Online Arabic Handwritten Digits Recognition

Sherif Abdel Azeem, Maha El Meseery, Hany Ahmed
Electronics Engineering Department
American University in Cairo,Cairo, Egypt
shazeem@aucegypt.edu, melmeseery@aucegypt.edu,hanyahmed@aucegypt.edu

Abstract

This paper fills a void in the literature of online Arabic handwritten digits recognition as no systems are dedicated to this problem. The two main contributions of this paper are introducing a large online Arabic handwritten digits dataset and developing an efficient online Arabic handwritten digits recognition system. In the dataset, we collected 30,000 online Arabic digits from 300 writers. The developed system uses a combination of temporal and spatial features to recognize those digits. The system achieved 98.73% recognition rate. Comparison with a commercial product demonstrates the superiority of the proposed system.

1 Introduction

The field of online handwriting recognition is one of the research areas that can immensely improve human computer interaction. Real-time online systems are currently under high demand due to the hand-held devices revolution. Online handwriting systems use a digitizing tablet or an input device that traces the user movement. In those systems the input device stores a series of x and y coordinates that represent the user's pen path.

Naming conventions for different numeral systems may be confusing. Digits used in Europe and several other countries are sometimes called Arabic Numbers; and digits used in the Arab world are sometimes called Hindi umbers. In this paper, we use a different naming convention, digits used in Europe will be referred to as Latin digits and those used in the Arab world as Arabic Digits. It is worthwhile mentioning here that Arabic, Urdu (digits used in India) and Persian (digits used in Iran) handwritten digits are similar but not identical. Table 1 shows the handwritten Arabic digits from 0 to 9 along with their different writing styles as well as their printed versions.

In the last few years, online Latin digits recognition problem has been extensively researched [5, 8, 3, 14]. In [8], Kherallah et.al used a trajectory and velocity elliptical modeling for the input strokes. They achieved 95.08% on 30,000 Latin digits. Also, Ahn et. al [3] obtained a result of 96% using an elastic curvature matching algorithm to match between curvatures of reference and test digits. It is clear from the amount of attention paid for online Latin digits recognition that the literature already has online digits databases, recognition systems, and research results that achieved high level of accuracy. On the other hand, online Arabic digits recognition systems are extremely rare or nonexisting. The available research is either for Farsi or Urdu digits [6, 13] that are similar but not identical to Arabic digits. Moreover, those systems have not tested their work using a comprehensive online digits database. Portus et.al [13] tested their system using 75 samples per digit and Razzak et.al [6] tested their system using 900 samples only. The online Arabic handwritten digits recognition research is absent from the literature.

Table 1: Arabic Printed and Handwritten Digits.

Latin Equivalent	0	1	2	3	4	5	6	7	8	9
Printed	٠	١	٢	٣	٤	٥	٦	٧	٨	٩
Typical Handwritten	٠	١	٢	٣	٤	٥	٦	٧	٨	٩
Other Writing Style	–	–	–	٣	–	–	–	–	–	–

This paper aims at filling this void in the literature. The first contribution of this paper is presenting an Arabic online digits dataset collected form 300 writers. The second main contribution of this paper is introducing an efficient system to recognize online Arabic digits. The system is divided into a a pre-processing stage that eliminates the noise coming from the input device. After pre-processing, the features extraction stage computes a set of both online and off-line features from the users' strokes. The final stage uses the computed feature vector to classify the input into one of the Arabic digits.

978-0-7695-4774-9/12 $26.00 © 2012 IEEE
DOI 10.1109/ICFHR.2012.249

Figure 1: Digit four (a) Before resampling (B) After resampling

2 Arabic On-line Digits Dataset : AOD

Even though there are Arabic character datasets [4, 10], there is no online Arabic digits dataset. Therefore, we collected a large Arabic online digits dataset (AOD). To ensure including different writing styles, the database was gathered from 300 writers varying over different age groups with more than 75% in the age group between 20 and 35. Our youngest writer is 11 years old and our oldest is 70 years old. More than 90% are right handed and around 60% of the writers are females. Each writer was asked to write an average of 10 sample per digit with no constraints on the number of strokes for each digits or the writing style in orientation or size. We collected 30,000 samples which means 3000 sample per digit. The data set was collected using DigiMemo 5.9×8.3 Inch. Then each digit is labeled and the ground truth data is stored along with the strokes information. The AOD is intended to be a benchmark for online Arabic handwritten digits recognition research and, henceforth, we have made it available for free at: http://www.aucegypt.edu/sse/eeng/Pages/AOD.aspx

3 Pre Processing

Due to the irregular movement of the hand and the inaccuracy of the digitalization process, the preprocessing step is crucial for the rest of the system. Preprocessing goes through two stages: Re-sampling and smoothing.

3.1 Resampling

Linear interpolation [12] is used to solve the problem of irregularity in the distribution of points and large distances between consecutive points, as shown in Fig. 1.

We noticed that some writers move their hands up while writing the digit which means more than one stroke in a digit, so we need to group those strokes together then concatenate the strokes into a continuous

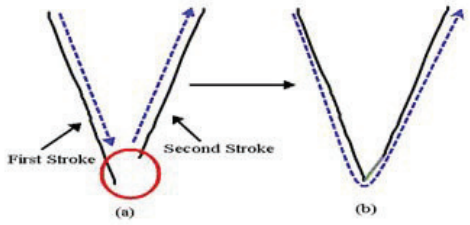


Figure 2: Linear interpolation concatenates the strokes of digit 7

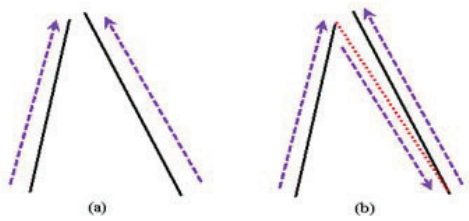


Figure 3: (a) Example of changing direction , (b) direct interpolation deforms the digit.

flow for proper recognition. Linear interpolation simply concatenates the strokes, as shown in Fig. 2.

As shown in Figure 2, the linear interpolation solved the problem of more than one stroke in a digit . Another problem is writing more than one stroke in a digit and changing the direction of the writing flow in the second stroke which results in deforming the digit , as shown in Fig. 3.

The previous problem cannot be solved using interpolation directly, the first step to solve this problem is to locate the pen up and pen down locations, the end and the start of the second stroke is tested to determine the direction of the stroke. If the user changes the writing direction of the second stroke, the points of the second stroke will be reversed before interpolating the points as shown in Fig.4.

The resampling stage also adjusts the number of (X, Y) points in each digit to a fixed number (80, empirically) for classification purposes.

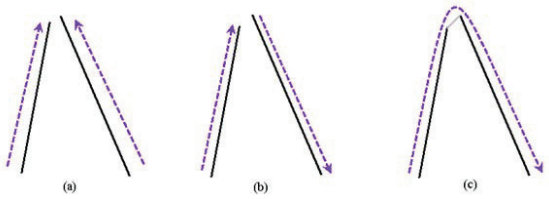


Figure 4: (a) Original writing order (b) changing the direction , (c) interpolation

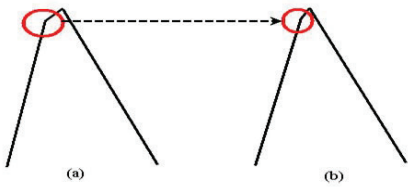


Figure 5: Digit 8 (a) Output from resampling , (b) after smoothing

3.2 Smoothing

To eliminate the noise, we perform smoothing using 5-point moving average algorithm [9] as shown in Fig. 5.

4 Feature Extraction

Feature extraction is the most important factor in achieving a good recognition rate. After studying various feature extraction methods, we selected a mixed feature vector containing a variety of temporal and spatial features.

4.1 Temporal Features

The temporal features are features based on the online information of the user's input stroke. They are computed for each point in the user's pen path. The stroke is defined as $P_i = (x_i, y_i)$, $i = 1, 2, 3..N$, where N = 80 is the number of points that represent the digit. The following features are computed:

4.1.1 F1: Directional Feature

The local writing direction at a point $P(t) = (x(t), y(t))$ at instant (t) is described [7]:

$$\cos(\alpha(t)) = \frac{\Delta y(t)}{\Delta s(t)}$$
$$\sin(\alpha(t)) = \frac{\Delta x(t)}{\Delta s(t)}$$

where $\Delta x(t)$, $\Delta y(t)$ and $\Delta S(t)$ are defined as follows:

$$\Delta x(t) = x(t-1) - x(t+1)$$
$$\Delta y(t) = y(t-1) - y(t+1)$$
$$\Delta s(t) = \sqrt{\Delta x^2(t) + \Delta y^2(t)}$$

Our experiments showed that the dependence on temporal features only was insufficient because the order of writing of some Arabic digits is not the same among different writers as shown in Fig. 6. Also, some writers tend to over-write some strokes of some digits as shown in Fig. 7 and, thus, the system gets confused and gives wrong results. Therefore, spatial features which describe the global shapes of the digits have been used.

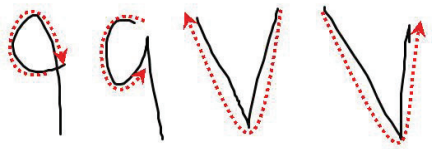


Figure 6: Various writing order of digits 7 and 9.

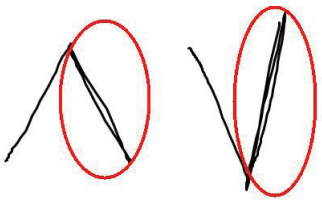


Figure 7: The over-tracing in writing a digit.

4.2 Spatial Features

Before extracting the spatial features, the user's strokes are converted into a bitmap image. The image is created from the online stroke by first connecting the stroke points by straight lines, then the image is resized to a normalized size of $30X30$ pixel. The spatial features used are divided into global shape descriptors and global concavity features.

4.2.1 F2:Global Features

- The aspect ratio characterizes the height-to-width ratio of the bounding box containing the digit.

$$Aspect = \frac{Height}{Width}$$

where $Height$ and $Width$ are the height and width of image before resizing.

4.2.2 F3: Concavity Features

The following concavity features are computed on the bitmap image $30X30$. The features describe the visual appearance of the digits and they represent a subset of several spatial features used for off-line Arabic digits recognition in [1]. Figure 8 shows the concavity features.

1. 'W2': Number of white pixels between the black pixels at the top of the digit, the black pixels at the left of the digit, the black pixels at the bottom of the digit, and the right corner of the bounding box. This feature distinguishes digit 2.

2. 'W3': Number of white pixels surrounded by black pixels to their right and left and by the top

corner of the bounding box from above. This feature distinguishes digit 3.

3. 'W4': Number of white pixels surrounded by black pixels from above and below and having the left corner of the bounding box to their left. This feature distinguishes digit 4.

4. 'W5': Number of white pixels surrounded by black pixels in all directions. This feature distinguishes digit 5.

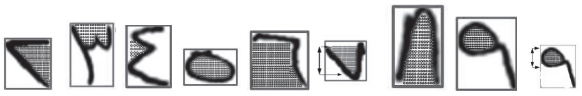
5. 'W6': Number of white pixels surrounded by black pixels from above and right and by the left and bottom corners of the bounding box. This feature distinguishes digit 6.

6. 'W7': The last row from the bottom that has 'W3' pixels is considered the depth of the 'W3' pixels. This feature differentiates between digits 3 and 7.

7. 'W8': Number of white pixels surrounded by black pixels to their right and left and by the bot tom corner of the bounding box from below. This feature distinguishes digit 8.

8. 'W9': Number of white pixels surrounded by black pixels in all directions in the upper left half of the bounding box. This feature distinguishes digit 9.

9. 'B69': The average height of the black pixels in the left half of the bounding box (That is, the distance between the first black pixel in one column and the last black pixel in the same colunm averaged over all the columns of the left half of the bounding box). This feature differentiates between digits 6 and 9.



(a) 'W2' (b) 'W3' (c) 'W4'. (d) 'W5' (e) 'W6 (f) 'W7' (g) 'W8 (h) 'W9' (i) 'B69'

Figure 8: Examples of the Spatial Features

4.3 Digit Zero Problem

The Arabic Digit zero '0' is very difficult to recognize because it has no specific way to write it, as it is only like a decimal point. Figure 9(b) shows different samples of the Arabic Digit '0'. The figure clearly shows that the digit has no specific way of writing and

(a) Arabic '0' Versus Other digits (b) Samples of the Arabic Digit

Figure 9: a) The difference in size between digits '0' and other digits. b) Different samples of the Arabic Digit '0'.

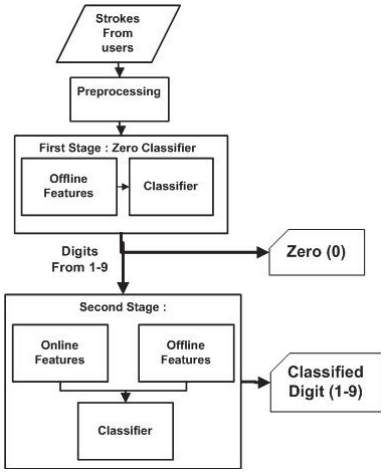


Figure 10: The system block diagram

it usually confuses with other digits especially Arabic digits '1' and '5'. The most discriminating feature for the digit '0' over the other digits is its size and number of points used to write it. Figure 9(a) shows the average size of Digit '0' versus other Arabic digits. Since digit '0' is written in a small area and random direction, it is better be recognized separately

Consequently, our final system is composed of two classification stages, the first stage recognizes the digit '0' and the second stage recognizes the rest of the digits. Figure 10 shows a block diagram of our recognition system. The system is divided into three main blocks; Preprocessing, digit '0' recognizer, and a final Classifier. The first classification stage output is either labeling the digit as a zero or continuing to the next stage. The final stage discriminates between Arabic digits from 1 to 9.

4.3.1 Digit Zero Classifier

As mentioned above, it is clear that the Arabic Digit '0' will need offline features to be distinguishible from others digits. The main feature that differentiates between 0 and other digits is the size, number of points and aspect ratio. Our experiments showed they were not enough to get the high accuracy required in the first stage. Therefore we tried various other offline features to detect the Arabic Digit '0'. The number of transi-

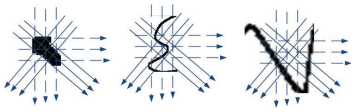


Figure 11: The number of transitions feature in digit ’0’ , ’4’ and ’7’

Figure 12

tions from white pixels to black pixel and vice-versa in a specific direction (see Fig. 11) proved to be very effective for this purpose. To reduce the dimensionality of the feature vector we chose to take only 3 transitions in every direction: vertical, horizontal, left diagonal, and right diagonal.

Figure 11 shows that Arabic digit ’0’ can be distinguished from other digits as it has a single transition in all direction in contrast to other digits. For example, digit ’4’ has a high number of vertical transitions and digit ’7’ has a high number of transitions in the horizontal direction.

5 Results

The system is composed of two stages (see Fig. 10). The first stage is the digit zero classifier and the second stage is used to classify digits from 1 to 9. Since the first stage focuses only on the digit ’0’, we used single linear SVM classifier. However, we needed a multi class classifier for the second stage. The nonlinear SVM is originally designed for 2-class problems. Extending it to multi-class can be done using the One Versus One (OVO) or the One Versus All (OVA) schemes. We used the OVO scheme to classify 9 digits which means we have 36 linear classifiers in the second classification stage.

The input feature vector for the first stage consists of 15 features divided into 12 transition features and three global feature. (see Section 4.3.1). The second stage uses a multi-class nonlinear SVM classifier with input feature vector of length 166. The 166-elements feature vector is a concatenation of the following features: F1 (156 elements), F2 (1 elements), F3 (9 elements) as explained in Sections 4.1 and 4.2.

The collected AOD dataset is divided into 80% of the writers as training set and 20% of the writers as test set. All the results given here are computed on the test set. The accurcay is defined as the nubmer of correctly classified digits divided by the total number of digits in the test set. Table 2 shows the accuracy of the different stages of the system. The results show that the first stage achieved accuracy of 99.69% and the second stage achieved accuracy of 98.89% on digits from 1 to 9 only. Table 2 shows that the overall accuracy of the system is 98.73%. Figure 12 shows a sample of some of the confused digits.

Table 2: Results of different stages

Stage	Accuracy
Stage 1	99.69%
Stage 2	98.89%
Overall System	98.73%

As we mentioned before

5.1 Comparison with commercial products

Vision Objects has built a cursive Arabic handwriting recognition system for the ICDAR 2009 Online Arabic Handwriting Recognition Competition [2] called MyScript Studio Notes Edition [11]. MyScript Studio Notes Edition attained the first place in the ICDAR 2009 competition. We chose to compare our system to MyScript Studio Notes Edition as one of the best commercial handwriting systems available today. Ten of our test set writers have been asked to write Arabic digits on My Script Notes Edition as shown in Fig.13(a).

The results of My Script recognition are shown in Fig. 13(b). It is clear from the figure that My Script confuses many Arabic digits with other Arabic characters or Latin digits because of the similarity in visual appearance with those characters. For example, Arabic digit ’4’ is confused with Arabic character ’٤’ and Arabic digit ’5’ is confused with Arabic character ’٥’. Also, Arabic digit ’0’ could not be recognized at all by My Script. This poor performance takes place because My Script recognizes both Arabic Characters and digits in one system. Our proposed system is dedicated to the recognition of Arabic digits only and, thus, produces superior results for the test digits shown in Figure 13(b) as reported in Table 3.

Table 3: Comparison between the results of the proposed system and those of My Script for the input test digits.

System	Accuracy
Our System	99.64%
MyScript	70.69%

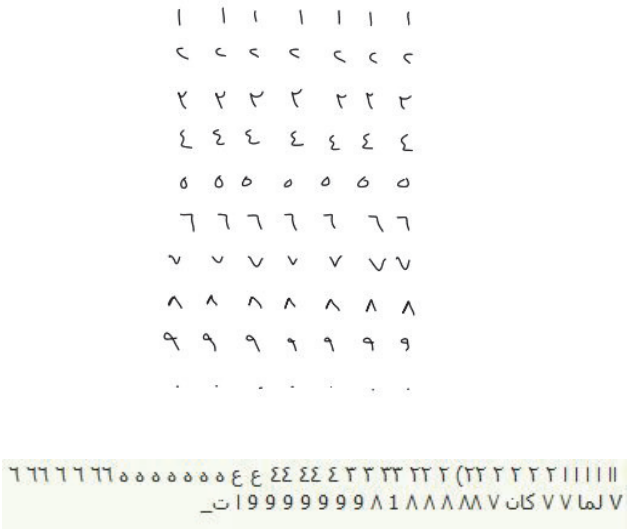


Figure 13: (a)Test digits (b) My Script results

6 Conclusion

In this paper we presented a system that uses both temporal and spatial information to recognize online Arabic digits. A new online Arabic digit dataset was collected to test the performance of the proposed system. The dataset consists of 30,000 samples collected from 300 different writers. The proposed system is divided into two main classification stages, the first stage is to recognize the Arabic digit zero and the second stage is to recognize the remaining digits. The overall recognition rate achieved is 98.73%. The paper also presented a comparison with other systems and the results show that the proposed system is more efficient.

References

[1] S. Abdelazeem. A novel domain-specific feature extraction scheme for arabic handwritten digits recognition. In *Proceedings of the 2009 International Conference on Machine Learning and Applications*, ICMLA '09, pages 247–252, Miami Beach, Florida, 13-15 December 2009.

[2] H. E. Abed, V. Margner, M. Kherallah, and A. M. Alimi. ICDAR 2009 Arabic Handwriting Recognition Competition. In *2009 10th International Conference on Document Analysis and Recognition*, pages 1383–1387, Barcelona, Spain, July 2009.

[3] J.-H. Ahn, J. Lee, J. Jo, Y.-H. Choi, and Y. Lee. Online character recognition using elastic curvature matching. In *ICAPR '09. Seventh International Conference on Advances in Pattern Recognition, 2009.*, pages 395 –397, feb. 2009.

[4] H. Alamri, J. Sadri, C. Y. Suen, and N. Nobile. A Novel Comprehensive Database for Arabic Off-Line Handwriting Recognition. In *Eleventh International Conference on Frontiers in Handwriting Recognition*, pages 664–669, Montreal, Canada, Aug. 2008.

[5] I. Guyon, L. Schomaker, R. Plamondon, M. Liberman, and S. Janet. Unipen project of on-line data exchange and recognizer benchmarks. In *Pattern Recognition, 1994*, volume 2, pages 29 –33 vol.2, oct 1994.

[6] M. Imran Razzak, S. Hussain, A. Belaïd, and M. Sher. Multi-font Numerals Recognition for Urdu Script based Languages. *International Journal of Recent Trends in Engineering (IJRTE)*, Dec. 2009.

[7] S. Jäger, S. Manke, J. Reichert, and A. Waibel. Online handwriting recognition: the npen++ recognizer. *IJDAR*, 3(3):169–180, 2001.

[8] M. Kherallah, L. Haddad, A. M. Alimi, and A. Mitiche. On-line handwritten digit recognition based on trajectory and velocity modeling. *Pattern Recogn. Lett.*, 29:580–594, April 2008.

[9] MATLAB. *version 7.9 (R2009b)*. The MathWorks Inc., Natick, Massachusetts, 2009.

[10] M. A. C. M. Mezghani, N. A new representation of shape and its use for high performance in online arabic character recognition by an associative memory. *Int. J. Doc. Anal. Recognit.*, 7:201–210, September 2005.

[11] V. Objects. *MyScript Note Edition*. Vision Objects 2010, 2010.

[12] M. Pastor, A. Toselli, and E. Vidal. Writing speed normalization for on-line handwritten text recognition. In *Proceedings of the Eighth International Conference on Document Analysis and Recognition*, ICDAR '05, pages 1131–1135, Seoul, South Korea, 2005.

[13] M. Y. Potrus, U. K. Ngah, H. A. M. Sakim, and S. A. AbdulRahman. Normalization and rectification method for online hindi digit recognition with partial alignment algorithm. In *2010 International Conference on Electronics and Information Engineering*, pages V1–223–V1–227. IEEE, Aug. 2010.

[14] L. Prevost, L. Oudot, A. Moises, C. Michel-Sendis, and M. Milgram. Hybrid generative/discriminative classifier for unconstrained character recognition. *Pattern Recognition Letters*, 26(12):1840 – 1848, 2005. ¡ce:title¿Artificial Neural Networks in Pattern Recognition¡/ce:title¿.