Challenges of Handwriting Recognition in Farsi, Arabic and Other Languages with Similar Writing Styles An On-line Digit Recognizer

Homayoon S.M. Beigi, **Krishna Nathan**, **Gregory J. Clary**, and **Jayashree Subrahmonia**
T.J. Watson Research Center, IBM
P.O. Box 704 / Room J2-N41
Yorktown Heights, New York 10598
EMail: beigi@watson.ibm.com

Keywords: Handwriting, Recognition, Digit, Farsi, Persian, Arabic, Ordu, Pashtu

Abstract

This paper will emphasize the necessity of having alternative man-machine interfaces to the already established keyboard and mouse, specifically for people residing in developing countries. An on-line system is presented for recognizing handwritten Farsi (Persian) and Arabic digits. This recognition scheme is based on statistical techniques which will be briefly explained. Then, the grounds will be set for developing a full-scaled Farsi recognizer and the difficulties associated with this task are assessed.

1 Introduction

The objective of this paper is to present an on-line handwriting recognizer for written languages such as Farsi (Persian), Arabic, Ordu, Pashtu and other similar writing styles. Most of the people who use these languages reside in developing countries which are starting to get more and more interested in the use of computers to enhance their living conditions. An average citizen of the respective countries, however, does not know how to type in his/her language. In order to increase the productivity of computers for these nations, an alternative interface to the computer is desirable which would be closer to the forms of communication with which these people are familiar. Speech and handwriting are probably the most natural forms of communication since they have been present for thousands of years. If these forms of communication are used in interacting with computers, more people would be able to use computers in public applications. There are certain merits and drawbacks to both Speech and Handwriting thus comprising the reason we still communicate using both methods. One of the most obvious reasons that handwriting recognition capabilities are important for future personal computing systems is the fact that in crowded rooms or public places one might not wish to speak to his computer due to the confidentiality or personal nature of the data. Another reason is that it might be annoying to us if someone sitting next to us in the train or airplane keeps speaking to his machine. Yet another reason for the practicality of a system which would accept hand-input is that with today's technology it is possible to have handwriting recognition in very small hand-held computers, however, speech systems cannot yet be made so small as to fit in a stand-alone hand-held device. One of the most important merits of speech systems is apparently the speed of data entry; it is much easier to dictate something than to write it.

In regard to desktop computers, neither handwriting nor speech would necessarily have to replace the keyboard. They could however be very useful complementary devices to the keyboard. In pen-computers, the pen is used to do the job of a mouse in addition to its specific task of handwriting input. Use of a pen instead of the mouse avoids having to learn fancy hand-eye coordination since in most pen-computers, the user writes directly on the display device.

The authors will concentrate here on on-line handwriting recognition specifically. Recently, a lot of attention has been given to producing pen-computers which rely on the pen as their primary source of human input. A few of these machines have recently been introduced into the market-place by major companies. Among these computers some are intended to be more of Personal Digital Assistants (PDA's) while others are actual notebooks computers which have an additional pen-interface. In the PDA line, one can name the following machines as examples: Apple Newton, Canon AINote, Epson Cyber Maida, and Sony Palmtop. Full-functioned pen-computers such as IBM ThinkPad 750P and 360P, GRID and GO are also available.†

In this paper, an experimental on-line handwriting recognizer is presented which is developed for the recognition of Farsi digits. (N.B., From this point on, by Farsi, the author means to include all similarly written writing styles such as Arabic, Ordu, Pashtu, etc.) Some work is also being done to extend this recognizer to a full scale handwriting recognizer able to handle truly unconstrained Farsi handwriting. First, the theory behind the proposed on-line real-time recognition is briefly described and its application to Farsi digit recognition is discussed. Then, proposals are made for extending this recognizer to a full-scaled on-line real-time Farsi recognizer. This proposal will include a brief introduction on the difficulties of attaining this objective. Later, a few hypothetical examples are given to enhance the understanding of usefulness of such a product in developing nations. Finally, a conclusion is given on the practicality of this project and the possible final product.

2 An On-Line Farsi Digit Recognizer

An on-line real-time Farsi recognition system was developed and implemented on an IBM ThinkPad 750P^{TM} running the IBM OS2^{TM} operating with Pen Extension. The recognizer is based on the statistical system used in ThinkWrite^{TM} for conducting unconstrained handwriting recognition for the English language at IBM Research. This system is based on Hidden Markov Models (HMM's) and it has been modified to handle Farsi digits. Different components of the system are briefly described here. For a survey on on-line handwriting recognition please see [1].

In this system, the motion of the tip of the stylus (pen) is sampled at equal time intervals using a digitizer tablet and the sampled points are passed to a computer which performs the handwriting recognition. Then, the data signal undergoes a filtering process and the scribbles are normalized to a standard size and their slant and slope are corrected. After normalization, the writing is segmented into basic units and each segment is classified and labeled. Using a search algorithm in the context of a language model, the most likely path is then returned to the user as the intended string (see Figure 1).

2.1 Digitization, Preprocessing and Normalization

Most digitizer tablets have built-in low-pass filters in hardware form which take away the jagged nature of the handwriting signal. These filters in some cases generate more problem than they solve. For example, the smoothing of the data at the hardware level could sometimes smooth away natural cusps and corners which are very important in recognizing certain characters. It is sometimes more desirable to do minimal filtering at the hardware level and to leave most of the filtering to the discretion of the recognition algorithm developer. On the other hand, hardware implementations may

†These are trademarks of their corresponding corporations

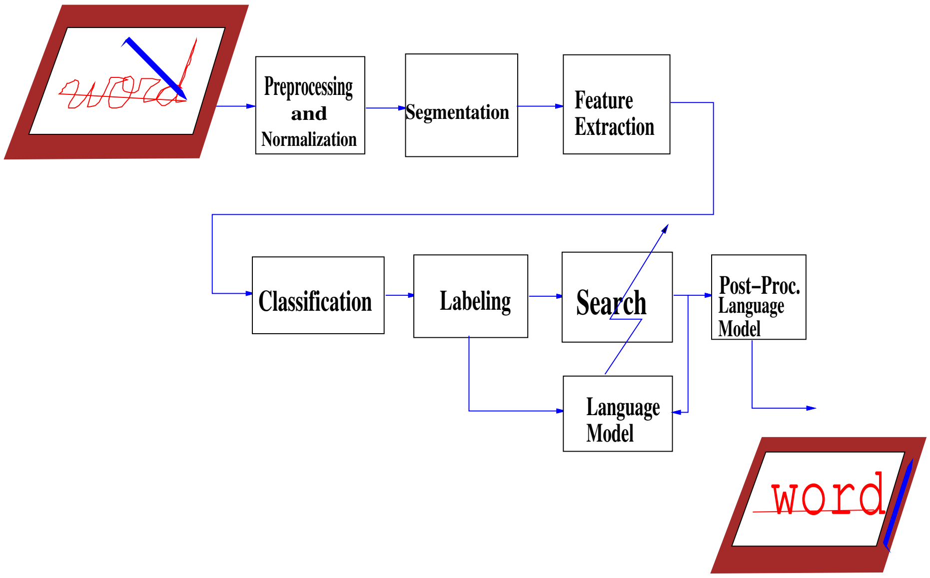

Figure 1: Generic Handwriting Recognition Process

increase the speed of this process in favor of a real-time system.

Here, the basic recognition algorithm performs best for a consistent nominal size of writing. Therefore, after some basic filtering is done on the handwriting signal, the writing is scaled to a standard height such that the overall recognition becomes size independent. To perform such normalization, the base-line (line 1) and the mid-line (line 2) of the words need to be estimated (see Figure 2). [2]

Figure 2 shows the words "principal lines" written in English and Farsi. Both of these scripts and those of similarly distributed written languages (German, French, Spanish, Italian, ..., and Arabic, Ordu, ...) possess equivalent features for performing this size normalization. These normalization techniques, however, are not applicable to the scripts of some other languages such as Japanese, Chinese, Korean, Hindi, etc. [1]

In addition to the above size normalization, slant and slope of the writing are corrected. For slant correction, mean of the slope of the dominant part of the writing is computed. The slope of the data is then offseted using the difference between the slope of the vertical axis and the computed slope. This correction is done through shearing since for small deformations shearing is a good enough approximation.

Slope correction is an iterative process which uses both of the above normalizations to estimate and re-estimate the slope of the base-line and then the data is slope-corrected by rotating it about the word's center of gravity, such that the base-line becomes horizontal. For a more thorough description of the normalization schemes used in this recognizer please see [2].

2.2 Segmentation

Figure 3 shows some sample data of handwritten Farsi digits. Each row gives a sample of the digits 1 through 9 and 0 written by different writers. Once the writing is normalized, the points are interpolated and then decimated such that the new stroke will be in terms of equidistant points in the

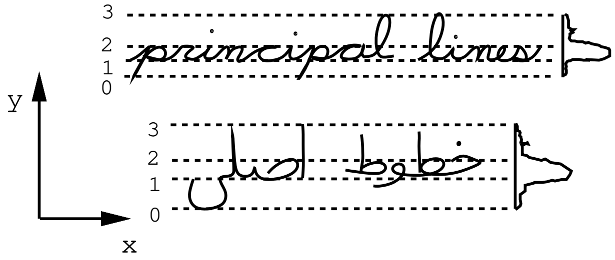

Figure 2: Principal lines of a word

Euclidean space. Geared toward the recognition of unconstrained handwriting [3], in this recognizer, segments are defined to be a subset of a character and they are generated based on points of extreme velocities (minimum x and y velocities). These segments are then sent to the core of the recognition engine for feature extraction and labeling or hypothesis generation. Centers of these segments are given by the above segmentation process and a window of equal number of equidistant points is associated with each segment. This produces a number of overlapping windows each having the same number of equidistant points.

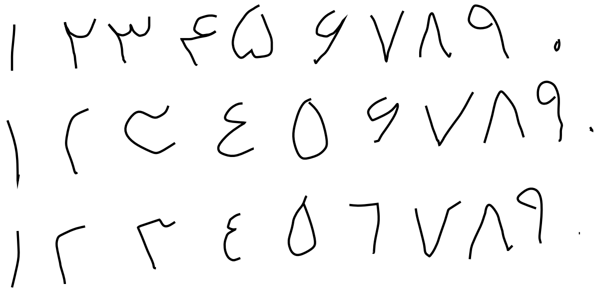

Figure 3: Sample of digits written in Farsi

2.3 Feature Extraction and Shape Classification

Once the writing is segmented into smaller units (windows or frames), these units are sent to a module which extracts features of the data, essential to the employed shape classification algorithm. These segments are sub-strokes and they carry information such as the $< x, y >$ coordinates of the points and their time-stamps which portray the speed and order information of the points.

In this implementation, the direct speed information of the data is thrown away and only the order information of the points is kept. This is done through using the equidistant points described in the previous section. In this sense, the recognizer uses mainly spatial features. [4]

The actual features, however, are the differences between adjacent coordinates (Δx and Δy), the sine and cosine of the angle that the tangent line makes with the stroke path at each point, and the the absolute y-coordinate of each point offseted by the computed baseline value from figure 2. Thus, there is a five dimensional feature vector which is associated with each of the points in the filtered signal. Since the segments are picked to have the same number of points, an augmentation of the local feature vectors will produce same length feature vectors associated with each window. Therefore, each of the overlapping windows has a long (of dimension $5x\#\ of\ points\ in\ each\ window$) vector associated with it.

These long vectors are later projected onto a lower dimensional space given by the principal components of all the data gathered in the training process. These lower (n) dimensional vectors are the feature vectors used in the recognition process. For the training data, the character label associated with each window is known. These labeled n-dimensional feature vectors are clustered in the n-dimensional space and each cluster is modeled by an n-dimensional Gaussian distribution. The means and covariances of these Gaussians are stored in the training process to be used later as prototypes for decoding.

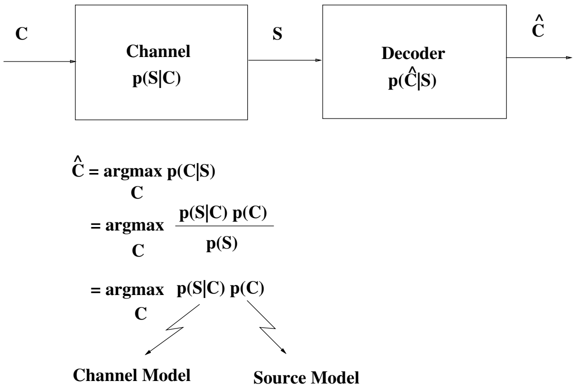

Figure 4: Source-Channel Model

2.4 The Statistical Recognition Model

Figure 4 shows a Source-Channel model for the process of handwriting recognition. The objective is to find the character which has most probably been inputted given the observation of the handwriting signal. Consider the following relation which describes the probability of any frame (window) given any valid character (digit). This relation is given by the sum of products of the probabilities of the frame given each Gaussian prototype and the probability of that prototype given the character (digit). This sum is performed over all the Gaussian prototypes which were computed at the clustering stage of the training process.

$$p(f_i \mid c_j) = \sum_k p(f_i \mid g_k) p(g_k \mid c_j) \quad where \quad g_k \sim N(\mu_k, \sigma_k)$$

Here, $p(g_k \mid c_j)$ is provided by keeping counts of the number of members of each digit c_j in each prototype g_k. $p(f_i \mid g_k)$ is computed given the Gaussian distributions modeling each prototype and the feature vector for the inputted frame in the decoding process. This model may be viewed as a single state degenerate Hidden Markov Model (HMM). A more detailed HMM model is also possible and it will provide information on the duration of staying at each state (see Figure 5).

The above model provides a set of probabilities associated with each incoming frame (window) given each of the different 10 digits in Farsi. At this stage, a sub-optimal search algorithm is used to find the most likely path which is in turn returned to the user as the recognition result. A sub-optimal search is used to make a real-time system possible.

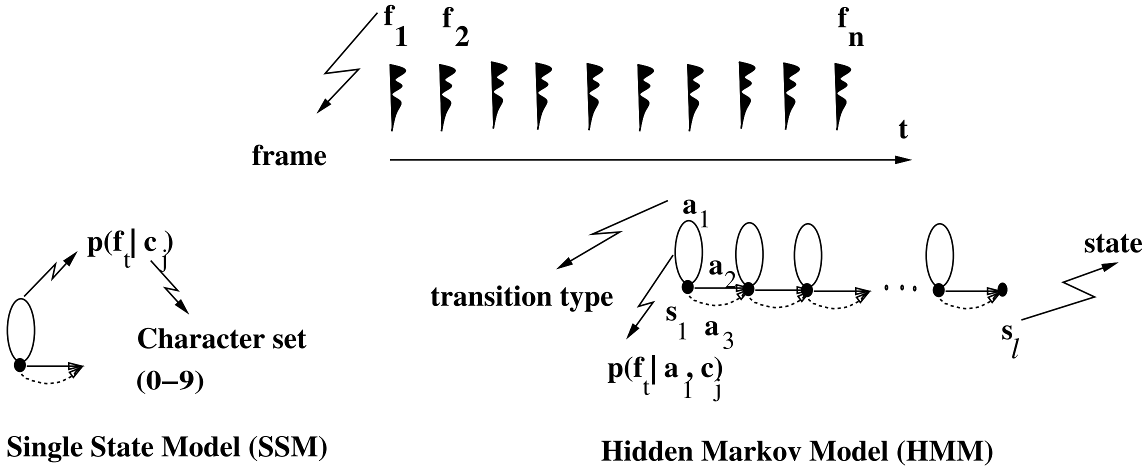

Figure 5: Hidden Markov Model for Handwriting Recognition

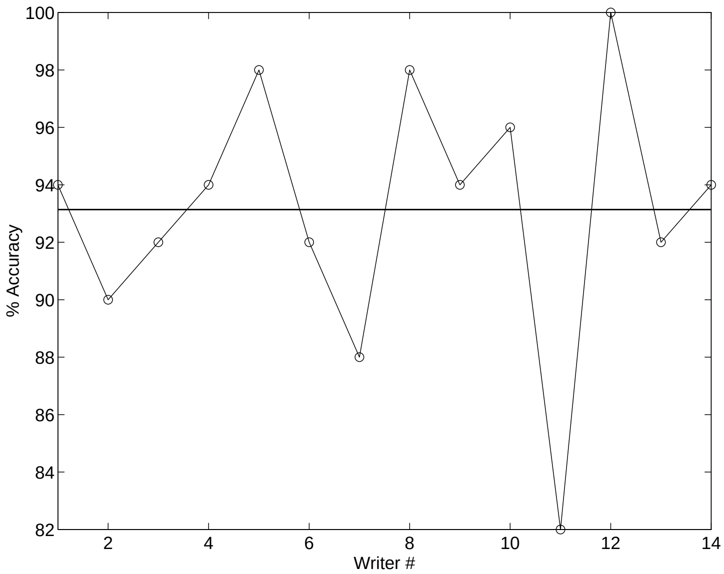

Figure 6: Recognition Accuracy of the System

2.5 Results

the system was trained on 600 samples of each digit written by native Iranian and Arab writers. This data amount of data is acceptable for a writer-independent system to be trained. The training data was generated by 20 different writers on a ThinkPad 750P^{TM} running the OS2 with Pen Extension. The system was tested by 14 writers who wrote each digit five times. This produced an average accuracy of 93.14% for these writers (see Figure 6). All writers were able to get a 100% accuracy once they got used to the system and observed some of the errors it was making. As mentioned earlier, this accuracy may be increased by using multiple-state HMM's which would model the duration of each part of the digit much better.

Figure 7: Sample Farsi Writing

3 A Full-Scale Farsi Recognizer

Unfortunately, to our knowledge there has not been any work on on-line Farsi or Ordu recognition. Some minor work has been done on on-line Arabic recognition. [5] To build a recognizer for these languages, lots of effort is needed. Farsi, Arabic, Ordu, Pashtu and other similar written languages are unconstrained by nature (Figure 7). This means that they are the most difficult styles to recognize. A feature of these languages which makes them even harder than English and other Latin style writings is the fact that dots carry a lot of information and based on the number of dots put under or above a basic structure, it could turn into different characters with totally different sounds (see Figure 8). Another basic problem is that vowels are not written in these languages. This reduces the average length of words which will in turn create more confusion and less context to help alleviate these confusions. A sentence-level language model is, therefore, essential for recognizing these forms of writing.

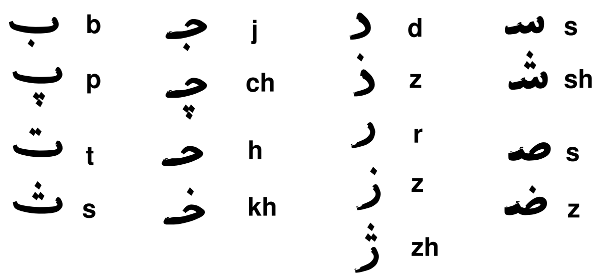

Figure 8: Some Confusing Characters in Farsi and Their Sounds

3.1 Language Models

Due to the ambiguity caused by the importance of noise-like features such as dots, very strong language models are necessary to aid the recognizer. To build such language models, lots and lots of on-line data is necessary such that models of the grammar and word N-grams could be built. There has been an effort in building stochastic models for the Arabic grammar. [6] Similar efforts should be started for other languages such as Farsi and Ordu in order to make handwriting recognition possible.

3.2 Vocabulary Size

Almost every recognition system which handles cursive and unconstrained writing, restricts itself to a given set of words at each instance. This helps increase the accuracy and the speed of recognition. However, once the number of words in an allowable dictionary gets large, keeping up with the writer in an on-line recognition system will become much harder. To alleviate this speed problem some holistic pruning algorithms have been developed which would reduce the number of legal words in a word list based on general shapes of the words. This type of pruning increases the speed of recognition. [7] These algorithms look at macro features such as descenders, ascenders, loops, etc. This is very similar to what speed readers usually do while reading.

3.3 Data Acquisition

A very important problem in developing recognition systems is usually the insufficiency of data. In the process of creating a recognizer, different types of data are required such as, a text corpus for language model generation, training data, test data, etc. This is specially hard for languages which are used in places where computers are not widely used such as in Iran, Arab countries, Pakistan, Afghanistan, etc. For these languages, it is very time consuming to generate language model or to put together a training data.

4 Conclusion

Handwriting recognition allows more efficient drafting and document generation as well as applications such as form filling and keyboardless interface with a computer. In desktop systems, the pen could be a very important complement to the keyboard for editing, marking, drawing, etc. As the handwriting recognition technology becomes more mature, applications such as longhand note-taking in the class-room are going to be closer to reality. Professionals could write their documents and have them converted to text instantly without having to go through a few iterations of having their secretary type the document for them. This is especially useful for countries such as Iran where almost all office transactions are done through handwriting and very little training would therefore be necessary for using pen-computers in those environments.

For countries such as Iran that have recently started using computers for common applications, it makes great sense to use pen-computing right from the beginning. Most of the people of these countries, although educated, cannot not type in their own language. Handwriting plays a very important role in these countries versus the western countries. According to a US Postal Service official, 83% of the mail pieces in the United States are typewritten. This figure in a country such as Iran should be very low in my intuition. Handwriting recognition in this sense is an essential simplification tool for computing in countries such as Iran.

References

[1] Homayoon S.M. Beigi, "An Overview of Handwriting Recognition," Proceedings of the 1^{st} Annual Conference on Technological Advancements in Developing Countries, Columbia University, New York, July 24-25, 1993, pp. 30-46.

[2] Homayoon S.M. Beigi, Krishna Nathan, Gregory J. Clary, and Jayashree Subrahmonia, "Size Normalization in On-Line Unconstrained Handwriting Recognition," Proceedings of the International Conference on Image Processing, Austin, Texas, Nov. 13-16, 1994.

[3] T. Fujisaki, H.S.M. Beigi, C.C. Tappert, M. Ukelson, and C.G. Wolf, "On-line Recognition of Unconstrained Handprinting: a stroke-based system and its evaluation," From Pixels to Features III: Frontiers in Handwriting Recognition, S. Impedovo and J. C. Simon (eds.), Elsevier Publishers, New York, 1992, pp. 297-312.

[4] Tetsu Fujisaki, Krishna Nathan, Wongyu Cho, Homayoon Beigi, "On-line Unconstrained Handwriting Recognition by a Probabilistic Method," Pre-Proc. of IWFHRIII, Buffalo, New York, May 25-27, 1993, pp.235-241.

[5] Samir Al-Emami and Mike Urber, "On-line Recognition of Handwritten Arabic Characters," IEEE Transaction on Pattern Analysis and Machine Intelligence, Vol. 12, No. 7, July 1990, pp. 704-710.

[6] Ayman El-Naggar, "A Finite State Automata of the Arabic Grammar," Proc. of IEEE, 1989, pp. 693-699.

[7] Sriganesh Madhvanath and Venu Govindaraju, "Holistic Lexicon Reduction," Pre-Proc. of IWFHRIII, Buffalo, New York, May 25-27, 1993, pp. 71-81.