HMM-based Offline Arabic Handwriting Recognition
Using new Feature Extraction and Lexicon Ranking Techniques

Hesham M. Eraqi and Sherif Abdelazeem
Electronics Engineering Department
The American University in Cairo
Cairo, Egypt
hesham.eraqi@gmail.com, *shazeem@aucegypt.edu*

Abstract

In this paper, a new offline Arabic handwriting recognition system is presented. The Douglas-Peucker algorithm is applied on the skeletonized parts of the offline images to convert it into piecewise linear curves that are used for efficient detection of diacritics, noise segments, and the baseline. A hidden Markov model (HMM)-based system is used with features extracted from the image before and after removing the diacritics. A reliable method of lexicon ranking and reduction based on the information of the image's diacritics, number of piece of Arabic words (PAWs), and dimensions information is used. The proposed system has been tested using the IFN/ENIT database and has achieved promising recognition rates.

1. Introduction

Arabic, one of the six United Nations official languages, is the native language for more than 221 million people in the world [1], and over 1 billion people use it in several religion and culture-related activities. The characters of Arabic script are inherently cursive for both of its printed and handwritten forms; writing isolated characters in ‘block letters’ is an unacceptable and unused writing style. The Arabic alphabet contains 28 letters. Each has between two and four shapes, and some characters, especially in Arabic handwriting, may overlap with their neighboring characters forming what is called a “ligature”. Arabic script is rich in diacritics that represent short vowels or other sounds and allow differentiating the notion of the letters. In this paper, we use the term “diacritics” even more broadly to also include the dots of the letters. Offline handwriting recognition is the task of determining what letters or words are present in a digital image of handwritten text.

The cursive nature of Arabic writing requires the segmentation of words into characters, or parts of characters, before recognition. The HMM-based methods provide the advantage of joint segmentation and recognition. The HMM-based offline handwriting recognition works as follows. Handwritten images are first converted to observation sequence of variable lengths, usually by using the sliding window technique, then the Baum-Welch algorithm is used to train (define) each character’s HMM model. Finally, the best sequence of characters (HMM models) that maximizes the a posteriori probability is located through the Viterbi algorithm.

2. Pre-processing

Firstly, the connected components of the image are extracted (connected components labeling [3]) by splitting the image into a set of connected components (regions of 8-connected foreground pixels).

2.1. Skeletonization and Lines Approximation

Skeletonization is a process that reduces the width of a pattern shape to just a single pixel. Generally, for a skeletonization algorithm to be effective, it should ideally compress data and retain the significant features of the pattern. Thinning is the process of transforming a pattern from one form to another with less thickness while maintaining the connectivity of the original pattern [4], and this is why skeletonization can be achieved through a thinning process. The thinning algorithm in [4] (page 879, bottom of first column

978-0-7695-4774-9/12 $26.00 © 2012 IEEE
DOI 10.1109/ICFHR.2012.214

through top of second column) is applied iteratively to the image until it converges; i.e., skeleton does not change nor vanish even if the iteration continues.

Then, the skeletonized shape of each connected component of the test image is converted into a connected series of line segments that approximates the geometric shape of the connected component (a piecewise linear curve). First, a hit-and-miss operation is applied to the skeletonized binary image of each connected component to obtain the end and intersection points. Then a set of paths is constructed by selecting any arbitrary end or intersection point and tracking the connected single pixel path of foreground pixels beginning from that point until meeting another intersection or end points (maybe the same beginning pixel in case of loops), then selecting another end or intersection point and constructing another path, and so on until the set of formed paths covers the entire skeletonized image. Then, the recursive Douglas-Peucker line-simplification algorithm [5] is applied on the sequence of pixel points of each path to obtain the piecewise linear curves of the connected component.

2.2. Diacritics and Noise Segments Detection

2.4.1. Noise Segments and Dots Detection. The noise segments (the spurious segments that often appear in the binarized image) and the dot didactics are being detected according to the rules described in Table 1. The threshold values obtained in the table are not optimum, statistics made on handwritten images extracted from samples of the IFN/ENIT database are used to define these thresholds. The conditions of Table 1 are checked for all the connected components of the image after calculating the required features.

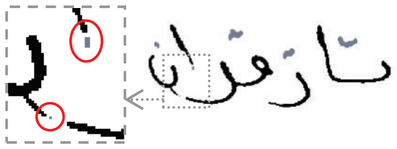

Figure 1. Examples of diacritics and noise segments. Noise segments are circled in red.

2.4.2. Other Diacritics Detection. The detected dots and noise segments are excluded from the estimation of the global average parameters of the test image, like the piecewise linear curves bounding box average area. Although the thresholds' values associated with the number of end and intersection points should have ideal values that are obtained from the standard writing styles of some diacritics (written in parentheses in Table 1), the actual values we used, obtained from the handwriting statistics, are different than those ideal values and give better detection accuracy.

2.4.3. Detection Verification. For all the detected diacritics of step (2.4.2) except for the 'Assured Diacritics' category, a verification process that confirms or rejects the detection decision of these diacritics is conducted. The objective of this step is to filter the detected diacritics except for the ones with clear geometric characteristics (which we call 'Assured Diacritics') and the dots by making use of the information of their relative writing positions to the connected components that are not detected to be diacritics ('non-diacritic'). Figure 2 shows an example of a valid diacritic component (The values of d_1 and d_2 are set empirically to 15 pixels). A diacritic component is valid if there exists a non-diacritic component such that they make a horizontal historgram overlap higher than 75% (if the diacritic component lie within L_2 it's 10% instead) and $L_1 \leq 0.35*Image\_Height$.

TABLE 1
DIACRITICS DETECTION FEATURES

Conditions \ Diacritic Types	Noise Segments	Dot	Two-Dots	'Shaddah'	'Hamza'	Triangle	Other Diacritics	Assured Diacritics
Minimum number of end points	2	-	2 (2)	2 (3)	1 (3)	2 (2)	-	2
Maximum number of end points	-	-	3 (2)	3 (3)	4 (3)	3 (2)	3	3
Maximum number of end and intersection points	2	-	4 (2)	10 (4)	6 (4)	4 (3)	5	10
Maximum ratio of the connected component bounding box area to the average value of all the components	20%	40%	40%	40%	40%	40%	40%	40%
Maximum ratio of the lines-approximated graph bounding box area to the average value of all the graphs	3%	25%	35%	40%	35%	45%	20%	30%
Maximum ratio of the lines-approximated graph sum of lines lengths to the average value of all the graphs	8%	25%	50%	55%	60%	45%	20%	40%
Minimum aspect ratio (Width/Height)	-	-	1	0.8	-	-	-	1.5
Maximum ratio of the height of the component to the image height	-	20%	20%	30%	30%	30%	30%	30%
α; where 2/α ≤ (Width/Height) ≤ α	-	-	-	-	2.5	3	3.5	-
Maximum ratio of the distance between the mean of the connected component and the horizontal histogram peak to the image height	-	10%	10%	10%	10%	10%	10%	20%
Maximum ratio of the distance between the mean of the connected component and the top of the image to the image height	-	-	-	60%	50%	60%	-	40%

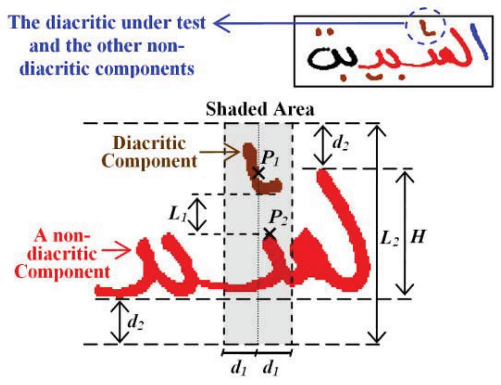
The diacritic under test and the other non-diacritic components
Shaded Area
Diacritic Component
A non-diacritic Component
P_1
P_2
L_1
L_2
H
d_1
d_2

Figure 2. Diacritics Verification. P_1 is the centroid of the diacritic. P_2 is the point in the non-diacritics component that is located within that shaded area and makes minimum distance to P_1.

2.3. Baseline Detection

Horizontal projection methods are commonly used by the OCR researchers to detect Arabic baseline [6]. In our technique, the horizontal histogram information is used along with the local writing direction information obtained by converting the image into the piecewise linear curves format in an iterative scheme, for a better detection of Arabic writing baseline.

1- Let L be the group of lines of all the piecewise linear curves of the non-diacritic components of the image. And $\alpha=15°$ (a predefind threshold).

2- H is the group of lines l_i in L, such that:

$$|\theta_i| \le \alpha \;\; or \;\; |\theta_i| \le 180°-\alpha \text{ , where } \theta_i \text{ is the angle of } l_i \quad \textbf{(1)}$$

3- IF the summation of lengths of all the lines of H is higher than a predefined threshold l_{max} (150 pixels empirically), THEN:
The centroids coordinates of all the lines of H are plotted and a weighted linear regression is computed based on minimizing the square error. The weight W_i of each centroid is determined according to three features of its line l_i:
- F_1: The length of l_i (Normalized to be from 0 to 1).
- F_2: The acute angle of l_i (From $0°$ to $90°$).
- F_3: The p value at the y-coordinate of l_i centroid divided by the peak value of p, where p is the horizontal histogram curve of the original image without diacritics (F_3 is normalized to be from 0 to 1).

And finally: $W_i = (F_1 * (1\text{-}F_2)/90°) + F_3$ **(2)**

4- ELSE increment α and repeat from step 2.

The resulting line from the linear regression represents the baseline of writing. It's important to mention that when calculating F_3, taking the mean value of p around a certain coordinate is proven to be better than taking only the exact p value on that coordinate.

2.4. Image Rotation and Spacing Regulation

Before feature extraction, a normalization step is done in two steps as shown in Figure 3. First, the image is rotated such that the baseline becomes horizontal. Second, the image is searched for any large zero periods (spacing) in the x-axis histogram of the image to be reduced to a fixed spacing amount. The latter pre-processing correction proved to give a better recognition performance than having a separate HMM model for the spacing between words and/or PAWs (Parts of Arabic Word).

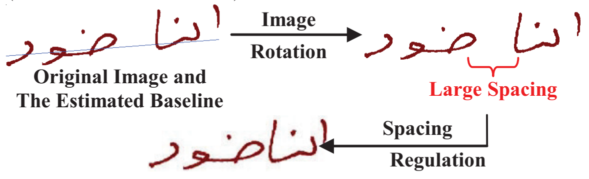

Figure 3. Image Rotation and Spacing Regulation.

3. Feature Extraction

In fact, the baseline information can be of great importance for efficient feature extraction. Unfortunately, the baseline is prone to errors, which causes considerable amount of errors for all baseline-dependent systems. In this work, integrating robust baseline-dependent and independent features has proven to be the better choice.

3.1. Baseline-independent Gradient-based Feature

The Sobel operator [7] is applied on the image to extract the vertical and horizontal gradient components:

$$g_x(x,y)=C(x+1,y-1)+2C(x+1,y)+C(x+1,y+1)-C(x-1,y-1)-2C(x-1,y)-C(x-1,y+1), \quad \textbf{(3)}$$

$$g_y(x,y)=C(x-1,y+1)+2C(x,y+1)+C(x+1,y+1)-C(x-1,y-1)-2C(x,y-1)-C(x+1,y-1). \quad \textbf{(4)}$$

And then the gradient operator [8] is applied to the image C to give two gradient components: strength $|g(x,y)|$ and direction $\angle g(x,y)$, for all the points of C:

$$|g(x,y)| = \sqrt{g_x^2(x,y) + g_y^2(x,y)}, \quad \textbf{(5)}$$

$$\angle g(x,y) = \tan^{-1}\left(\frac{g_y(x,y)}{g_x(x,y)}\right). \quad \textbf{(6)}$$

The gradient vector (expressed as strength and direction) at each point of the image C is then decomposed into the eight Freeman directions [7] such that each gradient vector is decomposed into two components in the nearest two Freeman standard directions, as shown in Figure 4. For each Freeman direction, an image is constructed from the projection of points in that direction; so that eight directional sub-images are generated.

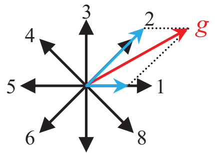

Figure 4. Decomposing the gradient vector g into two components (in blue) in the nearest two Freeman directions.

The assignment of direction codes to pixels can be viewed as a directional decomposition of the image into eight directional sub-images. A Gaussian mask is applied to each of the eight directional sub-images in order to dilate the writing contour as shown in Figure 5. Then, each directional sub-image is binarized, where the gray scale image is converted to a black and white image.

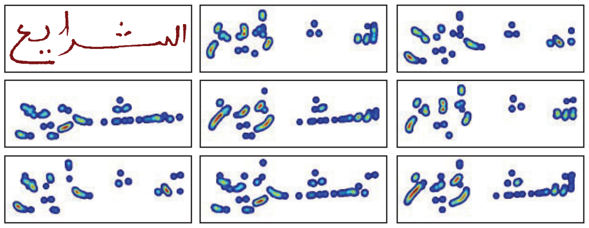

Figure 5. Directional decomposition of a sample image into eight directional sub-images.

For each image of the nine images (the normalized image C plus the eight directional sub-images), a rectangular sliding window of a fixed width W equal to five pixels and with an overlap of four pixel is used for feature extraction as shown in Figure 6. The window is divided into three vertical equal cells, and is shifted from right to left (in accordance with the Arabic writing direction). At each shift position, the following features are calculated:

1- The density of foreground pixels of each cell (D_1, D_2, and D_3).
2- The density of the window foreground pixels (D).
3- The window centroid (C).
4- The distance between the uppermost foreground pixel of the window and the lowest one (H).

3.2. Baseline-dependent Features

3.2.1. Diacritics-dependent Height and Depth Feature. A rectangular sliding window of a fixed width of one pixel and without overlap is used for feature extraction. The window is shifted from right to left across the image to generate two features (height and depth) at each shift position. The first feature (height) H is equal to the absolute distance from the baseline to the uppermost foreground pixel above the baseline. And the second one (depth) D is equal to the absolute distance from the baseline to the lowest foreground pixel under the baseline as shown in Figure 7.

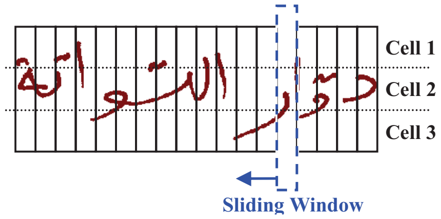

Figure 6. Sliding window-based feature extraction on on image of the nine.

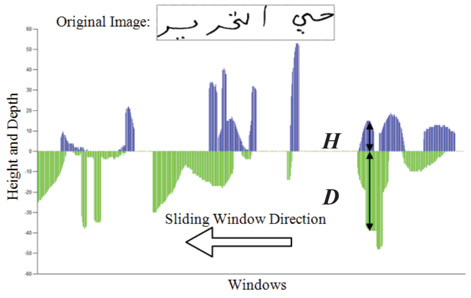

Figure 7. Height and Depth Feature Extraction.

This feature has proven to be effective for a better modeling of different Arabic characters that share the same primary part and are only different in their diacritics. For example, if we consider the sequence of the depth features of the second letter of the test word of Figure 7 'ي', it is obvious that it has values almost double the depth feature sequence for the letter 'ى' which do not have two dots under it, even though they share the same non-diacritic part.

3.2.2. Diacritics-independent Segments Feature. The detected diacritics are removed from the image. A rectangular sliding window of one-pixel width and without overlap is shifted from right to left across the image's skeleton graph. At each shift position, the number of segments (contiguous group of foreground pixels) N is computed. Actually, diacritics removal is crucial for this feature as, for Arabic handwriting, diacritics are usually drifted from their associated letters.

3.2.3. Diacritics-independent Height and Depth Feature. The same feature of section 3.2.1 is used after removing the detected diacritics from the image in order to increase the amount of information extracted from the primary parts of writing.

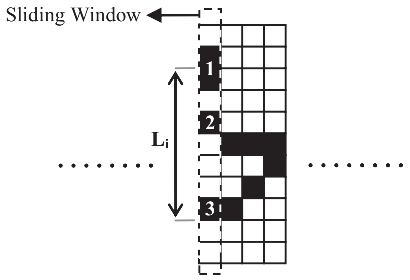

Figure 8. Pixel-level view of four consecutive sliding windows.

4. Lexicon Ranking and Reduction

A lexicon ranking and reduction technique that is based on our online lexicon reduction method proposed in [9] is used to enhance the performance of the proposed system. Instead of the online data, the ups and downs sequence *S* is obtained from the detected diacritics of the offline images, after arranging them from right to left. The vertical position of each diacritic component (up or down) is obtained according to their writing position relative to its corresponding non-diacritic component (as discussed in Section 2.4.3). The estimated baseline information is used to decide the vertical position of the diacritics that have background pixels above and below it at the same time.

As discussed in [9], the obtained diacritics sequence *S* is matched to the estimated diacritics sequence S^*_i of every lexicon candidate *i* using the minimum edit distance (MED) algorithm. A lexicon candidate *i* is eliminated from the lexicon if any of the following conditions is satisfied; where *N* is the number of the non-diacritic components of the image, $N_{min,i}$ is the number of PAWs of the candidate *i* (obtained as in [9]), *A* and *R* are the area and aspect ratio (width/height) of the image, and A^*_i and R^*_i are the average area and aspect ratio of the candidate *i* estimated using the training samples of that candidate:

$$[N > N_{min,i}] \text{ and } [(N-N_{min,i}) > 5] \quad \textbf{(7)}$$

$$[N < N_{min,i}] \text{ and } [(N_{min,i}-N) > 3] \quad \textbf{(8)}$$

$$\text{MED between } S \text{ and } S^*_i > 3 \quad \textbf{(9)}$$

$$(A/A^*_i) > 4] \text{ or } [(A/A^*_i) < (1/4) \quad \textbf{(10)}$$

$$(R/R^*_i)>2] \text{ or } [(R/R^*_i) <(1/2) \quad \textbf{(11)}$$

On the other hand, the resulting pruned lexicon is ranked according to minimizing the previous five criteria, according to the following equation, where W_1, W_2, W_3, and W_4 are weighting factors.

$$C = W_1*|N-N_{min,i}| + W_2*MED + W_3*|A-A^*_i| + W_4*|R-R^*_i| \quad \textbf{(12)}$$

5. HMM and System Block Diagram

In the proposed system, the same density HMM classifier as implemented in the HTK Speech Recognition Toolkit [10] is used for segmentation and recognition. However, we implement our own parameters of the HMM. HTK supports multiple steps in the recognition process: data preparation, training, recognition and post- processing. The data preparation process supports only the speech data, so we do not use HTK for this step that includes lexicon reduction (using the task grammar), the dictionary, and feature extraction and coding process.

HMM models the feature vector with a mixture of Gaussians distributions (32 Gaussians in our system) and uses the Viterbi algorithm in the recognition phase, which searches for the most likely sequence of letter HMM models given the input feature vector. We train a separate HMM model for all the possible shapes of the Arabic letters. For example, the Arabic letter Baa' 'ب' has 3 models: ('بـ', 'ـبـ', and 'ـب') and another 3 models in case that it comes with a 'Shaddah' diacritic ('بّـ', 'ـبّـ', and 'ـبّ'). A left-to-right HMM is implemented. Figure 9 shows the case of a five-state HMM, showing that we allowed transition to the current and the next states only. The same number of states and Gaussians are adopted for all Arabic letters.

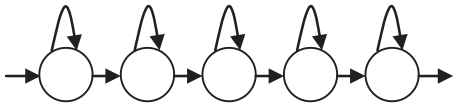
Figure 9. A five-state left-to-right HMM.

Figure 10 shows the block diagram of the system, where the final decision is made according to the HMM confidence and the ranking of the lexicon candidate.

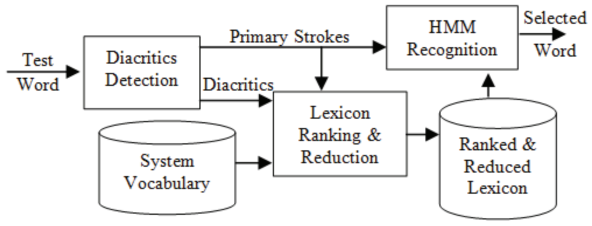

Figure 10. System Block Diagram.

6. Results

The proposed system is evaluated using the offline Arabic IFN/ENIT database. The database in version 2.0 patch (v2.0p1e) consists of 32,492 Arabic words handwritten by more than 1000 writers. Table 2 compares the proposed system's results to those of the participating systems in the off-line Arabic recognition competition held at the 8th International Conference on Document Analysis and Recognition (ICDAR 2005) [14] and the recent work in the literature. All the systems are trained using the data sets a, b, c, and d, while the test is conducted on set e.

Table 3 compares the proposed system's results to those of the participating systems in all the off-line Arabic recognition competition held at ICDAR 2007, 2009, 2011, and the 12th International Conference on Frontiers in Handwriting Recognition (ICFHR 2010).

TABLE 2
COMPARISON OF OUR RESULTS WITH PREVIOUS WORKS

System	Top 1	Top 5	Top 10
Proposed System (Without Lexicon Ranking and Reduction)	84.32	94.23	96.01
Proposed System (With Lexicon Ranking and Reduction)	**87.04**	95.11	96.78
Kessentini et al. [11] (2010)	79.60	-	-
Elbaati et al. [12] (2009)	54.13	-	-
Hamdani et al. [13] (2009)	81.93	-	-
ICRA	65.74	83.95	87.75
SHOCRAN	35.70	51.62	51.62
TH-OCR	29.62	43.96	50.14
UOB	75.93	87.99	90.88
REAM	15.36	18.52	19.86
ARAB-IFN	74.69	87.07	89.77

TABLE 3
COMPARISON OF OUR RESULTS WITH THE RECENT ARABIC HANDWRITING COMPETITIONS

System	Appr-oach	Test Set f			Test Set s		
		Top1	Top5	Top10	Top1	Top5	Top10
Proposed System *(Without Lexicon Ranking and Reduction)*	HMM	84.84	92.83	94.58	69.87	81.69	84.55
Proposed System *(With Lexicon Ranking and Reduction)*	HMM	**87.08**	93.47	94.64	**73.05**	82.96	85.25
Results of the systems at ICDAR 2011 [15]							
JU-OCR	ES	63.86	80.18	84.65	49.75	66.86	72.46
CENPARMI	SVM	40.00	69.33	74.00	35.52	54.56	63.84
RWTH-OCR	HMM	**92.20**	95.73	96.15	**84.55**	91.99	93.52
REGIM	HMM	81.36	81.52	79.03	68.44	81.99	84.98
Results of the systems at ICFHR 2010 [16]							
UPV PRHLT	HMM	**92.20**	95.72	96.29	**84.62**	91.42	93.32
REGIM	HMM	79.03	89.35	91.34	68.44	81.99	84.98
CUBS-AMA	HMM	80.32	88.26	88.96	67.90	78.58	79.87
RWTH-OCR	HMM	90.94	95.31	96.00	80.29	89.83	91.80
Results of the systems at ICDAR 2009 [17]							
UOB-ENST	HMM	83.98	91.85	93.00	72.28	85.19	87.92
REGIM	HMM	57.93	73.43	78.10	49.33	65.10	71.14
Ai2A	HMM	89.42	95.33	95.94	76.66	88.01	90.28
MDLSTM	RNN	**93.37**	96.46	96.77	**81.06**	88.94	90.72
RWTH-OCR	HMM	85.69	93.36	94.72	72.54	83.47	86.78
LITIS-MIRACL	HMM	82.09	90.27	92.37	74.51	86.14	88.87
LSTS	TNN	15.05	29.58	35.76	11.76	23.33	29.62
Results of the top 3 (out of 14 Systems) systems at ICDAR 2007 [18]							
Siemens	HMM	**87.22**	94.05	95.42	**73.94**	85.44	88.18
MIE	LDC	83.34	91.67	93.48	68.40	80.93	83.73
UOB-ENST	HMM	81.93	91.20	92.76	69.93	84.11	87.03

All the systems in Table 3 are trained using the data sets a, b, c, d, and e and tested on sets f and s. The table shows that the proposed system is ranked sixth out of 25 different recent systems on sets f and s, which confirms the effectiveness of the proposed feature extraction, and lexicon ranking and reduction techniques.

7. Conclusion

In this paper, novel diacritics detection and feature extraction techniques are proposed. The strategy used in the proposed feature extraction technique lies in combining efficient baseline-dependent and baseline-independent features that are extracted from the image before and after removing the diacritics segments, in an HMM-based system which has proven to achieve promising recognition rates indicating the effectiveness of the proposed techniques.

References

[1] M. Paul Lewis, *Ethnologue: Languages of the World*, 16th ed. SIL Int'l, 2009.

[2] M. Pechwitz, S. S. Maddouri, V. Mrgner, N. Ellouze, and H. Amiri, "IFN/ENIT-database of handwritten Arabic words," in proc. of CIFED, pages 129–136, 2002.

[3] Yebin Fan, Shengsheng Yu, Hualong Zhao, "A novel line based connected component labeling algorithm," *IEEE Inter. Conf. on Computer Science & Info. Tech. (ICCSIT)*, 2010.

[4] Lam, L., Seong-Whan Lee, and Ching Y. Suen, "Thinning Methodologies-A Comprehensive Survey," *IEEE Trans. on Pattern Anal. and Machine Intelligence*, vol. 14, no. 9, 1992.

[5] D.H. Douglas, T.K. Peucker, "Algorithms for the reduction of the number of points required to represent a digitized line or its caricature," *Cartographica: the International Journal for Geographic Information and Geovisualization*, vol. 10, pp. 112–122, 1973.

[6] H. M. Eraqi and S. Abdelazeem, "An On-Line Arabic Handwriting Recognition System Based on a new On-line Graphemes Segmentation Technique," *In proc. of the 11th Inter. Conf. on Document Analysis and Recogn.* (ICDAR '11), pp. 409-413, 2011.

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

[8] C. Liu, K. Nakashima, H. Sako, and H. Fujisawa, "Handwritten digit recognition: benchmarking of state-of-the-art techniques," *Pattern Recogn.*, vol. 36, 2003.

[9] S. Abdelazeem and H. M. Eraqi, "On-line Arabic Handwritten Personal Names Recognition System Based on HMM*," in proc. of the 11th intern. Conf. on Document Analysis and Recognition (ICDAR '11)*, 2011.

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

[11] Y. Kessentini, T. Paquet and A. BenHamadou, "Off-line handwritten word recognition using multi-stream hidden Markov models," *Pattern Recogn. Letters*, vol. 1(1), 2010.

[12] A. Elbaati, H. Boubaker, M.Kherallah, A. M. Alimi, A. Ennaji, and H. 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, 2009.

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

[14] 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*, vol. 1, pp. 70–74, 2005.

[15] V. Märgner and H. El Abed, "ICDAR 2011 - Arabic Handwriting Recognition Competition," *2011 International Conf. on Document Analysis and Recognition (ICDAR)*, 2011.

[16] V. Märgner and H. El Abed, "ICFHR 2010 – Arabic Handwriting Recognition Com-petition," *in Proceedings of the 12th International Conference on Frontiers in Handwriting Recognition (ICFHR)*, 2010.

[17] H. El Abed and V. Märgner, "ICDAR 2009 – Arabic Hand-writing Recognition Competition," *International Journal on Document Analysis and Recogn.*, vol. 14(1), pp. 3–13, 2011.

[18] V. Märgner and H. El Abed, "ICDAR 2007 – Arabic Hand-writing Recognition Competition," *in Proceedings of the 9 the International Conf. on Document Analysis and Recognition (ICDAR)*, vol. 2, 2007.