [1][2]. Following this framework, multi-script recognition systems cumulate the complexity of each individual recognition system.

Some other approaches propose a unified recognition framework able to deal with multi-script handwriting. Such systems don't use any script specific methodology and therefore appear to be more general and applicable to a wider range of multi-lingual applications. However, few works have investigated this strategy until now. In [5], the authors propose a handwriting recognition system of various scripts such as Latin, Devanagari, and Kanji. In [6], the authors propose an OCR system to read two Indian language scripts: Bangla and Devanagari.

To the best of our knowledge, no multi-script recognition system has been proposed for Arabic and Latin scripts. In this context, we propose a unified approach for multi-script handwritten word recognition. As we want the approach to be script independent, the system must proceed without explicit segmentation of handwriting into graphemes. This is because explicit segmentation methods generally rely on script specific rules to find segmentation points. According to the proposed strategy it is therefore mandatory that the system can operate on low level frame features such as directional or colour densities. In order to overcome the poor discriminative power of such low level features the proposed approach is based on the multi-stream paradigm that provides a way of combining individual feature streams in order to obtain an overall hypothesis. Such approach has been particularly studied in the domain of Automatic Speech Recognition (ASR) and presents multiple advantages [11][12].

The paper is organised as follows. Section 1 reviews the multi-stream formalism. In section 2, the overall system organization is presented. In section 3, feature extraction and modelling techniques are illustrated. Experimental results are given in section 4, they show that the system give interesting results both on Latin and Arabic scripts. Conclusion and future work are addressed in section 5.

2. Multi-stream statistical model

The multi-stream formalism is an adaptive method to combine several individual feature streams using cooperative Markov models.

This problem can be formulated as follows: assume an observation sequence X composed of K input streams

X_k representing the utterance to be recognized, and assume that the hypothesized model M for an utterance is composed of J sub-unit models $M_j(j=1,\ldots,J)$ associated with the subunit level at which we want to perform the recombination of the input streams (e.g., characters). To process each stream independently of each other up to the defined sub-unit level, each sub-unit model M_j is composed of k models M_j^k (possibly with different topologies). Recombination of the k stream models M_j^k is forced at some temporal anchor states. The resulting statistical model is illustrated on Figure 1. A detailed discussion of the mathematical formalism is given in our previous work [8].

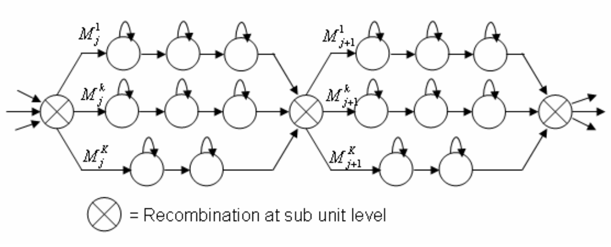

Figure 1: General form of k-stream model with anchor points between sub-units models

As discussed in [8], the multi-stream training and recognition problems can be addressed using various statistical formalisms. In our case, stream models are trained separately as described below. During recognition the best word model M maximising the likelihood p(X|M) is search for. Multi-stream HMM introduces some recombination states between subunits. Two solutions have been investigated in the literature:

- Recombination at the HMM state level: Although it does not allow asynchrony or different topologies of the different stream models, it amounts to performing a standard Viterbi decoding in which local probabilities are obtained from a linear or nonlinear combination of the local stream probabilities.
- Recombination at the sub-unit level: it can force the streams to be synchronous where synchrony is required (end of sub-units) and can allow for asynchrony where asynchrony might take place. It requires a more sophisticated decoding procedure than the Viterbi search. Two different algorithms have been applied to solve the problem of decoding in this case: Two level dynamic programming [9] and HMM-decomposition (or recombination) [11] that will be detailed below.

Multi-stream decoding – The Product HMM

The principle of the multi-stream HMM is to model independently each stream between two pre-determined synchronization points, using a number (here, two) of single-stream HMMs. In this study, the synchronization states are taken to be the character boundaries (see Figure 2). Decoding based on this integration method requires to individually compute the best state sequence for both streams. To avoid the computation of two best state paths, the model can be formulated as a composite or product HMM (see Figure 3). Decoding under such a model requires computing a single best path using the well known Viterbi decoding algorithm.

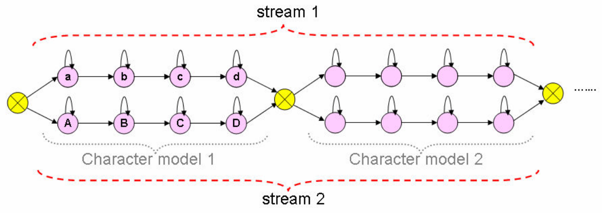

Figure 2 : Example of a multi-stream HMM with 2 streams and 4 states in each character model.

In this model, the observation log-likelihood conditioned on composite state a-A, that has been composed from the stream states a and A, respectively, is given using a weighted sum of log-likelihood combination function by:

$$\log P(X_1(t),X_2(t)/a\text{-}A)=\alpha_i^1 \log P(X_i^1(t)/a)+\alpha_i^2 \log P(X_i^2(t)/A)$$

where X1(t) (eg. X2(t)) is the observation vector corresponding to stream 1(eg. stream 2). The transition probabilities of the single-stream HMMs are shared by several transition probabilities in the composite model.

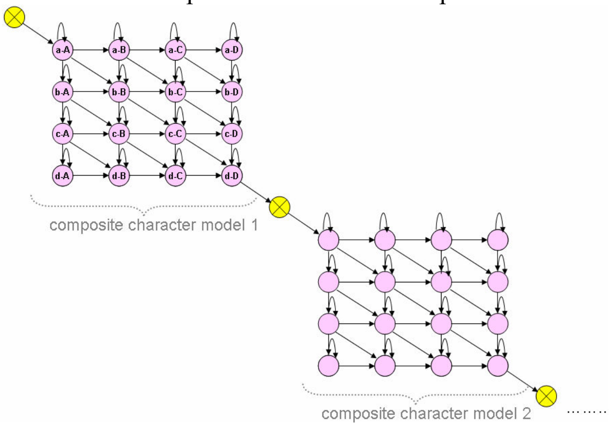

Figure 3 : The corresponding product (composite) HMM.

3. A multi-script recognition system

The proposed system is based on a multi-stream Hidden Markov Model (HMM) for the recognition of two scripts handwritten words (Arabic and Latin scripts).

HMMs have been successfully applied to handwriting recognition. They offer several advantages allowing us to propose a unified multi-script recognition system. These possibilities are mainly the automatic training of character models on non-segmented words (embedded training), and the segmentation-free recognition paradigm that fits particularly well to a multi-script approach.

In the first step, a set of pre-processing is applied to the word image. Two feature sets have been tested in this work: contours based feature are extracted respectively from the lower and the upper contours, and density based feature are calculated on two different sliding windows each with a particular width (see Figure 5). For each feature type two feature streams representing the input word image are computed. Each stream model is then separately trained using Baum Welch algorithm, combination weights of the multi-stream model are then optimized using relative frequency strategy. The last step is recognition during which the two HMM models are simultaneous decoded according to the multi-stream formalism presented above (see Figure 4).

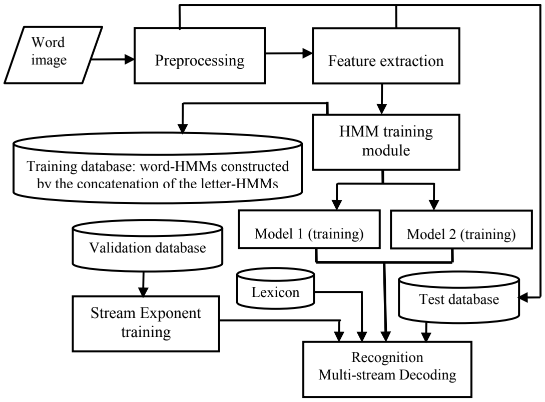

Figure 4. Methodology for the 2-stream training and decoding

3.1. Preprocessing and Features extractions

A set of preprocessing is applied to word images in order to eliminate the noise and to simplify the procedure of feature extraction. It is worth noticing that these preprocessing are script independent.

• Normalization: In an ideal model of handwriting, a word is supposed to be written horizontally and with ascenders and descenders aligned along the vertical direction. In real data, such conditions are rarely respected. We use slant and slope correction to normalize the word image [15].

• Contour smoothing: Smoothing eliminates small blobs on the contour.

• Base line detection: Our approach uses the algorithm described in [10] based on the horizontal projection curve that is computed with respect to the horizontal pixel density (see Figure 5). Baseline position is used to extract baseline dependent features which emphasize the presence of descenders and ascenders.

In order to build the feature vector sequence, the image is divided into vertical overlapping windows or frames. The sliding window is shifted along the word image from right to left (case of Arabic words) or left to right (case of Latin words) and a feature vector is calculated for each frame.

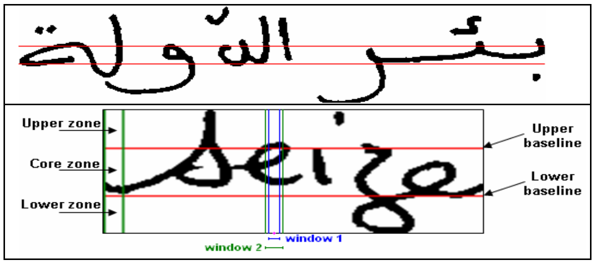

Figure 5. Upper and lower baselines detection

Tow feature sets are tested in this work. The first one is based on contour feature and has been already described in our previous work [8]. The second one is based on foreground (black) pixel densities and is detailed in [7].

Density features

On each frame 26 features are extracted (with a window width of 8 pixels). There are two types of features: distribution features based on foreground (black) pixel densities, and concavity features. In order to compute some of these features (f2, f15) the window is divided into cells where the cell height is fixed (4 pixels in our experiments) see Figure 6.For each frame, the features are the following:

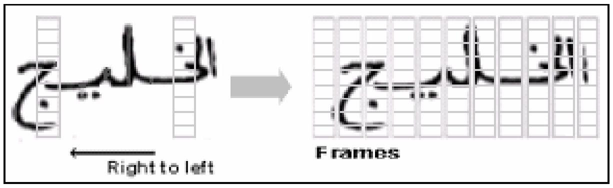

Figure 6. Word image divided into vertical frames (here without overlap) from [7]

- f1: density of foreground (black) pixels.
- f2: number of transitions between two consecutive cells of different density levels.
- f3: difference in y position of gravity centers of foreground pixels in the current frame and in the previous one.
- f4 to f11: densities of black pixels for each vertical column of pixels in each frame (if the width of the frame is 8 pixels).
- f12: normalized vertical position of the center of gravity of the foreground pixels in the whole frame with respect to the lower baseline.
- f13-f14: density of foreground pixels over and under the lower baselines for each frame.
- f15: number of transitions between two consecutive cells of different density levels above the lower baseline.
- f16: zone to which the gravity center of black pixels belongs with respect to the upper and lower baselines (above upper baseline, a middle zone, and below lower baseline).
- f17 to f26: concavity features in each frame and the concavities in the core zone of a word, that is, the zone bounded by the two upper and lower baselines (see Figure 5).

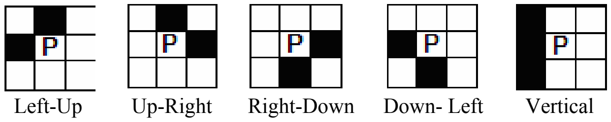

Figure 7. Five types of concavity configurations for a background pixel P

The density feature set has been chosen in order to capture the presence of ascenders, descenders and dots whatever their exact position in the word image. Concavity features are added to reflect local concavity and stroke directions.

Contour Feature

These features are extracted from the word contour representation. Each word image is represented by its lower and upper contours (see Figure 8). A sliding window is shifted along the word image, two parameters characterize a window: window width (8 pixels) and window overlap between two successive positions (5 pixels). For each position of a window, we extract the upper contour points (resp. the lower contour points). For every point in this window, we determine the corresponding Freeman direction and the directions points are accumulated in the direction histogram (8 features).

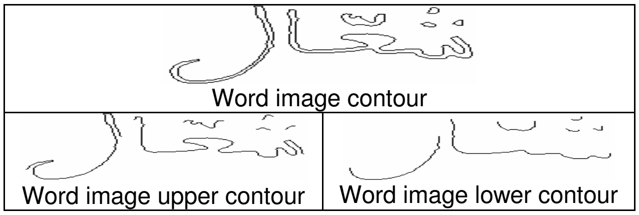

Figure 8. Word image contours

The second feature set is computed by determining for every point of the upper contour (resp. lower contour) the nature of the vertically face to face contour corresponding point. Depending on the properties of the reached point, 4 situations can be distinguished. The point can belong to a:

- Lower contour (resp. upper contour) (see red points on Figure 9).
- Interior contour on closure (see blue points on Figure 9).
- Upper contour (resp. lower contour) (see yellow points on Figure 9).
- No point found (see green points on Figure 9).

The black points in Figure 6 represent the lower contour.

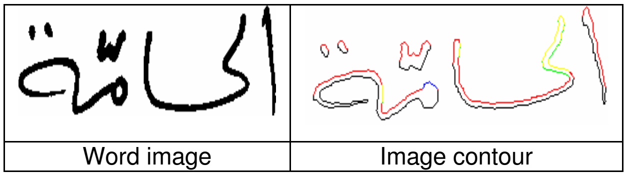

Figure 9: Contour feature extraction

The histogram of the four kinds of points is calculated in each window. This second features set provides additional information about the loops, the turning points ('t' for example), the simple lines and the end points on the word image (4 features).

The third feature set indicates the position of the upper contour (resp. lower contour) points in the window. For this purpose, we localize the core zone of the word image. More precisely, we extract the lower and upper baselines of word images. These baselines divide the image into 3 zones: 1) a middle zone, 2) the lower zone, 3) the upper zone.

The third feature set (3 features) provide as supplementary information's about ascending and descending characters, which are salient characteristics for recognition in Latin script, as well as in Arabic script.

3.2. Parameter estimation

In order to model the Latin characters, we built up 26 character models. In the case of Arabic characters, we built up to 159 character models. In fact, an Arabic character may have different shapes according to its position in the word (beginning, middle, end word position). Other models are specified with additional marks such as “shadda”. In both Latin and Arabic script, each character model is composed of 4 emitting states. The observations probabilities are modelled with Gaussian Mixtures (3 per state).

Training the multi-stream HMM consists in two tasks: First, estimation of its stream component parameters (mixture weights, means, variances, and state transition probabilities) and estimation of appropriate stream exponents. Maximum likelihood parameter estimation by means of the EM algorithm can be used in a straightforward manner to train the first set of parameters. This can be done in two ways: Either train each stream component parameter set separately, based on single-stream observations, and subsequently combines the resulting single-stream HMMs, or train the entire parameter set (excluding the exponents) at once using the bimodal observations.

In our case, the two stream models are trained separately. Embedded training is used where all character models are trained in parallel using Baum-Welch algorithm applied on word examples. The system builds a word HMM by concatenation of the character HMM corresponding to the word transcription of the training utterance. The final training step concerns the optimization of the stream weights. It is performed using two different strategies.

- **Equal combination weights:** This strategy consists in attributing equal weights for the various streams. Its main advantage is that no data for estimation of the weights is needed and no extra time for the calculation of the weights has to be expended.
- **Relative frequency weights:** Employing the forced segmentation given by the multi-stream decoding algorithm, we compute the ratio between the number of times an expert (ie. A stream) performs best for a given

character, and the number of times this character occurs in the database.

$$\alpha_j^k = \frac{n_{k,j}}{n_j}$$

where $n_{k,j}$ is the number of training frames for which expert k has the largest probability, over all experts, for character j, and n_j is the number of frames for character j in the training data. During our experiments we noticed that the 2 weighting strategies perform similarly.

4. Experiments and Results

To evaluate the performance of our recognition system, experiments have been conducted on two publicly available databases: IFN/ENIT benchmark database of Arabic words and IRONOFF database for Latin words (French and English). In all experiments, recognition scores at the word level have been evaluated using the percentage of samples which are correctly classified.

4.1. IFN/ENIT database

The IFN/ENIT [13] contains a total of 26459 handwritten words of 937 Tunisian town/villages names written by different writers. Some town/village names occur in the database with slightly different writing. From this it follows that our lexicon consists of about 2100 valid entries. We report recognition rates on word level. Four distinct sets (a, b, c, d) are predefined in the database (3 sets for training and 1 set for testing).

The recognition results are given for different lexicon sizes. The lexicon generation is achieved by random selection of N-1 words among a lexicon of 2100 words (complete lexicon of the annotated database) and adding the word transcription of the image to be recognized.

Table1 shows the experimental results of the performance evaluation of our recognition system using contour feature sets. Results using the upper contour model, the lower contour model and the multi-stream model are given.

Table 1. IFN/ENIT recognition performances using contour features

Models	*Lexicon size*		
	10	*100*	*500*
Upper contour	99.4	94.6	81.8
Lower contour	98.2	92.6	79.4
Multi-stream	**99.8**	**96**	**86.2**

Table 2. IFN/ENIT recognition performances using density features

Models	*Lexicon size*		
	10	*100*	*500*
Window *1*	93.8	85.8	73.8
Window *2*	96.4	89.4	77.4
Multi-stream	**98.2**	**93.8**	**81.2**

In table 2, we present the recognition results using the density feature extracted from 2 window widths. The first window width is fixed to 8 pixels (Window 1). The second is determined by widening the first window by 3 pixels on left and right sides (Window 2).

In both experiments, we notice that the multi-stream combination approach improves the recognition rate (for both feature sets). In [8], the experiments have shown that the multi-stream approach performs better than the other standard combination strategies of each individual stream model namely fusion of representations and fusion of decisions. [16] presents the results of the second competition of Arabic handwritten word recognition systems using the IFN/ENIT database. 14 systems have participated in the competition. The systems were tested on known and unknown data. Tests with unknown data show three systems with a recognition rate less than 20%. The others are between 59.01% and 87.22%. Although we don't use the same data for testing, we notice that our performance system is reasonable compared to the competition performance systems. We can also compare our result to [7], they achieved a recognition rate of 86.51% by using a lexicon of size 450. We obtain almost the same performance with a lexicon size of 500 (86.2%).

4.2. IRONOFF database

The IRONOFF database contains a total of 36,396 isolated French word images from a 196-word lexicon [14]. Although the database contains both on-line and off-line information of the handwriting signals, only the off-line information is used for our experiments. The offline handwriting signals are sampled with spatial resolution of 300 dots per inch (DPI), with 8 bits per pixel (256 gray levels). The training data set contains 24,177 word images, and another 12,219 images are used as test data set. We name the full database as IRONOFF-196. A subset of the IRONOFF-196, which consists of only French cheque-word, is named IRONOFF-Cheque. IRONOFF-Cheque has only 30 word lexicons, and has 4481 test images.

Table 3. IRONOFF-196 recognition performances using contour features

Models	*Top*	*Lexicon size*			
		10	*100*	*196*	*500*
Upper contour	*1*	92.6	74.8	70	67
	5	99.4	92.6	87.2	86
Lower contour	*1*	89.8	77	69.2	61.4
	5	99.2	93.2	88.2	79
Multi-stream	*1*	**95.2**	**89**	**83.8**	**78.4**
	5	**99.8**	**95.6**	**94.6**	**91.6**

In the experiments, recognition scores at the word level have been computed by the percentage of samples which are correctly classified (Top 1). Generalizing this concept, we also compute the cumulated recognition rate of the first 5 positions in the candidate list (Top 5).

Table 3 and 4 shows the recognition results obtained using respectively contour and density features. The recognition results are given for different lexicon sizes. To further challenge the recognizer, we have also performed tests using a 500 word lexicon.

Table 4. IRONOFF-196 recognition performances using density features

Models	*Top*	*Lexicon size*			
		10	*100*	*196*	*500*
Window 1	*1*	75.6	51	45	37.6
	5	96.8	74.8	70.8	52.4
Window *2*	*1*	77.8	52.8	45.2	37.8
	5	97.2	75	70.2	53
Multi-stream	*1*	**95.6**	**84.2**	**79.6**	**76.2**
	5	**99.8**	**94.6**	**90.8**	**90**

Table 5 and 6 show the IRONOFF-Cheque performances. These results are reasonable compared to some performances in bank check reading given in [8].

Table 5. IRONOFF-Cheque performances using contour features

Models	*Top*	
	1	*5*
Upper contour	82.6	94.6
Lower contour	73.2	92.4
Multi-stream	**86**	**95.4**

Table 6. IRONOFF-Cheque performances using density features

Models	*Top*	
	1	*5*
Model 1	77.6	89.6
Model 2	76.6	88.8
Multi-stream	**90.60**	**98.8**

Performances on the IRONOFF-196 database are inferior compared to IRONOFF-Cheque set. In fact, IRONOFF-196 words lexicon contain a capital letters and a special characters like (^, ', `, -, à, ç…). In addition, many words like “nous” and “vous”, “donc” and “dont”, “tout” and “tous”, “cette” and “celle” increase the confusion.

5. Conclusion and Perspectives

This paper presents a multi-stream HMM-based approach for off-line multi-script handwritten word recognition. The proposed system is script independent: it proceeds without explicit segmentation of handwriting into graphemes. Low level feature sets have been extracted based on directional and colour densities. Features are then combined allowing to the multi-stream framework. The developed system has been experimented on two publicly available databases: the benchmark database IFN/ENIT (for Arabic script) and IRONOFF database (for Latin script). The results show significant improvement of the recognition rate coming from using a multi-stream approach. Future work consists to combine more than 2 streams and to study the complexity and the limits of this approach.

References

[1] G. Nagy, S. Seth, and X. Zhang, Multi-Character Field Recognition for Arabic and Chinese Handwriting, *Proceedings of the Summit on Arabic and Chinese Handwriting Recognition*, September 2006, College Park, MD, pp. 93-100.

[2] J.J. Lee, M. Nakajima and J. Kim, A Hierarchical HMM Network-based Approach for On-line Recognition of Multi-Lingual Cursive Handwritings, *IEICE Transactions on Information and Systems*, 1998, vol. E81-D, No. 8, pp. 881-888.

[3] D. D’Amato, E. Kuebert, A. Lawson, Results from a performance evaluation of handwritten address recognition systems for the United States Postal Service, *in: IWFHR*, Amsterdam, 2000, pp.189–198.

[4] N. Gorski, V. Anisimov, E. Augustin, O. Baret, D. Price, J. Simon, A2iA check reader: a family of bank check recognition systems, *in Proceedings of ICDAR*, Vol. 1, Bangalore, 1999, pp. 523–526.

[5] A. Malaviya, C. Leja, L. Peters, Multi-script handwriting recognition with FOHDEL. *New Frontiers in Fuzzy Logic and Soft Computing Biennial Conference of the North American Fuzzy Information Processing Society-NAFIPS*. IEEE, Piscataway, NJ, USA, 1996, pp. 147-151

[6] B.B. Chaudhuri, U. Pal, An OCR system to read two Indian language scripts: Bangla and Devnagari (Hindi), *ICDAR '97,* vol 2, 1997 , pp. 1011 -1015.

[7] R. El-Hajj, L. Likforman-Sulem, C. Mokbel, Arabic handwriting recognition using baseline dependent features and Hidden Markov Modeling, *ICDAR 2005*, Seoul, Corée du Sud.

[8] Y. Kessentini, T. Paquet, A. BenHamadou. Combinaison d’Information pour la Reconnaissance de l’Ecriture Manuscrite Hors-Ligne. *RFIA’2008*, Amiens, France.

[9] H.Sakoe, Two-level DP matching - A dynamic programming-based pattern matching algorithm for connected word recognition, *IEEE Transactions of the IECE of Japan*, 1979, vol 27, pp 588- 595.

[10] A.Vinciarelli, S.Bengio, H.Bunke, Offline Recognition of Unconstrained Handwritten Texts Using HMMs and Statistical Language Models, *IEEE Transactions on PAMI*, 2004, vol. 26, No 6, pp. 709-720.

[11] H.Bourlard, S.Dupont. “Sub-band-based Speech Recognition”. *In IEEE Int. Conf. on Acoust., Speech, and Signal Processing*, 1997, pp. 1251-1254.

[12] H.Bourlard, S. Dupont, and C. Riss. *Multi-stream speech recognition. Technical Report IDIAP-RR 96-07*, 1996.

[13] Pechwitz M., Maddouri S., Maegner V., Ellouze N, "IFN/ENIT–DataBase for Handwritten Arabic words", *CIFED’02*, Hammamet, Tunisia, 2002, pp. 129-136.

[14] C. Viard-Gaudin, P.M. Lallican, S. Knerr, P. Binter, “The IRESTE On/Off (IRONOFF) Dual Handwriting Database”, *ICDAR’99.*

[15] F. Kimura, S. Tsuruoka, Y. Miyake & M. Shridhar, “A Lexicon Directed Algorithm for Recognition of Un-constrained Handwritten Words”. *IEICE Trans*. Inf. &Syst., vol. E77-D, no. 7, 1994.

[16] V. Märgner, H. El Abed: Arabic Handwriting Recognition Competition. *ICDAR 2007,* vol. 2 pp. 1274-1278