Online Arabic Character Recognition for Handheld Devices

Khaled Al-Ghoneim
Department of Computer Engineering,
College of Computer and Information Sciences
King Saud University
P.O. Box 300873, Riyadh 11372, Saudi Arabia[1]
Email: kagkag@gmail.com

Abstract

In this paper, a new system for classification of online isolated handwritten Arabic characters on a handheld device (Palm Pilot) is introduced. Unlike other languages, Arabic has many characters that cannot be written in one stroke. The first stroke of most of these characters is identical. That led us to design a new Arabic graffiti. A neural network was used for classification. It achieved acceptable performance in terms of error rates, and recognition speed.

Key words: Optical Character Recognition (OCR), Neural networks (NN), On-line recognition, Decision tree, Handheld devices, and graffiti.

1) Introduction

Since the invention of the computer, researchers have been working on new and improved methods of communicating with it. Faster computers and the need to make the interaction as natural as possible are driving the move from switches, to punched cards, to keyboards, and now to voice and character recognition.

The initial work in this field was concerned with off-line recognition of typed characters where the full image is presented to the system at once. No temporal information about the sequence of pixels is available. Most of the commercial systems available now can handle only typed characters and hand-printed numerals. Dealing with handwritten characters is a more difficult problem because of the greater variation of the shape of the characters resulting from writing habits, style, and the social and educational level of the writer. The problem is simplified if the characters are presented online, so that the system knows the order of the pixels.

---

[1] Personal address

As the character recognition technology matured, researchers were challenged to port it to handheld devices such as the palm pilot. These devices do not usually have enough space to have a keyboard. It was crucial that the device implements a space-saving method for interaction. The lack of space was solved using online isolated character recognition. Still, this problem proved to be more than the capabilities of the limited processor. So, researchers developed a special shorthand notation for the different letters called a graffiti, where most characters are written as usual. However, some characters which confuse traditional OCR systems are slightly modified. This forces the user to learn the new graffiti. However, since human are much more intelligent than computers, this tradeoff has been very successful in practice as evident from the wide spread use and adoption of the palm pilot. It usually takes about 15 minutes of learning for the user to adapt to the graffiti.

In this paper, we propose Arabic language graffiti. We then design and implement an Arabic online character recognition system using this graffiti. We embedded our system in a handheld device, while paying special attention to the tradeoff between recognition accuracy and speed.

This paper is organized as follows. First we discuss the design of our Arabic graffiti. Then we detail the different steps of online character recogntion systems in section 3. These steps involve data acquistion, preprocessing, and feature extraction. In section 4, we detail the results of our experiments. We performed both simulation experiments and actual speed measurements of the implemented system. Finally, our conclusions and future work are presented in Section 5.

2) Arabic Graffiti

In this section, we describe the design of our Arabic graffiti. Although Arabic is a cursive language in both printed and handwritten form, we must define an isolated letter graffiti because of the limitation of the writing area of the handheld device. The writing area consists of two rectangles. The letters are written in the left, while numbers are written on the right.

The graffiti must include all Arabic letters, numbers, and symbols. Recall that the objective of the graffiti is to make it easier for the machine to distinguish the characters. At the same time, it must be easy for the user to learn and memorize it.

In designing our graffiti, we took into consideration the relative frequency of each character. More frequent characters should be given a shorter notation to make it easier and faster for the user to write. We used a standard frequency table of Arabic characters for this purpose [4].

One distinct feature of the graffiti is that each character is written in a single stroke. As soon as the user lifts the stylus, the system processes the input stroke. Unlike other languages Arabic has many characters which can not be written in one stroke mainly because of the dotted characters. Most of the characters have two parts; dots and zigzags are called secondary, and they are located above, below, or even inside the primary part. Many characters share the same primary part.

Letters

Period
(Double)

Return   Space   Back Space   Cursor Left   Cursor Right

Right

Numbers

1   2   3   4   5   6

Zero
(Double)

7   8   9   0   '   6   Short
                                   Cut

ـ   ш   ∠   ┬   Menu   Tab
                          Cmd

(   )   ,

Punctuation shift = Tap once (•)

Symbols

'   "   -   '   ?   ¦

⊕   #   *   ™   ^   &

•   :   ;   <   >   [

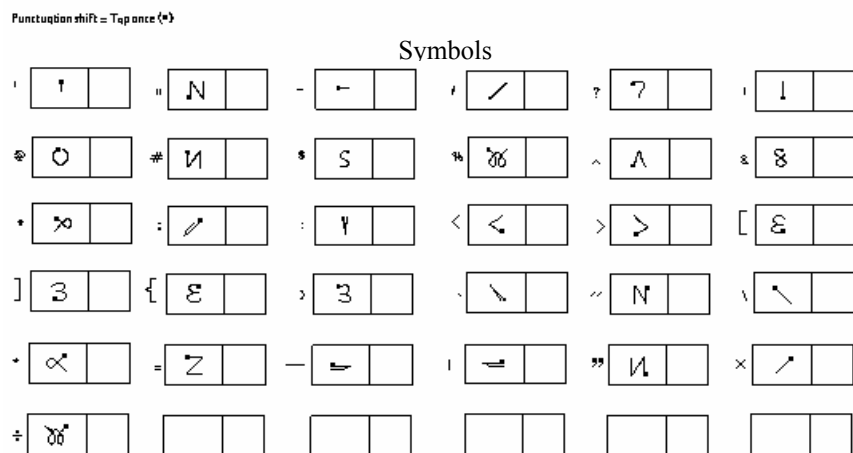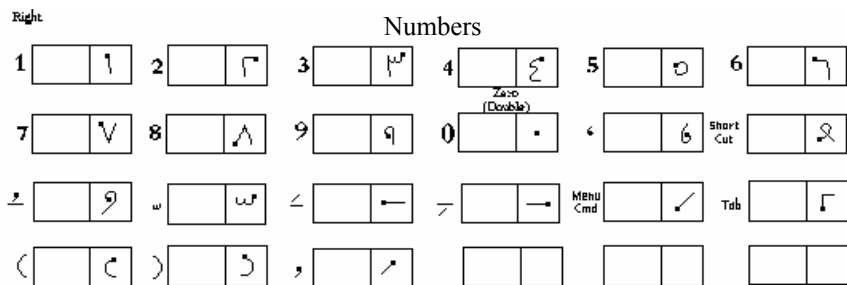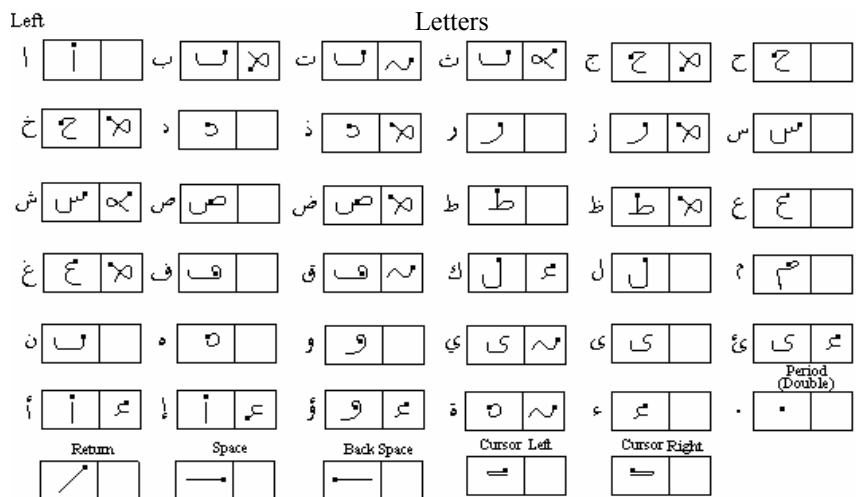]   {   >   `   ~   \

'   =   —   ¦   "   ×

÷

Figure 1: Suggested Graffiti "two phases"

We considered two approaches for the solution of this problem. The first approach is to change the shape of the dotted characters with a one stroke shape. We did not pursue this approach because it burdens the user with learning completely new and unusual shapes for some characters. We decided to use the second approach which involves recognizing the primary part first and then recognizing the secondary part. We no longer insist on recognizing the character in one stroke. However, a stroke represents either the primary or the secondary part of the character. When the user draws the stroke for the primary part, it will be immediately recognized and displayed as the undotted character. If the user then draws the secondary part, the

character will be modified to its dotted counterpart. We use the numbers area for the secondary part.

Our proposed graffiti is shown in Fig. 1. Each Arabic character is drawn as shown in the two boxes on its right. The heavy square dot in each stroke indicates the starting point for drawing the stroke. Numbers are written directly in the right box.

Note that the upper dot, lower dot, lower double dots, etc, are always drawn in the same way. Furthermore, both upper and lower dots have the same basic shape. The only difference is the direction of the stroke. For an upper dot, we start from the top, while we start from the bottom for the lower dot. This will make it easier for the user to memorize these strokes.

To distinguish symbols from letters, the user must precede them by a dot. This is the same punctuation shift used in the English graffiti, and we use the same English graffiti for the symbols.

3) Algorithm description

In this section, we describe the components of our system. Having designed the Arabic graffiti, we focus now of the design of the system. The system consists of four phases: data acquisition, preprocessing, feature extraction and classification.

The input stroke is captured and digitized in the data acquisition phase. It is then preprocessed to reduce noise and variability. Preprocessing includes smoothing, thinning, and translation of the image. Features are then extracted from the input. Feature extraction is a very crucial component of the system. We spent a lot of time optimizing the subset of features to be used. Classification accuracy and Classification speed are usually contradicting goals. The final classification step is performed using artificial neural networks (ANNs) which have been the subject of intensive research over the last decades [7,8]. These steps are detailed in the following subsections.

a) Data acquisition

We used the handheld device itself to collect both the training and testing data. A simple program was written that instructs the user to write a given Arabic character according to our graffiti. The data are stored in the device and then transffered to our development host. Using the mouse or an external tablet attached to the development host might have been a faster approach to collect the data. But, the data then will not be compatible (in terms of resolution, sampling rate, etc) with our handheld device. Our approach ensured that the data will be sampled at the correct resolution and sampling rate.

b) PREPROCESSING

The main purpose of the preprocessing phase is to eliminate as much noise as possible and to reduce data variation. Handheld devices can handle only fixed-point operations. Floating-point operations are emulated. Thus it is important to reduce the number of floating point operations required in order for the speed to be acceptable to the user. We used four steps of preprocessing, namely:

Translation: We compute the image's center of gravity[1], then translate the image such that its origin is the center of gravity.

Scaling: We scale the image so that the maximum radius for the character pixels equal to half the grid size. The radius of a pixel is defined as the length of the straight line connecting the pixel to the origin[1].

Connected Line Generation: Once the pen touches the digitizer, the generated pixels are collected in order in a chain of pixels. When the pen movement is fast, the digitizer will not be able to capture all the pixels. Some pixels will be dropped from the shape resulting in broken curve. We use Bresenham's Line Generator (BLG) [3] to fill the gaps. BLG was chosen because it requires only integer operations.

Smoothing: Finally, we smooth the input curve by inspecting each subsequence of pixels, and replacing it by a shorter version.

c) Feature Extraction

Feature extraction is one of the most difficult and important problems of pattern recognition. The selected set of features should be small, whose values efficiently discriminates between patterns of different classes but are similar for patterns within the same class.

We tried 15 classes of features. After computing these features off-line in the development host, we experimented with the contribution of each feature to the final accuracy. A trade-off between the accuracy provided by the feature and its computational requirements must be handled carefully. A smaller feature set also improves generalization. The reason is that the smaller number of inputs means a smaller number of tunable parameters (weights). In this section, we describe the final set of features that was actually implemented in the handheld device. The features are:

1. Divide the image into 25 (5x5) non-overlapping blocks. Each block has 22×22 pixels. The number of black pixels in a block represents one feature.
2. Coordinates of start and end points.
3. Order of blocks traversed, where 25 mean that the corresponding region contains the start point, whereas 0 means that the pen did not visit the corresponding region.
4. Moments. Two-dimensional moments have been used as features in a wide range of pattern recognition applications. We have used the following moments.

$$\eta_{01}, \eta_{02}, \eta_{03}, \eta_{10}, \eta_{11}, \eta_{12}, \eta_{13}, \eta_{20}, \eta_{21}, \eta_{22}, \eta_{30}, \eta_{31}$$

This set consist of 66 elements, which is still a larg set. We experimented with several combinations of these features.

d) Classification

We have three neural networks. Each one is dedicated for classifying a subset of patterns.

- *Symbols:* this is the set of characters and symbol shapes that come after the dot (the user shall tap a dot each time he/she wants to write one character from this set). The set is drawn on the left part of the writing area.
- *Letters:* this is the set of characters and control keys that can be written directly at the left side of the hand writing area.
- *Numbers:* this is the set of numbers and symbols that can be written directly at the right side of the hand writing area. This set also includes the secondary strokes.

4) Experimental results

In this section, we will describe our experimental and simulation results. We start first by describing the data set that was collected for training and testing our classifiers. Then we discuss the different experiments we performed. The system was implemented on a handheld device with limited processing power. Our concern was speed and accuracy.

For the training set, we collected about 100 samples (each sample is contains all the grafiti symbols), from 70 different persons. Also, we collected 40 samples for testing set from different people so the testing set is completely different from the training set.

The first stage in training hidden layer neural network structure is to find the suitable number of neurons in the hidden layer. When the number of neuron in the hidden layer increases, the time to get the output from the net increases, so we must chose the smallest number of nuerons that produces a good accuracy. In order to specify this number we test the accurcy (on the test data) of neural network for different number of nuroen in the hidden layer for 7 combinations of features. After these experiments, we conclude that 40 nuroens is the best number.

Table 1 represents a comparision between the linear neural network and a hidden layer nueral network according to the overall accurcy and the time of the recognition.

Combination	Start & End Point	Number of Pixles in Each Block	Order of Block	Moment	Accuracy Hidden Layer(40) LD	Accuracy Hidden Layer(40) LN	Accuracy Hidden Layer(40) RR	Accuracy Linear LD	Accuracy Linear LN	Accuracy Linear RR	Time (#Operations) Hidden Layer(40)	Time (#Operations) linear
C1	X			X	89.7	93.98	92.02	70.45	67.8	82.3	1920	512
C2	X	X			85.76	91.37	88.14	66.89	77.54	76.91	2440	928
C3	X		X		83.94	85.28	82.48	84.14	88.01	80.21	2440	928
C4	X	X	X		78.17	83.35	79.88	77.21	84.45	80.89	3440	1728
C5	X	X		X	85.52	91.26	83.86	72.83	79.62	78.96	2920	1312
C6	X		X	X	83.87	89.15	85.27	87.48	89.8	81.72	2920	1312
C7	X	X	X	X				82.76	88.05	82.43	3920	2112

Table 1: a comparison between the linear neural network and hidden layer neural network

The hidden layer NN was unacceptably slow for the online user. We were forced to use the linear NN. We used C6 for the symbols, and letters. We used C1 for numbers.

5) Conclusion

This paper describes work on online isolated Arabic character recognition using the palm pilot machine. In particular, we address the building of OCR systems

based on neural network. We made several experiments on collected data from real users and we evaluated many feature sets. We designed a graffiti that have the required simplicity, dissimilarity, and effectiveness.

Our concern in this paper was classification time in addition to classification accuracy. Several design decisions contributed to this. Mainly, we reduced the feature set size in order to improve speed, and generalization. Using integer operations only for the NN will be investigated in future research.

6) References

[1] Majid M. Altuwaijri, "A Parallel Recognition System for Arabic Cursive Words with Neural Learning Capabilities." Phd Thesis, University of Southern Louisiana, 1995.

[2] B. Al-Badr and R.M. Haralick, , 1995. *Segmentation-Free Word Recognition with Application to Arabic*, ICDAR'95: Third International Conference on Document Analysis and Recognition}, Montreal, Canada, August 14--16.

[3] Bruce Mielke, Integrated Computer Graphics, West Publishing Company, St. Paul, 1991.

[4] Ibrahim Al-qathi, "Information security and encryption" .Al-Mohandes magazine vol 8 no 2.

[5] Gyeonghwan Kim and Venu Govindaraju, "A Lexicon Driven Approch to Handwritten Word Recognition for Real-Time Applications", IEEE transaction on pattern analysis and machine intelligence, vol.19 no 4 april 1997.

[6] abuhaiba and P.Ahmed , "Restoration of Temporal Information in off-Line Arabic Handwriting", pattern recognition vol 26 no 7 pp.1009-1017, 1993

[7] Christopher M.Bishop, neural network for Pattern Recognition. Oxford University Press, Inc., 1995.

[8] Simon Haykin "Neural Networks a comprehensive foundation", Prentice Hall; 2nd edition, 1998.

[9] B. Albadr and S.A. Mahmoud, "Survey and bibliography of Arabic optical text recognition," Signal Processing, vol. 41, no. 1, 1995, 49-77.

[10] "*Basic Handbook for the Palm III organizer*" 1998.