

A Robust Free Size OCR for Omni-font Persian/Arabic Printed Document using Combined MLP/SVM

Hamed Pirsiavash^{1,3}, Ramin Mehran^{2,3}, Farbod Razzazi³

¹Department of Electrical Engineering, Sharif University of Technology, Tehran, Iran
h_pirsiavash@mehr.sharif.edu

²Department of Electrical Engineering, K.N.Toosi Univ. of Tech., Tehran, Iran
rmehran@gmail.com

³Paya Soft co., Tehran, Iran
razzazi@payasoft.com

Abstract. Optical character recognition of cursive scripts present a number of challenging problems in both segmentation and recognition processes and this attracts many researches in the field of machine learning. This paper presents a novel approach based on a combination of MLP and SVM to design a trainable OCR for Persian/Arabic cursive documents. The implementation results on a comprehensive database show a high degree of accuracy which meets the requirements of commercial use.

1. Introduction

Optical character recognition (OCR) has been extensively used as the basic application of different learning methods in machine learning literature [1, 2]. Consequently, there are also a large number of commercial products available in the market for recognizing printed documents. However, the majority of the efforts are focused on western languages with Roman alphabet and East Asian scripts. Although there has been a great attempt in producing omni-font OCR systems for Persian/Arabic language, the overall performance of such systems are far from perfect. Persian written language which uses modified Arabic alphabet is written cursively, and this intrinsic feature makes it difficult for automatic recognition.

There are two main approaches to automatic understanding of cursive scripts: holistic and segmentation-based [3]. In the first approach, each word is treated as a whole, and the recognition system does not consider it as a combination of separable characters. Very similar to the speech recognition systems, in almost all significant results of holistic methods, hidden Markov models have been used as the recognition engine [4, 5]. The second strategy which owns the majority in the literature, segments each word to containing characters as the building blocks, and recognizes each character then.

In comparison, the first strategy usually outperforms the second, but it needs a more detailed model of the language which its complexity grows as the vocabulary

gets larger. In addition, in this method, the number of recognition classes is far more than similar number in segmentation-based methods. Recently, there is also a trend toward hybrid methods which incorporates the segmentation and recognition systems to obtain overall results; these methods are usually called segmentation-by-recognition [6, 7].

One of the main concerns of designing every OCR system is to make it robust to the font variations. Thus, successful examples are omni-font recognition systems with ability to learn new fonts from some tutor. In holistic methods, as the OCR problem is considered on the whole, and the system globally uses learning mechanisms, it is easy to transform it into an omni-font learning system. On the other hand, the segmentation-based systems mainly use learning methods only in recognition process, and to the best of our knowledge, the learning systems are never used for the segmentation process in the literature [8]. Usually, human recognizes unfamiliar words by segmenting them and recognizing each character separately to understand the whole word. With this perspective, in this research, the whole task is broken down into two separate learning systems to gain from reduction of complexity in hierarchy as well as adaptability of learning systems.

The layout of this paper is as follows: Section 2 emphasizes on the characteristics of Persian script that were crucial for the design of OCR systems. In section 3, we will discuss the proposed algorithm. Segmentation and recognition modules are described in separate subsections. Section 4 presents implementation details and results. This is accompanied with conclusive remarks and acknowledgements.

2. Some Notes on Persian/Arabic Script

In this section, we will briefly describe some of the main characteristics of Persian/Arabic script to point out the main difficulties which an OCR system should overcome. As one of the main properties, the script consists of separated words which are aligned by a horizontal virtual line called "Baseline". Words are separated by long spaces and each word consists of one or more isolated segments each of them is called Piece of a Word (PAW). On the contrary, PAWs are separated by short spaces, and each PAW includes one or more characters. If one PAW has more than one character, each of them will be connected to its neighbors along the baseline. Fig. 1 shows a sample Persian/Arabic script where a represents the space between two different words, and b is the short space between PAWs of the first word which is also shown larger in Fig. 2.

In the latter figure, the first PAW on the right, comprises three characters and the second one, on the left, consists of only a single character, and denotes the pen width value which is heuristically equal to the most frequent value of the vertical projection in each line.

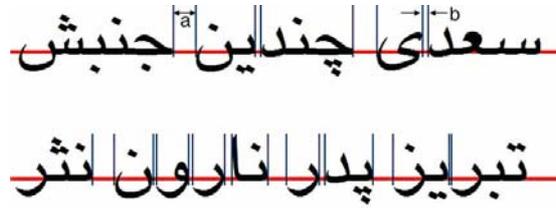


Fig. 1. Sample of Persian script and virtual baseline shown for demonstration only

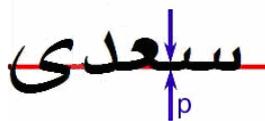


Fig. 2. An example of a Persian word consists of two PAWs

3. Proposed Algorithm

The overall block diagram of the system is presented in Fig. 3 which depicts layout analysis, post-processing, and natural language processing (NLP) subsystems in addition to recognition and segmentation blocks. The details of the NLP and layout analysis sections are out of scope of this paper and will not be discussed here.

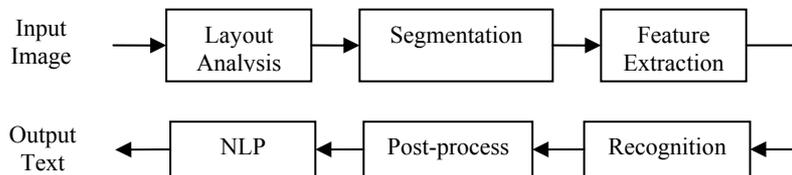


Fig. 3. Overall block diagram

In our design, we exploited the segmentation-based approach, and we considered some measures to overcome the main weaknesses of it. In addition, we examined different methods for segmentation including a system based on If-Then rules, a fuzzy inference system, and an artificial neural network (ANN) system. Finally, we concluded that an ANN approach with extended features provides the best solutions (Section 3.1). In recognition section, we obtained a definite set of features from each segmented symbol which was fed to a support vector machine (SVM) classification engine to obtain the recognized symbol. Using large margin classifiers enables us to

achieve high recognition rates which are in coherence with the best results in the literature [2].

We also decomposed each character of Persian script to more primitive symbols called graphemes. This novel decomposition has decreased the complexity of the recognition and segmentation procedures and has improved the overall result. Few different characters could share a single grapheme, and additionally, several joint graphemes could build a single character. Persian language includes many characters which the only difference they have is the number of dots and placement of them.

To finalize the character recognition task, a post-processing section is implemented to combine the result of grapheme recognition and the number of dots. Besides, this section corrects some common grapheme recognition errors using an embedded confusion matrix. Fig. 4 shows the combination of grapheme recognition and post-processing blocks with dot recognition module.

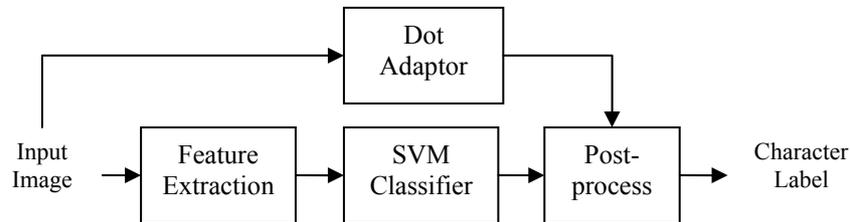


Fig. 4. Grapheme recognition subsystem is combined with dot recognition modules and post-processing blocks to recognize characters

Before proceeding further, we provide concepts of some frequently used terms in this paper for clarification:

Grapheme: In this research, we refer grapheme to any graphical image that would be a character or a part of it which acts as a fundamental building block of words. This resembles the concept of phonemes in speech, but we don't directly choose them in relation to real phonemes.

Pen tip: The vertical position of the pen in the skeleton PAW image.

Junction points: The horizontal position of the grapheme boundary. Thus, cutting the word at junction points results separated graphemes.

3.1. Segmentation

The ultimate goal of the segmentation block is to find the exact junction points in each PAW. In this research, three distinct methods have been used for segmentation of the PAWs. The decision about how segmenting a line is based on specific features which are identical for all of these methods, but the decision maker is different.

1- If Then Rule: In this method, we defined some conditional rules to compare the computed features with some predefined thresholds. The results of these conditions

have been combined to determine the junction points. We fine-tuned the rule base and thresholds by observing the test results. The following algorithm, describes how we obtain junction points through this If-Then rules.

Initially, the upper contour of the character image is extracted, and its first derivative is computed (Figures 5 and 6). In order to have robust derivatives, neighboring points should participate in the derivative calculation of each point. This is done by using the convolution of the upper contour signal and $h(t)$ which is defined as follows:

$$h(t) = \begin{cases} t & -n \leq t \leq n \\ 0 & \text{otherwise} \end{cases} \quad (1)$$

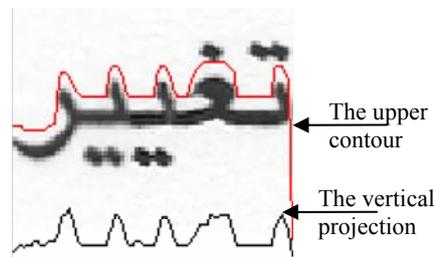


Fig. 5. Sample PAW image with contour and vertical projection

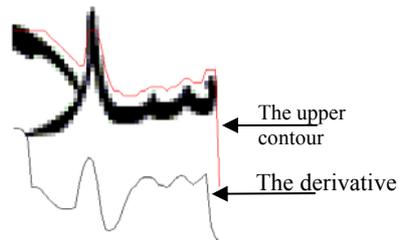


Fig. 6. Sample PAW image with contour (red) and its derivative (gray)

As it is depicted in Fig. 6, in junction points, the derivative of the upper contour has a significant peak. Hence, zero crossings of the second derivative of the upper contour are candidate junction points. On the other hand, the pen tip should be near to the baseline in the junction points; therefore, neighboring pixels of the candidate junction points are searched for black pixels near the baseline. Finally, the obtained points are the junction points.

2. Fuzzy Inference System (FIS): As an alternative to the previous method, FIS is used to determine the junction points with increased robustness. The used features in this method are as follows:

- a. Vertical projection of the line image (Fig. 5).
- b. The first derivative of the upper contour.
- c. The distance of the pen tip from the baseline.

The calculation of the first two features is evident, and we will explain the third (f_3) in more details. This feature is computed using the weighted average of the pixel values on each column of the image matrix, where the weights are chosen Gaussian functions.

$$f_3 = \frac{1}{\sqrt{2\pi}\sigma} \sum_y e^{-\frac{(y-\text{baseline})^2}{\sigma^2}} \cdot \text{image}(x, y) \quad (2)$$

where $\sigma = \frac{\text{penwidth}}{3}$

These three features are fed to an FIS which is used as a filter to create zero crossings at the junction points. Thus, applying a zero-crossing detector to the output of the FIS will result the junction points.

3. Artificial Neural Network (ANN): Previously explained systems have a short memory in the derivative calculation. Therefore, the final decision for each point depends on values of a small neighborhood of that very point. It is obvious that using a larger neighborhood can result in a more accurate decision. Hence, in the neural network method, features of a large window participate in making decision for its center point. On the other hand, this method uses a learning system to find the junction points. In this approach, similar features to the fuzzy system are calculated over a window of width equal to 4 times the pen width to make a feature vector. Resulting vector is fed to a Multi Layer Perceptrons (MLP) with one output neuron that estimates the probability of the center point being the junction point.

Our train set includes labeled junction points. The target vector of the neural network should be equal to one for the junction points and zero for the others. To assist the learning procedure, a Gaussian function with variance equal to 1/6 times the pen width is placed at each junction point. Although this smoothing reduces the accuracy, the results are quite acceptable (Fig. 7).

In practice, our neural network should have a predefined number of inputs; consequently, the input image is normalized with the pen width value in order to have pen width equal to 5 points in all images. In the implementation stage, the neural network window width is set to 20 i.e. 4 times the pen width. Thus, the neural network has 60 neurons in the input layer and 5 hidden neurons.

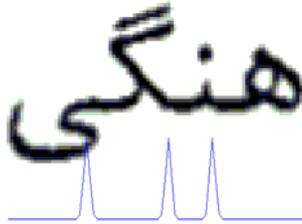


Fig. 7. Sample PAW image neural network target signal, which is used for training procedure

The described neural network uses a hyperbolic tangent sigmoid as the transfer function in all neurons, and it is trained using the standard gradient decent with momentum training algorithm which has adaptive learning rate [9]. In order to facilitate the training process and escape from the local minimums, a simulated annealing algorithm is added to the training process. The peaks of the neural network output are selected as the candidate junction points. Finally, the neighboring points of the candidates are checked for the location of the pen tip, and the final junction points are specified.

3.2. Recognition

The task of this section is to recognize the input grapheme image. In Persian script, every letter can have two to four different shapes in respect to their position in their containing PAW. The four different positions are at the beginning, in the middle, at the end, or character as an isolated word. These positions correspond to the connectivity of the letter from left, both sides, right, or no connection respectively. Table 1 shows the example of four different shapes of character "HEH". This fact is addressed in some previous works [6, 10] as one of unique characteristics of Persian/Arabic scripts which increases the literally available 52 characters of Persian script to 134 different characters in shape. The main idea to overcome this diversity is to recognize the position of characters separately and generate four different recognition systems for each position.

As Table 1 presents, we assigned a group number to each case of character shapes. In contrast to the methods available in the literature, classifiers were designed to recognize graphemes instead of characters in different positions inside words. By introducing graphemes instead of characters, the number of recognition classes reduces from 134 characters to 85 graphemes. Similarly, graphemes have four groups according to their position in the PAW. Meanwhile, a post-processing system is needed to recognize the number and positions of the dots and also the sequence of the graphemes to produce the final recognized characters. As mentioned before, this strategy helps simplifying the segmentation section and decreases the complexity of the classification process significantly.

Table 1. Four possible shapes of a character in a PAW and corresponding character group of each shape and its connectivity direction

| <i>Group Number</i> | 1 | 2 | 3 | 4 |
|-------------------------------|---|---|---|--|
| <i>Character "HEH"</i> |  |  |  |  |
| <i>Connectivity Direction</i> | Left | Both | Right | None |

Table 2. Some characteristic features for grapheme recognition

| <i>i</i> | <i>Feature Vector (F_i)</i> |
|----------|---|
| 1-7 | Moment invariant Hu features [13, 15]. |
| 8 | (The variance of the horizontal projection) / (The variance of the vertical projection) |
| 9 | (The number of the black pixels in the upper half of the image) / (The number of black pixels in the lower half of the image) |
| 10 | (The number of black pixels in the right half of the image) / (The number of black pixels in the left half of the image) |
| 11 | (The number of black pixels in the whole image) / (Area of the bounding box of the image) |
| 12 | (The width of the bounding box of the image) / (The height of the bounding box of the image) |
| 13 | (The variance of the horizontal projection of the upper half of the image) / (The variance of the horizontal projection of the lower half of the image) |
| 14 | The 2-D standard deviation of the image |
| 15-34 | The elements of the vectors that are extracted from the chain code [14] of the contour of the thinned image [12]. |
| 35-37 | Quantitative measures of curvatures of the thinned image. |

For every grapheme image, a feature vector of length 50 is computed which consists of normalized values of both statistical and structural features. The former is mainly gathered from the statistical distribution of the grapheme skeleton, while the latter is mostly related to the shape and morphological characteristics of Persian script [10, 11].

In addition to some new features specially designed for the printed script recognition, we used a number of features from our previous work on designing a recognition system for isolated handwritten Persian characters [12]. Table 2 provides some of the characteristic features extracted from binary image in brief.

We used an SVM classifier with RBF kernel and parameters listed in Table 3 which are optimized by cross-validation.

Table 3. SVM optimized parameters

| Group | Kernel | γ | c | SVs | Classes |
|-------|--------|----------|-----|------|---------|
| 1 | RBF | 0.01 | 1 | 1195 | 13 |
| 2 | RBF | 0.01 | 2.2 | 951 | 12 |
| 3 | RBF | 0.01 | 2.2 | 930 | 20 |
| 4 | RBF | 0.01 | 1 | 1664 | 40 |

4. Implementation and Results

In order to have confidential results, a comprehensive database of characters is needed, and since there was no standard dataset available for Persian script, we decided to build it from scratch. For our OCR system with segmentation-based strategy which uses learning systems in both recognition and segmentation sections, two different datasets are needed. First set should include the train and test samples of labeled PAWs for neural network which performs segmentation task and the second set should contain labeled graphemes for the SVM classifier. To achieve higher recognition rates, we decided to gather the second set based on the behavior of the neural network segmentation process. Hence, we carried out following procedures to create those two datasets.

At the first move, we designed a primitive segmentation system based on the If-Then rules, and used this system to segment about 40 pages of Persian script from daily newspapers in different font types. Since the results of such segmentation system was not satisfactory, we developed software for manually verifying the results of that primitive segmentation system. Therefore, a labeled dataset has been created for learning perfect segmentation procedure and evaluating different segmentation methods.

On the way to have complete datasets, we trained the segmentation neural network with the first set, and used this system to segment a large number of printed documents in 20 fonts to create four groups of grapheme database. Fig. 8 shows an example of some of these fonts. This dataset is also verified to have a complete multi-font labeled dataset of Persian printed documents. Our database comprises 40,000 sample PAWs for segmentation and 170,000 graphemes. The train and test sets are chosen uniformly random with ratio of 1/3.

A prototype system is implemented in MATLAB environment. The neural networks implementation is based on MATLAB Neural Network toolbox, and the SVM classification is built with the help of OSU-SVM toolbox. To speed up the algorithms and increase the efficiency, the overall system is implemented in Delphi platform. In addition, a modified version of Lib-SVM library is used in Delphi implementations.

Tables 4 to 6 provide the detailed results of our system for segmentation, recognition, and overall results.

Table 4. Correct segmentation rate

| Segmentation | Train set | Test set |
|--------------|-----------|----------|
| If Then Rule | - | 89% |
| FIS | - | 91% |
| NN | 99.4% | 98.7% |

Table 5. Grapheme classification rate

| Group | Samples | Train set | Test set |
|-------|---------|-----------|----------|
| 1 | 43521 | 99.6% | 99.5% |
| 2 | 38562 | 99.8% | 99.3% |
| 3 | 39452 | 99.8% | 99.6% |
| 4 | 48521 | 99.1% | 98.3% |

Table 6. Overall recognition rate

| Type | Train set after NLP | Test set |
|-----------|---------------------|----------|
| Character | - | 99.2% |
| Word | 95.46% | 91.09% |

**Fig. 8.** Example of a Persian script with different font types in each line

5. Conclusion

With the proposed design which uses learning systems in both segmentation and recognition sections, we have achieved a highly accurate OCR system for omni-font free size Persian printed documents. The commercial version of this system will be

exploited in Iranian Civil Organization in year 2005. This novel strategy could be extended to other cursive scripts as well with a proper training dataset.

Acknowledgement

The authors would like to thank and acknowledge the Paya Soft co. that sponsored this research.

References

1. A. Amin: Off line Arabic character recognition - a survey. Proceedings of the International Conference on Document Analysis and Recognition, vol.2 (1997) 596-599
2. Y. Lecun, L. Bottou, Y. Bengio, P. Haffner: Gradient-based learning applied to document recognition. Proceedings of the IEEE, vol. 86, no. 11, IEEE, USA (Nov. 1998) 2278-2324
3. B. Al-Badr, R.M. Haralick: Segmentation-free word recognition with application to Arabic. Proceedings of the Third International Conference on Document Analysis and Recognition, Part vol.1, IEEE Comput. Soc. Press., Los Alamitos, CA, USA, vol. 1 (1995) 355-359
4. I. Bazzi, R. Schwartz, J. Makhoul: An omnifont open-vocabulary OCR system for English and Arabic. IEEE Transactions on Pattern Analysis & Machine Intelligence, vol. 21, no. 6, IEEE Comput. Soc., USA (June 1999) 495-504
5. AH. Hassin, Tang Xiang-Long, Liu Jia-Feng, Zhao Wei: Printed Arabic character recognition using HMM. Journal of Computer Science & Technology, vol. 19, no. 4, Science Press, China (July 2004) 538-543
6. A. Cheung, M. Bennamoun, N.W. Bergmann: An Arabic optical character recognition system using recognition-based segmentation. Pattern Recognition, vol. 34, no. 2, Elsevier, UK. (Feb. 2001) 215-233
7. H. Weissman, M. Schenkel, I. Guyon, C. Nohl, D. Henderson: Recognition-based segmentation of on-line run-on handprinted words: input vs. output segmentation. Pattern Recognition, vol. 27, no. 3, UK. (March 1994) 405-420
8. R. Azmi, E. Kabir: A new segmentation technique for omnifont farsi text. Pattern Recognition Letters, vol. 22, no. 2 (2001) 97-104
9. S. Haykin: Adaptive Filter Theory. 3rd edition, Upper Saddle River, NJ: Prentice-Hall (1996).
10. M. Kavianifar, A. Amin: Preprocessing and structural feature extraction for a multi-fonts Arabic/Persian OCR. Conference on Document Analysis and Recognition, IEEE Computer Soc., pp. 213-216. Los Alamitos, CA, USA (1999)
11. B.M. Kurdy, M.M. AlSabbagh: Omnifont Arabic optical character recognition system. Proceedings of Int. Conf. on Information and Communication Technologies: From Theory to Applications, IEEE, Piscataway, NJ, USA (2004) 469-70
12. H. Pirsiavash F. Razzazi: Design and Implementation of a Hierarchical Classifier for Isolated Handwritten Persian/Arabic Characters. IJCI Proceedings of International Conference on Signal Processing, Vol. 1, no. 2, Turkey (Sep. 2003)
13. M.K. Hu: Visual Pattern Recognition by Moment Invariants. IEEE Transactions on Information Theory, Vol. IT-8, IEEE (1962) 179-187
14. R.C. Gonzalez, P. Wintz: Digital Image Processing. Addison Wisely Publishing Company, 2nd Edition (1987) 392-423