Effective Technique for the Recognition of Writer Independent Off-line Handwritten Arabic Words

Sherif Abdel Azeem and Hany Ahmed
Electronics Engineering Department
American University in Cairo (AUC)
Cairo, Egypt
shazeem@aucegypt.edu, hanyahmed@aucegypt.edu

Abstract

In this paper we present a novel segmentation-free Arabic handwriting recognition system based on hidden Markov model (HMM). Two main contributions are introduced: a novel pre-processing method and a new technique for dividing the image into non uniform horizontal segments to extract the features. The proposed system first pre-processes the input image by setting the thickness of the input word to three pixels and fixing the spacing between the different parts of the word. The input image is then divided into constant number of non uniform horizontal segments depending on the distribution of the foreground pixels. A set of robust features representing the foreground pixels is extracted using vertical sliding windows. The proposed system builds character HMM models and learns word HMM models using embedded training data. The performance of the proposed system is very promising compared with other Arabic handwriting recognition systems available in the literature.

1. Introduction

A handwriting recognition system can either be online or offline. The offline handwriting is based on Optical Character Recognition (OCR) and is usually applied on scanned documents. On the other hand, in online handwriting; the pressure is applied on a digital instrument and a sequence of points is traced out by the pen. Offline handwriting recognition involves the automatic conversion of text in an image into letter codes which are usable within computers and text-processing applications and it is generally observed to be harder than online handwriting recognition. In the online case, features can be extracted from both the pen trajectory and the resulting image, whereas in the offline case only the image is available.

In recent years, some research has been done on the problem of offline Arabic handwriting recognition [2, 3, 5, 10 and 12]. Despite this fact, offline Arabic handwriting recognition is still very challenging because of the varying writing style from person to person, difficulty of segmentation because of the cursive nature of the Arabic writing. Moreover, the Arabic alphabet contains 28 letters, each has between two and four shapes and the choice of which shape to use depends on the position of the letter within its word or sub-word. Also, there is the problem of diacritical points which are known as delayed strokes that can change the meaning of the word or sub-word. More details about Arabic writing characteristics can be found in [23].

Due to the advantages of hidden Markov models, many researchers have used them for Arabic handwriting recognition [1, 2, 3, and 5]. Since these are stochastic models; they can cope with noise and variations in the handwriting, also the observation sequence that corresponds to features of an input word can be of variable length, and most importantly, word HMMs can solve the problem of segmentation implicitly.

In this paper we introduce a new approach for pre-processing the input image and a new method of extracting features by dividing the input image horizontally with non-uniform heights. Our target is to study the recognition of offline handwritten Arabic words of the IFN/ENIT database [21] using HMM theory.

This paper is organized as follows: Section 2 describes the pre-processing stage. In section 3, we present the proposed recognition system with the use of HMM. In section 4, the word modeling is presented followed by the feature extraction algorithm within each window. Section 6 reports the experiments carried out on the IFN/ENIT database of handwritten city names. In Section 7, some conclusions and perspectives are discussed.

978-0-7695-4774-9/12 $26.00 © 2012 IEEE
DOI 10.1109/ICFHR.2012.200

2. Pre-Processing

The pre-processing step is intended to increase the ease of extracting features and decrease the noise in the image and decrease the variability of writing styles from one person to another. The pre-processing step involves changing the thickness of the word to a fixed number of pixels and reducing the distance between parts of word (POWs) if it is greater than a pre-determined threshold.

2.1 Changing the thickness of the word to a pre-determined number of pixels:

We detected the fact that there are differences in the thickness of the words in the database. It has been discovered after several experiments that the rate of recognition increases after normalizing the thickness of the input image. Normalizing the thickness of a word in the input image goes through two stages:

1) Thin the input word [25].

As shown in Fig. 1.b, the original word is thinner than another one for the same word shown in Fig. 1.a. After thinning, the thickness of each word becomes the same (1 pixel) as shown in Fig. 1.c.

2) Dilate the binary image with 3-pixels surrounding the original thinned pixel, as shown in Fig. 1.d. Thus, the thicknesses of the two words become the same.

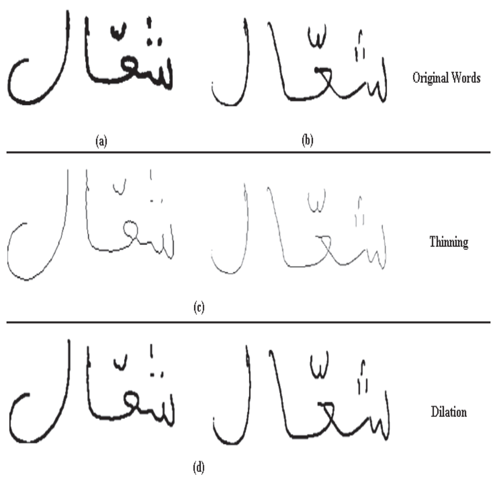

Figure 1. The steps used in changing the thickness of the input word, (a,b) Same original word whose thickness is different in the two figures ,(c) After thinning ,(d) After fixing the thickness.

2.2 Reducing the distance between parts of words (POWs):

Vertical projection of the input image has been used to find the places where there are no pixels, and then the width (d) of each place is calculated, as shown in Fig. 2.a. If the width is greater than a pre-determined threshold (6 pixels, empirically), then (d - threshold) will be removed from this place to guarantee fixed separation between POWs as shown in Fig. 2.b.

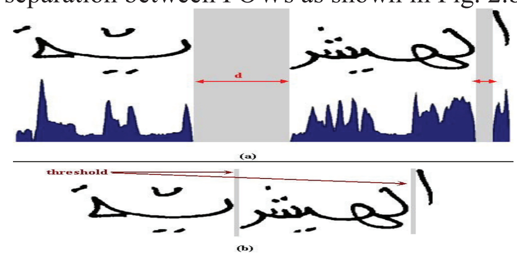

Figure 2. (a) Original word, (b) The word after removing the empty spaces.

In the results section, we show that the pre-processing stage is very important in enhancing the performance of the system. The results also show that without the pre-processing stage the recognition rate of the system drops by 2.6%.

3. Hidden Markov Model

The hidden Markov model is a double stochastic process, which can efficiently model the generation of sequential data. The HMM used in this paper is a continuous HMM with one HMM for each character. In this paper, we use the same HMM classifier without modification as implemented in HTK Speech Recognition Toolkit [11]. However, we implement our own parameters of the HMM. HTK models the feature vector with a mixture of Gaussians. It uses the Viterbi algorithm in the recognition phase, which searches for the most likely sequence of a character given the input feature vector.

The hidden Markov model is a finite set of states; transitions among the states are governed by a set of probabilities called transition probabilities. In a particular state, an observation can be generated according to the associated probability distribution. Given an observation sequence, an HMM process deals with all feature observations in all states, and then emits probability distribution indicating the confidence of the object represented by the model. Based on the extracted feature sequence from an unknown pattern, the classification is achieved by finding a model which generates a maximum probability.

Figure 3 displays the case of an eight-state HMM in total, showing that we allowed transition to the current and the next states only. Six of these are emitting states and have output probability distributions associated with them. The transition matrix for this model has 8 rows and 8 columns. Each row will sum to one except for the final row which is always all zero since no transitions are allowed out of the final state.

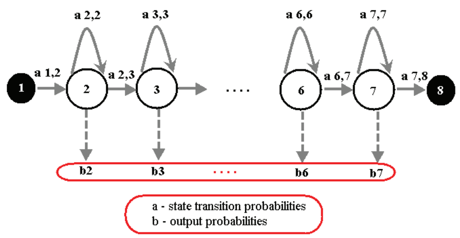

Figure 3. 8-stats left to right.

HTK is principally concerned with continuous density models in which each observation probability distribution is represented by a mixture Gaussian density. In this case, for state j the probability $b_j(o_t)$ of generating observation o_t is given by:

$$b_j(o_t) = \prod_{s=1}^{S} [\sum_{m=1}^{M_s} c_{jsm} \Psi(o_{st}; \mu_{jsm}, \Sigma_{jsm})]^{\gamma_s}$$

Where M_{js} is the number of mixture components in state j for stream s, The exponent γ_s is a stream weight and its default value is one, c_{jsm} is the weight of the m th component and ψ (o; μ,) is a multivariate Gaussian with mean vector μ and covariance matrix , that is

$$\Psi(o; \mu, \Sigma) = \frac{1}{\sqrt{(2\pi)^n \mid \Sigma \mid}} e^{-\frac{1}{2}(o-\mu)^T \Sigma^{-1}(o-\mu)}$$

Where n is the dimensionality of o.

64-Mixtures have been chosen after large number of experiments to give a robust model for each character and high recognition rate.

4. Word modeling and dictionary building

The character shape in Arabic is context sensitive, that is, depending on its position within a word (isolated or beginning or middle or end), also Arabic characters are rich in diacritic marks and delayed strokes (dots, Shadda, Hamza, etc.), as shown in Fig. 4.

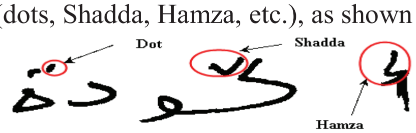

Figure 4. Figure shows diacritic marks and delayed strokes.

A dictionary of all the different unique words in the IFN/ENIT database along with their delayed strokes and diacritic marks has been constructed by tracing the ground truth of each image in the IFN/ENIT database. A total of 166 character models have been generated by taking into consideration the fact that an Arabic character may have different shapes according to its position in a word as found in the database. Other models are created to model characters with special marks such as shadda (ً).

5. Feature Extraction

The input image in binary format as given in the IFN/ENIT database is used to extract a feature vector after the pre-processing stage. The background pixels of the image are labeled by logic "0" and the foreground pixels by logic "1".

The input image is segmented into n horizontal segments with approximately equal number of foreground pixels in each segment. The total number of pixels in the image is counted and then divided by n which results in the required number of pixels per segment.

Figure 5 shows different cases of dividing the image into different number of segments.

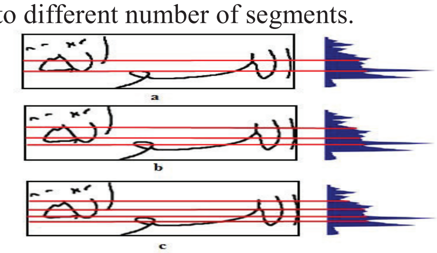

Figure 5. (a) Three horizontal segments , (b) Four horizontal segments ,(c) Five horizontal segments.

The input image is then divided into vertical overlapping frames. Each frame is divided into small cells with fixed width, while the cell's height depends on the word's image and the distribution of foreground pixels in the images. The sliding window is shifted along the word image from right to left and a feature vector is calculated for each frame, as shown in Fig. 6.

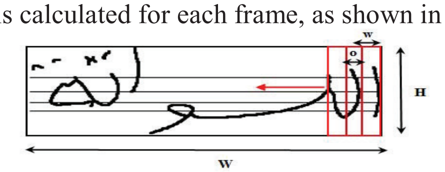

Figure 6. Feature extraction using a sliding window with overlapping.

Where 'H' is the height of the image, 'W' is the width of the image, 'w' is the width of the vertical frame, and 'o' is the width of overlapping of the vertical frames.

In our experiments we found that the best case is when the width of the vertical frame is six pixels with three pixels overlapping. Features are then extracted as follows.

5.1 F1- Number of foreground pixels:

We used five cells (empirically) per vertical frame with non uniform heights to extract this feature as shown in Fig. 7.

For each cell, the number of "1" pixels is calculated to produce five features per frame (5 cells per frame x 1 feature per cell).

Figure 7. Five features per frame of F1 by dividing the image into five non uniform heights.

5.2 F2- Gradient Features:

The gradient operator [15] is applied to the image to give two gradient components: strength and direction for all the points of the image. This is done by applying the Sobel operator [7] on the image to extract the vertical and horizontal gradient components. Only the direction is used in the computation of our feature vector, this direction can range from 0 to 360 degrees. This range is split into 8 non-overlapping regions of 45 degrees, as shown in Fig. 8.

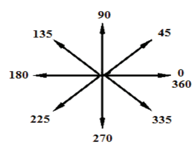

Figure 8. Eight non-overlapping regions of 45 degrees.

In each sampling region, a histogram of gradient directions is taken at each pixel of the region. Each histogram value corresponds to the count of each gradient direction in the region. To extract the gradient features we used three cells (empirically) with non uniform heights per frame as shown in Fig. 9. For each cell, eight gradient features produced to form 24 features per frame (8 features per cell x 3 non uniform cells per frame).

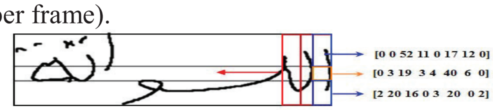

Figure 9. Twenty four features per frame of F2 by dividing the image into three non uniform heights.

Another eight features extracted from the whole frame representing the histogram of the eight gradient directions in the frame, as shown Fig. 10, are added to the previous features to form 32 features per frame.

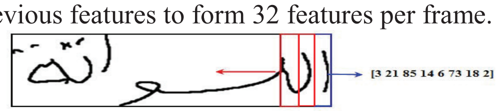

Figure 10. Eight features per frame of F2 .

The total number of features extracted per frame is 37 features (5 F1 + 32 F2).

Delta and acceleration coefficients [4] were calculated and then concatenated with the previous features.

6. Experiments and results

To evaluate the performance of the proposed recognition system, experiments have been conducted on the five publicly available subsets a, b, c, d and e of the benchmark IFN-ENIT database.

Table 1 describes the final results of the whole system and compares with other systems using sets a, b, c, d and e for training and two sets f and s for test under the same conditions of the last competitions on Arabic handwriting recognition ICDAR 2005, ICDAR 2007, ICDAR 2009 , IFCHR 2010 and 2011.

Table 1 – Recognition results in % of correctly recognized images on training datasets d and e and test datasets f and s.

System	d	e	f	s
Results of the 3 best systems at ICDAR 2011 [20]				
JU-OCR	75.49	63.75	63.86	49.75
RWTH-OCR	99.67	98.61	92.2	84.55
REGIM	94.12	86.62	79.03	68.44
Results of the 3 best systems at ICFHR 2010 [19]				
UPV PRHLT	99.38	98.03	92.2	84.62
CUBS-AMA	89.97	80.8	80.32	67.9
RWTH-OCR	99.66	98.84	90.94	80.29
Results of the 3 best systems at ICDAR 2009 [18]				
MDLSTM	99.94	99.44	93.37	81.06
Ai2A	97.02	91.68	89.42	76.66
RWTH-OCR	99.79	98.29	85.69	72.54
Results of the 3 best systems at ICDAR 2007 [17]				
Siemens	94.58	87.77	87.22	73.94
MIE	93.63	86.67	83.34	68.40
UOB-ENST	92.38	83.92	81.93	69.93
Proposed System	**99.26**	**96.7**	**91.98**	**82.52**

As shown in previous table, our system is one of top four systems out of 41 systems that have been tested in ICDAR 2007, ICDAR 2009, ICFHR 2010 and ICDAR 2011. This proves that the use of the proposed new technique of pre-processing and dividing the image into non uniform horizontal segments is very effective and very promising.

The results in Table 1 show that the proposed system comes in the fourth place when testing set *f* after the MDLSTM (2009) [18], UPV PRHLT (2010) [19], and RWTH-OCR (2011) [20] systems and comes in the third place when testing set *s* after the UPV PRHLT(2010) and the RWTH-OCR (2011) systems.

In ICDAR 2005 Arabic handwriting recognition competition [16], the sets a, b, c and d were used for training while set e was used for testing .The results obtained by the proposed system clearly outperform those presented in the ICDAR 2005 Arabic handwriting recognition competition , as shown in Table 2.

Table 2- Comparison with other word recognition systems that have been presented in the ICDAR 2005 Arabic handwriting recognition competition

Systems	Training Sets	Test Set	Accuracy %
ICRA [16]	a ,b, c, d	e	65.74
SHOCRAN [16]	a ,b, c, d	e	35.70
TH-OCR [16]	a ,b, c, d	e	29.62
UOB [16]	a ,b, c, d	e	75.93
REAM* [16]	a ,b, c, d	e	15.36
ARAB-IFN [16]	a ,b, c, d	e	74.69
Proposed system	**a ,b, c, d**	**e**	**91.96**

*System is tested on reduced set with 1000 names

Table 3 describes the comparison of the performance of the proposed system with other word recognition systems published in the period between 2003 and 2011 outside the competitions of the ICDAR and ICFHR. The performance of the proposed system is very competitive and outperforms the best known systems in the literature.

Table 3- Comparison with other word recognition systems

Systems	Training Sets	Test Set	Accuracy %
Pechwitz et al. [22]	a, b, c	d	89.74
Xiang et al. [24]	a , b, c	d	84.15
R. Al-Hajj et al. [1]	a, b, c	d	87.2
R. Al-Hajj et al. [2, 3]	a, b, c	d	90.96
El Abed et al. [9]	a, b, c	d	89.1
AlKhateeb et al. [12]	a, b, c	d	89.24
Benouareth et al. [5]	a, b, c	d	88.12
Benouareth et al. [6]	a, b, c	d	83.79
Dreuw et al. [8]	a, b, c	d	92.86
Proposed system	**a, b, c**	**d**	**96.36**
Elbaati et al. [10]	a, b, c, d	e	54.13
Hamdani et al. [13]	a, b, c, d	e	81.93
Kessentini et al. [14]	a, b, c, d	e	79.6
AlKhateeb et al. [12]	a, b, c, d	e	83.55
Proposed system	**a ,b, c, d**	**e**	**91.96**

The previous table shows that the proposed system outperforms all other systems reported in the literature on the two test sets (d and e).

As described before, we have presented two main contributions in this paper that improve the performance of the whole system. In the following sections, we are going to show how each of the two contributions affects the results.

6.1 Effect of pre-processing stage:

Our pre-processing stage plays an important role in enhancing the performance of the whole system because of two main reasons. First, normalizing the thickness of the letters enhances the HMM characters models and makes them not sensitive to different thicknesses. Second, fixing the width of the empty spaces in a given image to avoid the problem of the different size of spaces in the image which negatively affect the HMM performance. The recognition rate of our system without the pre-processing stage is 89.36 (dropped by 2.6 %).

Figure 11 describes some images which could not be recognized without pre-processing stage.

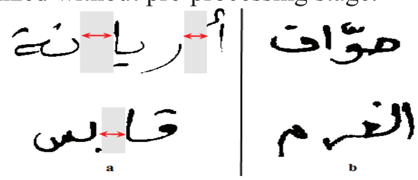

Figure 11. (a) Large spaces between different POWs, (b) thick characters .

In Figure 11.a the problem of large spaces with different widths between the POWs appears. This problem is the reason why the proposed HMM-based system could not recognize the shown words without the proposed preprocessing stage.

In Figure 11.b the problem of the different thickness of the written words appears. This problem affects the extracted features which depend on the distribution of the foreground pixels.

6.2 Effect of segmenting the image horizontally with non uniform heights:

Segmenting the image horizontally with non uniform heights depending on the distribution of the foreground pixels enhances the recognition rate of the whole system when compared with segmenting into uniform heights while using the same features and the same number of segments as shown in Table 5.

Table 5 – Comparison between two systems using uniform and non uniform heights.

Case	Training Sets	Test set	Accuracy%
Uniform	a, b, c , d	e	88
Non Uniform	a, b, c , d	e	91.96

Figure 12 shows some image which have been correctly recognized when using non uniform heights and could not be recognized when using fixed heights per frame.

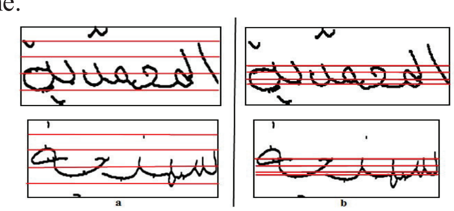

Figure 12. (a) Words that could not be recognized using fixed heights ,(b) Same words recognized using non uniform heights .

As shown in the previous figure, the non uniform segments focus on the places with concentration of pixels unlike uniform segments which do not take care of the distribution of pixels in the image. The above results show that more attention should be given, in the feature extraction stage, to the locations in the input image with high density of pixels.

7. Conclusions

In this paper, an HMM-based offline Arabic handwriting recognition system has been proposed. The first main contribution of the proposed system is the preprocessing stage. In the preprocessing, the thickness of the input word is fixed to three pixels and the spacing between the different parts of the word is fixed to an empirical threshold. The preprocessing stage has improved the performance of the overall system by 2.6 %. The second main contribution is a new technique of splitting the image into non uniform heights which has improved the performance of the system by 3.96% when compared with using fixed heights. The results achieved by the proposed system are very promising compared with other systems in the literature. Future work includes the optimization of the HMM model parameters as well as the combination of this system with systems using other types of features.

References

[1] R. Al-Hajj, L. Likforman-Sulem, and Chafic Mokbel. Arabic Handwriting Recognition Using Baseline Dependant Features and Hidden Markov Modeling. In proceedings of the Eighth International Conference on Document Analysis and Recognition (ICDAR'05), 2005.

[2] R. Al-Hajj, L. Likforman-Sulem, and Chafic Mokbel. Combination of HMM-Based Classifiers for the Recognition of Arabic Handwritten Words. In proceedings of the Ninth International Conference on Document Analysis and Recognition (ICDAR'07), 2007.

[3] R. Al-Hajj, L. Likforman-Sulem, and Chafic Mokbel. Combining Slanted-Frame Classifiers for Improved HMM-Based Arabic Handwriting Recognition. IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 31, No. 7, July 2009.

[4] Nick Bardici, and Björn Skarin. Speech Recognition using Hidden Markov Model. M.Sc.Thesis, Department of Telecommunications and Signal Processing , Blekinge Institute of Technology, 2006.

[5] A. Benouareth, A. Ennaji, M. Sellami. HMMs with explicit state duration applied to handwritten Arabic word recognition. In the proceeding of 18^{th} Int. Conf. Pattern Recognition (ICPR), 2006.

[6] A. Benouareth, A. Ennaji, M. Sellami. Semi-continuous HMMs with explicit state duration for unconstrained Arabic word modeling and recognition. Pattern Recognition Lett. 29, 1742–1752, 2008.

[7] R. C. Gonzales and R. E. Woods, Digital image processing, second edition, Addison-Wesley, 2002.

[8] P. Dreuw, S. Jonas, H. Ney. White-space models for offline Arabic handwriting recognition. In the Proceeding of 19^{th} Int. Conf. Pattern Recognition (ICPR), 2008.

[9] Haikal El Abed and Volker Märgner. Comparison of Different Preprocessing and Feature Extraction Methods for Offline Recognition of Handwritten ArabicWords. in proceedings of the Ninth International Conference on Document Analysis and Recognition (ICDAR'07), 2007.

[10] Abdelkarim Elbaati, Houcine Boubaker, Monji Kherallah, Adel M. Alimi, Abdellatif Ennaji, and Haikal El Abed. Arabic handwriting recognition using restored stroke chronology. in proceedings of the 10th International Conference on Document Analysis and Recognition (ICDAR), pp. 411–415, July 2009.

[11] HTK Speech Recognition Toolkit, http://htk.eng.cam.ac.uk/

[12] Jawad H. AlKhateeb, Jinchang Ren, Jianmin Jiang and Husni Al-Muhtaseb . Offline handwritten Arabic cursive text recognition using Hidden Markov Models and re-ranking. Pattern Recognition Letters 32, 2011.

[13] M.Hamdani, H. El Abed, M. Kherallah and Adel M. Alimi. Combining Multiple HMMs Using Online and Offline Features for Offline Arabic Handwriting Recognition. in proceedings of the 10th International Conference on Document Analysis and Recognition (ICDAR), 2009.

[14] Y. Kessentini , T.Paquet and Abdel Majid Ben Hamado. Offline handwritten word recognition using multi-stream hidden Markov models. Journal Pattern Recognition Letters, Vol.1, Issue 1, January, 2010.

[15] C. Liu, K. Nakashima, H. Sako, and H. Fujisawa. Handwritten digit recognition: benchmarking of state-of-the-art techniques. Pattern Recognition, vol. 36, pp. 2271-2285, 2003.

[16] V. Märgner, M. Pechwitz, and H. El Abed. ICDAR 2005 Arabic handwriting recognition competition. in Proceedings of the 8th Inter. Conf. on Document Analysis and Recognition, volume 1, pp. 70–74, 2005.

[17] V. Märgner and H. El Abed. ICDAR 2007 Arabic handwriting recognition competition. in Proceedings of the 9th Inter. Conf. on Document Analysis and Recognition (ICDAR), pp. 1274–1278, 2007.

[18] V. Märgner and H. El Abed. ICDAR 2009 Arabic handwriting recognition competition. in Proceedings of the 10th Inter. Conf. on Document Analysis and Recognition (ICDAR), pp. 1383–1387, 2009.

[19] V. Märgner and H. El Abed. ICFHR 2010 Arabic handwriting recognition competition. in Proceedings of the 12th Inter. Conference on Frontiers in Handwriting Recognition(ICFHR), 2010.

[20] V. Märgner and H. El Abed. ICDAR 2011 Arabic handwriting recognition competition. in Proceedings of the 11th Inter. Conf. on Document Analysis and Recognition (ICDAR), 2011.

[21] 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 surl'Ècrit et le Document (CIFED '02), pp. 129–136, Hammamet, Tunisia, October 2002.

[22] M. Pechwitz and V. Maergner. HMM Based Approach for Handwritten Arabic Word Recognition Using the IFN/ENIT-Database. In proceedings of the Seventh International Conference on Document Analysis and Recognition (ICDAR'03), 2003.

[23] M. S. Khorsheed. Offline Arabic character recognition - a review. Pattern Analysis & Apps., vol. 5, pp. 31-45, 2002.

[24] D. Xiang, H.Yan, X.Chen and Y.Cheng. Offline Arabic Handwriting Recognition System based on HMM. Computer Science and Information Technology ICCSIT, 3rd IEEE International Conference, 2010.

[25] Ching Y. Suen , L. Lam and Seong-Whan Lee. Thinning Methodologies-A Comprehensive Survey. IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 14, No. 9, page 879, September 1992.