On-line Arabic Handwritten Personal Names Recognition System based on HMM

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

***Abstract*— In this paper a new on-line handwriting recognition system for Arabic personal names based on Hidden Markov Model (HMM) is presented. The system is trained with the ADAB-database using two different methods: manually segmented characters and non-segmented words. This work presents a recognition system dealing with a large vocabulary of 2800 Arabic personal names using a new lexicon reduction method that depends on the delayed strokes formation and the number of strokes. Besides, a new delayed strokes detection method is used to reduce the temporal variation of the on-line sequence. A dataset of on-line Arabic handwritten names has been collected to validate the system and a highly encouraging recognition rate is achieved compared to the results of commercially available recognition systems on the same dataset.**

Keywords- on-line handwriting; Arabic personal names; lexicon reduction; delayed strokes detection

I. INTRODUCTION

In recent years, on-line handwriting recognition systems started to have several applications; mainly due to the increasing popularity of personal digital assistants (PDA), tablet PCs, and smart phones that use a pen as a convenient and portable input method. There have been significant advancements in the area of handwriting recognition for Latin-based languages. However, Arabic handwriting recognition received less attention by the research community. The cursive nature of Arabic, characters overlap "ligatures", and delayed strokes are some of the key problems that make Arabic recognition more difficult than other languages such as Latin or Chinese [6]. ". In Arabic writing, each word consists of at least one PAW, while each PAW is composed of some connected characters (at least one character). More details about Arabic writing characteristics can be found in [6].

In this paper, we present a recognition system dealing with a large vocabulary of unconstrained handwritten Arabic personal names which has important applications, especially for smart phones. However the methods discussed in this paper are still valid if the personal names lexicon is replaced with another.

A set of around 70,000 personal names were collected from the databases of different organizations. This set is reduced to remove the repeated names and resulted into 2,800 different Arabic personal names that represent the vocabulary of our system. The system is trained using the on-line database ADAB of Tunisian town names that is divided into 3 sets containing 23,251 words (122,559 characters) [1].

II. PRE-PROCESSING

A. *Smoothing and Resampling*

Smoothing is important to remove the jaggedness of the contour resulting from the handwriting irregularity and the imperfection caused by the acquisition device. Every point $P_t = [x(t), y(t)]$ in the trajectory is replaced according to the following equation:

$$P_{t_{new}} = \sum_{k=-n}^{n} \alpha_k P_{t+k} \ , \ \sum_{k=-n}^{n} \alpha_k = 1$$

For each point to be the mean value of itself and its (2n) neighbors: $\alpha_k = (2n+1)^{-1}$. Besides, a writing speed normalization (resampling) algorithm based on trace segmentation method explained in [7], is used to redistribute data points (originally sampled in equal time intervals) to enforce even spacing (resampling distance) between them. Experiments with our system have showed that resampling distance should be inversely proportional to the number of states of every character HHM model.

Figure 1 shows the effect of smoothing and resampling on the data. After resampling, points of every stroke are equidistant, regardless how many points are produced in each stroke.

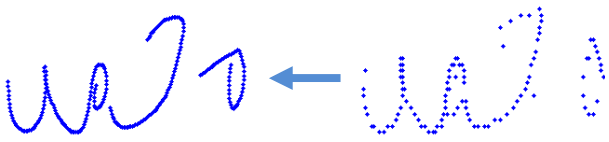


Figure 1. Smoothing and Resampling of the Arabic name "محمد"

B. *Delayed Strokes Detection*

Many of the Arabic characters share the same primary part and are distinguished from each other by the secondary part which we call in this paper "delayed strokes" (shown in Figure 2, in red). Delayed Strokes are a well-known problem in online handwriting recognition. These strokes introduce additional temporal variation to the online sequence because the writing order of delayed strokes is not

1520-5363/11 $26.00 © 2011 IEEE
DOI 10.1109/ICDAR.2011.262

fixed and varies among different writers. Hence, detecting and removing delayed stroke is an important key step in our system to allow meaningful online features extraction.

Experiments with delayed strokes characteristics showed that all the Arabic handwriting delayed strokes can be divided into 8 categories. All the delayed strokes categories share a set of common global properties, like stroke's dimensions, direction, number of points, trace duration, and the vertical distance from the baseline. Figure 2 shows some examples of delayed strokes where each one is labeled with the number of the category it belongs to. The baseline is being detected using the traditional horizontal projection method [8].

Figure 2. Delayed Strokes Categories

The main differences between the delayed strokes categories can be summarized in the following description of each category:

1- Single dot or connected two-dot stroke that is fully contained above or below a primary stroke and is characterized by its small area, short trace duration, and not being within the range of the baseline.
2- Single dot or connected two-dot stroke that share the main characteristics of (1), but is slightly drifted from its primary stroke.
3- Connected three-dot stroke that is fully contained like (1), but have a larger area and aspect ratio around 1.
4- The "Hamza" stroke which has similar characteristics as (3), but also has an explicit zigzag shape that is clear to be detected.
5- The "Hamza" stroke associated with the Arabic characters "أ" and "إ", which is located above or under a vertically straight primary stroke.
6- The "Maad" stroke associated with the Arabic character "آ", which comes above a vertically straight primary stroke and don't have the zigzag shape.
7- The slant straight stroke associated with the Arabic character "ك". It's the only delayed stroke that may be written before its primary stroke.
8- The vertically straight stroke associated with the Arabic characters "ظ" ,"ط", and the ligature "لا".

Delayed strokes are detected using a holistic approach that is based on a set of Boolean expressions describing the stroke dimensions, shape, number of points, trace duration, and the vertical distance from baseline for each category of the eight. Besides the associated main stroke relative area and position to the delayed stroke. Once a stroke is detected to belong to any of the delayed strokes categories, it's considered a delayed stroke that should be removed from the on-line sequence during different phases of our system as it is discussed later.

III. LEXICON REDUCTION

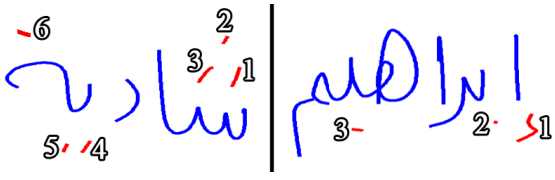
Lexicon reduction is the process of initially eliminating lexicon entries unlikely to match the given test word. This process, lexicon reduction, has desirable effects not only on the recognition time, but also upon the recognition accuracy [2]. In our system, lexicon reduction is based on the number of primary strokes besides the number, vertical position and the writing order of the delayed strokes. The lexicon reduction method used in our system is described in the following steps.

A. Test Word Information

First of all, the delayed strokes of the test word are detected using the method discussed in Section II, and then we obtain the following two pieces of information:

1- The number of primary strokes (N):
   It is obtained by counting the number of strokes that are not detected to be delayed strokes.
2- The delayed strokes sequence string (S):
   Firstly, the writing order of the delayed strokes is rearranged so that the delayed stokes to the right comes first as shown in Figure 3. The vertical position of each delayed stroke (up or down) is determined by making use of the baseline information and the relative position of the delayed stroke with respect to its primary stroke.
   The delayed strokes sequence is a string that consists of downs (D) and ups (U) which represents the vertical positions of the rearranged delayed strokes as shown in Figure 3.

Figure 3 shows two samples from the validation dataset of the Arabic names "إبراهيم" and "شادية", where the detected delayed strokes and primary strokes are shown in red and blue. The numbers on the figure describe the modified writing order of the delayed strokes.



	شادية	إبراهيم
N	3	4
S	"U U U D D U"	"D D D"

Figure 3. Delayed Strokes Rearrangement and Test Word Information

B. *Lexicon Entries Information*

In order to know which lexicon entries are unlikely to match the given test word to be eliminated from the lexicon, we obtain the following two pieces of information for all the lexicon entries:

1- The minimum number of strokes (N_{min}):
The minimum number of strokes of a word is equal to the number of PAWs of it. For any Arabic word (single word, i.e. no spacing) the last character of every PAW (except for the last PAW of the word) is guaranteed to be one of the following characters:

ا أ إ آ ة ء د ذ ر ز و ؤ ى

These characters, end-of-the-PAW characters, are the only Arabic characters that don't come in the middle nor the beginning of a PAW. While all the other Arabic characters don't come at the end of a PAW unless they are already the last character of the word. So the minimum number of strokes can be calculated by:

N_{min} = Number of PAWs =
Number of End-of-the-PAW Characters of the word +1

2- The delayed strokes sequences regular expression (S*):
The number of the delayed strokes associated with some Arabic characters is not fixed, and varies among different writers within a fixed lower and upper bounds. For example, the Arabic character "ش" delayed strokes may be written in one triangle stroke, two strokes, or three dot strokes as shown in Figure 4.

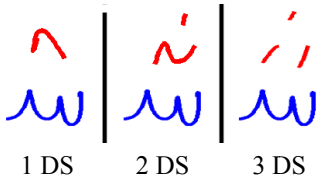


Figure 4. Number of Delayed Strokes of the Arabic Character "ش"

For the lexicon entry, a regular expression (S*) that describes all the possible strings that describe the delayed strokes sequence is obtained. Table I shows all the Arabic characters that have delayed strokes and the lower and upper bounds of them. Besides, the regular expression part for each character.

TABLE I. ARABIC CHARACTERS DELAYED STROKES

Arabic Characters	**Delayed Strokes**	**Reg. Expression (S*)**
ك (In beginning or middle of a PAW)	Either 1 up or no delayed strokes	[U]
ج (In isolated form)	Either 1 up or 1 down delayed stroke	U|D
آ أ ؤ ئ خ ذ ز ض ط غ ف ن	1 up delayed stroke	U
The ligature: لا		
ق ة ت	Either 1 or 2 up delayed strokes	U [U]
ش ث	Either 1, 2, or 3 up delayed strokes	U [U] [U]
ظ	2 up delayed strokes	U U
The ligatures: لآ لأ		
إ ب	1 down delayed strokes	D
ج (In beginning or middle of a PAW)		
ي	Either 1 or 2 down delayed strokes	D [D]
The ligature: لإ	1 up then 1 down delayed strokes	U D

By iterating all the characters of the word, the regular expression (S*) which describes all the possible delayed strokes sequence strings is obtained, where the vertical bars denote alternatives and the square brackets denote optional delayed strokes.

These two pieces of information (N_{min} and S*) are obtained for all the 2800 lexicon entries and are stored offline to increase the speed of the system. More examples of (N_{min} and S*) are in Table II.

C. *Eliminating Lexicon Entries*

For each test word, N and S are obtained, according to the methods explained before, as well as the stored values of N_{min} and S* for all the lexicon entries. For each lexicon entry, if one of the following 2 conditions is satisfied, this lexicon entry is eliminated from the lexicon:

1- $N < N_{min}$.
2- S doesn't match S*.

Table II shows some examples of the eliminated and not eliminated lexicon entries for the two test names of Figure 3 ("شادية" and "إبراهيم").

TABLE II. LEXICON REDUCTION EXAMPLE

Lexicon Entry	N_{min}	S*	Eliminated	
			شادية N=3 S:UUUDDU	إبراهيم N=4 S:DDD
إبراهيم	4	D D D [D]	√	
الإسكندرانى	6	D U U	√	√
إيهاب	3	D D [D] D	√	
ثنية	1	U [U] [U] U D [D] U [U]		√
شادية	3	U [U] [U] D [D] U [U]		√
نور الدين	5	U D [D] U	√	√

IV. FEATURE EXTRACTION

Feature extraction is a very important step in every on-line recognition system. In this phase of the system, the input data represented in the captured sequence of points $(x(t), y(t))$ are prepared to be used by the HMM classifier that needs temporal information of the input data. Furthermore, the stochastic modeling offered by HMMs, is able to cope with sequence of observations (feature vectors) of variable lengths.

In our system, 6 types of on-line features are used, that include some features that can already be found in the literature and others which that we have implemented for the first time. The on-line features used in our system are divided into two types of features: Local and vicinity features.

A. Local Features:

1- Delta X and Y:
The relative change of each point's (P_t) x-value and y-value with the following point which is represented with Δx(t) and Δy(t) as shown in Figure 5, where:

$$\Delta x(t) = x(t+1) - x(t), \Delta y(t) = y(t+1) - y(t).$$

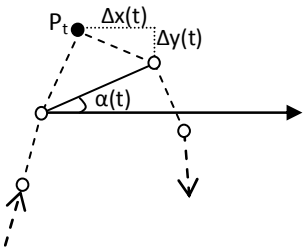


Figure 5. Delta X and Y, and Writing Direction Features

2- Writing Direction:
It describes the local writing direction using the cosine and sin of α(t), where α(t) is the angle between the line connecting P_{t-1} and P_{t+1} and the positive direction of the x-axis [3] as shown in Figure 5.

3- Angle:
The angle θ(t) between each two points of the trajectory in radians, as shown in figure 6.

B. Vicinity Features

Figure 6 shows the vicinity of a point P_t, containing P_t and the group of the preceding and succeeding points, where the number of points of the vicinity (N) equal to 9 points.

1- Aspect:
It characterizes the height-to-width ratio of the bounding box of the vicinity of P_t as shown in Figure 6. It's represented with A(t) [9], where:

$$A(t) = \frac{\Delta y^*(t) - \Delta x^*(t)}{\Delta y^*(t) + \Delta x^*(t)}$$

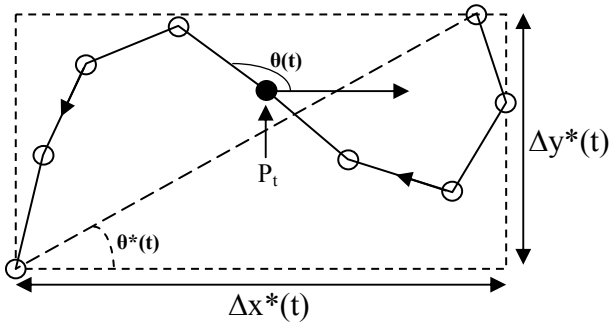


Figure 6. Feature Extraction for Vicinity Features

2- Curliness:
Curliness C(t) is a feature that describes the deviation from a straight line in the vicinity of Pi [5], where:

$$C(t) = \frac{L(t)}{\max(\Delta x^*(t), \Delta y^*(t))} - 1$$

And L(t) is the length of the trajectory in the vicinity of P_t, i.e., the summation of lengths of all line segments that are shown Figure 6.

3- Slope:
The slope S(t) of the straight line joining the first and last point in the vicinity of P_t as shown in Figure6, where:

$$S(t) = \tan \theta^*(t)$$

Experiments with the on-line recognition system discussed in this paper showed that the size of the vicinity (N) depends on the resampling distance, i.e., for shorter the resampling distance, more points are needed in the vicinity.

V. HMM AND TRAINING

In our system, the same density HMM classifier without modification as implemented in HTK Speech Recognition Toolkit [4] is used for segmentation and recognition. However, we implement our own parameters of the HMM. HTK models the feature vector with a mixture of Gaussians distributions (16 Gaussians in our system) and uses the Viterbi algorithm in the recognition phase, which searches for the most likely sequence of characters given the input feature vector.

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.

In this paper a left-to-right HMM is implemented. Figure 7 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 is adopted for all Arabic characters. And the same number of Gaussians as well.

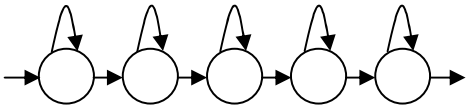


Figure 7. A Five-state Left-to-right HMM

Delayed strokes are detected as discussed previously in Section II (B) and removed from all the files of the database. And then the feature vector is obtained using the methods described previously in Section IV for all the database files that represent the observation list for training. HMMs are trained with the primary part of the data. So, the Arabic characters which have a common primary part and are only different in delayed strokes (like character "س" and "ش"), belong to the same class, and are given the same label during the HMM training phase.

There are some Arabic personal names which have the same primary part and the only difference between them is the delayed strokes, like for example the Arabic names "الغالى" and "العالى". They both have the same primary part and they correspond to the same sequence of HMMs. This confusion is solved during the lexicon reduction phase (Section III), i.e., there exists only.

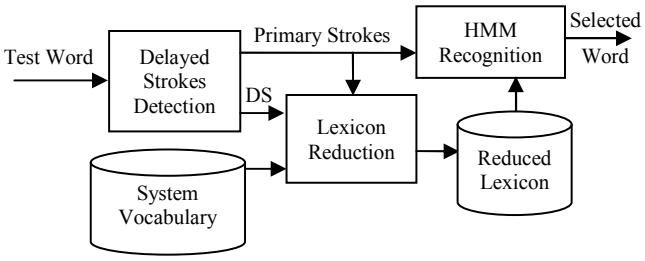


Figure 8. System Block Diagram

Most of the HMM-based cursive writing recognition systems, especially for Arabic, use the samples of the training database in its cursive form, where HMM is used for automatic characters segmentation. In our system we have manually annotated the ADAB database with character segmentation boundaries that is freely available for non-commercial research (hesham.eraqi@gmail.com) and used it for our system training which gave better performance.

VI. Results

A test dataset of 300 handwritten personal names have been gathered to evaluate the system performance. 10 writers (5 females and 5 males) with different educational background and ages (between 12 and 70) were chosen and asked to type randomly any 30 Arabic personal names they think of. While the dataset is collected using a Digimemo® tablet.

This dataset is used to validate our system. Table II shows the test results of our system on the collected dataset and the effect of lexicon reduction on system performance, besides the results of the commercially available product VisionObjects MyScript® Studio Notes Edition [10] on the same dataset.

TABLE III. System Performance

Test Type	Word Recognition Rate		
	Top1	*Top2*	*Top10*
Our System (Without Lexicon Reduction)	55.37%	65.44%	77.52%
Our System (With Lexicon Reduction)	92.05%	93.38%	96.03%
MyScript® Studio Notes Edition	73.47%	-	-

Using lexicon reduction has not only decreased the recognition time, but also it has a remarkable effect on the system's word recognition rate.

VII. Conclusion and Future Work

We presented a new on-line recognition system for the Arabic handwritten personal names using HMM. The system is based on new delayed strokes detection and lexicon reduction methods. The system is validated using a dataset, and a highly encouraging recognition rate is achieved compared to the results of commercially available recognition systems on the same dataset.

Increasing the training samples from multi writers, especially samples written in "Ruq□ah" style is expected to give much better performance. Also, making different models with different number of states and number of Gaussians for each character and adding some offline features are expected to give better performance. Also, for the lexicon reduction approach, it would be important to know if the information in N and S* could be used in a softer way by weighting the lexicon rather than pruning it.

References

[1] H. El-Abed, M. Kherallah, V. Märgner and A. M. Alimi, "On-line Arabic handwriting recognition competition - ADAB database and participating systems," International Journal on Document Analysis and Recognition (2010).

[2] D. Guillevic, D. Nishiwaki and K. Yamada, "Word lexicon reduction by character spotting," in Proc. *7th International Workshop on Frontiers in Handwriting Recognition*, pp. 373–382 (2000).

[3] I. Guyon, P. Albrecht, Y. Le Cun, J. Denker and W. Hubbard, "Design of a Neural Network Character Recognizer for a Touch Terminal," Pattern Recognition, vol. 24(2), pp. 105–119 (1991).

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

[5] S. Jaeger, S. Manke, J. Reichert, and A. Waibel, "Online Handwriting Recognition: The NPen++ Recognizer," International Journal on Document Analysis & Recognition, vol. 3(3), pp. 169-180, (2001).

[6] M. S. Khorsheed, "Off-line Arabic character recognition - a review," Pattern Analysis & Apps., vol. 5, pp. 31-45 (2002).

[7] M. Pastor, A. Toselli and E. Vidal, "Writing speed normalization for on-line handwritten text recognition," in *Proc. ICDAR,* 2005.

[8] M. Pechwitz and V. Maergner, "Baseline estimation for arabic handwritten words," in *8th Inter. Workshop on Frontiers in Handwriting Recognition (IWFHR'02)*, pp. 479–484 (2002).

[9] M.E. Schenkel, "Handwriting Recognition Using Neural Networks and Hidden Markov Models," in Series in Microelectronics, vol. 45 (1995).

[10] Vision Objects. Myscript handwriting recognition engine. http:// www.visionobjects.com/handwriting-recognition/how-does-myscript-work/ Nov 2009.