Farsi Font Recognition Based on Spatial Matching

Yaghoub Pourasad , Hushang Hassibi
Faculty of Electrical and Computer Engineering
K. N. Toosi University of Technology
Tehran, Iran
y_pourasad@ee.kntu.ac.ir
Hassibi@eetd.kntu.ac.ir

Majid Banaeyan
Faculty of Electrical Engineering
MUT University of Technology
Tehran, Iran
Majidbanaeyan@gmail.com

Abstract—**In this paper a new approach for Farsi font recognition is presented. In this approach using the Euclidean distance between spatial descriptors and gradient value in each point of boundary of some especial letters, font of a document image is recognized. To test this approach we constructed a dataset consisting of some letters in 20 widely used Farsi fonts and obtained recognition rate more than 94%. An advantage of this approach is that existence of only a few words is enough to recognize the font of document image. Another advantage of this approach is that noise doesn't disturb it. This approach is applicable on other languages, too.**

Keywords- Document Image; Font Recognition; Spatial Matching; Gradient

I. INTRODUCTION

Nowadays there are many digital libraries which store scanned documents. These scanned documents are simply presented as raw bit-maps and this leads to difficulties in document retrieval. A feasible solution for these documents retrieval is Optical Character Recognition (OCR). There are many researches on Arabic [1],[2],[3], and Farsi [4],[5],[6] Optical Character Recognition. It is obviously clear that understanding the font of text of a document image, can help us to have better results in character recognition. Font recognition can be very helpful in retrieval systems too. The field of text font recognition in document images especially in Farsi language is new and needs more attention. In [7] an approach for the recognition of Farsi fonts is proposed. In this paper font recognition is performed in line level using a feature based on Sobel and Roberts gradients in 16 directions, called SRF. SRF is extracted as texture features for the recognition. This feature requires much less computation rather than other textual features and therefore can be extracted very faster than common textual features like Gabor filter, wavelet transform or momentum features. The reported recognition rate is about 94.2% using 5000 samples of 10 popular Farsi fonts. In [8] an approach for Arabic font recognition is presented. Their proposal is to use a fixed length sliding window for the feature extraction and to model feature distributions with Gaussian Mixture Models (GMMs). The main advantage of this approach is that a priori segmentation into characters is not necessary and the authors reports performances above 99% on a set of 9 different fonts and 10 different sizes. In [9] the use of global texture analysis for Farsi font recognition in machine-printed document images is examined. They consider document images as textures and use Gabor filter responses for identifying the fonts. Two different classifiers including Weighted Euclidean Distance (WED) and Support Vector Machine (SVM) is used for classification. Authors reported average accuracy of 85% with WED and 82% with SVM classifier on 7 different face types and 4 font styles. In [10] dots of document are extracted and size of dots is estimated using weighted sum variance. Then pen width is supposed to be nearly square of dot size. But for font size estimation as writers have noticed, there isn't fixed relation between pen width and font size; therefore they assumed an approximate relation between font size and pen width. This approach is fast but only estimates an approximate value for font size and doesn't recognize the font of text of document. In [11] first, second and third order moments of the input image are used as features and correlation coefficients are used to recognize Farsi fonts.

Farsi language consists of 32 letters. These 32 letters may have at least two states depend on the location of them in a word; Because Farsi letters may connect to other letters in a word or may be disconnected of other letters. We focus on the disconnected state of these letters and assume that even in a small sentence of a Farsi text, there are some of these disconnected letters. This assumption is an acceptable assumption in Farsi texts. An interesting point in these letters is that some of them are similar to each others. It means that some of them have similar bodies but differ with existence and location of their dots. For example in letters ث and ت if we delete their dots both are in the form of ـب. Or letters ج ، خ ، چ without their dots, are ح. In these cases ح is the body of all these 4 letters. To prevent of mistake in font recognition especially in noisy documents, we do font recognition based on the body of letters after deleting their dots. For this purpose we deleted all dots of letters, therefore 18 different bodies for all 32 letters of Farsi language obtained. These 18 bodies are symbols for all 32 letters of Farsi language. Table 1 shows these symbols.

In Farsi, there are more than 500 different fonts. Developing a system that considers all these fonts is difficult and useless. Therefore we concentrate on 20 widely used fonts.

In this paper we propose a new method for recognition of Farsi fonts by using spatial matching technique. For 20 widely used Farsi fonts their 18 symbols were constructed and reserved in a dataset. To recognize font of a query letter, boundary points of query letter and dataset symbols are

extracted and spatial matching between boundary points of query letter and all dataset symbols are done. Finally, analyzing C and SSD determines the font of query letter.

Table I: 18 SYMBOLS FOR FARSI LANGUAGE LETTERS

ه	ل	ا
ن	م	س
ر	د	و
ی	ص	خ
ف	ق	ع
ط	ک	ب

This paper is organized as follows: In section 2 our method (feature extraction and matching algorithm) is presented. In section 3 experimental results are analyzed and finally in section 4, conclusion is given.

II. OUR METHOD

A. Extracting Contour points

Since edges have more changes in comparison of other part of the characters, they have more information and are used to indicate the contour points. The number of these points is usually high and reliable in different images but in order to avoid having so many contour points which cause the matching process consume a lot of time, the number of these points is decreased in this method. For this purpose a sampling process is applied on contour points to decrease them to a value which indicated with n. The sampling process should be random and also uniform. The input character and its correspondence sample points have shown in fig. 1

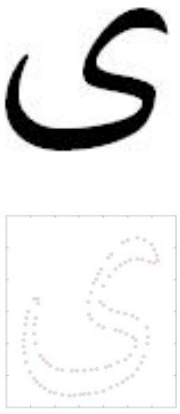


Figure 1. The input character image and n samples of contour points, where n = 100

B. Gradient of Sample Points

The gradient of a point indicates the direction of maximum intensity change in the area of its location. The gradient vector of a sample point is defined in equation

$$\begin{bmatrix} G_x \\ G_y \end{bmatrix} = \begin{bmatrix} \partial I/\partial x \\ \partial I/\partial y \end{bmatrix} \quad (1)$$

Where, I is the input image. Magnitude and direction of this vector are:

$$|G| = \sqrt{G_X^2 + G_y^2} \quad (2)$$

$$\theta = \tan^{-1}(G_y/G_x) \quad (3)$$

For each of the sampling points which are extracted from previous step, the gradient vector is computed. In this case, to reduce the computational time we use twelve intervals of gradient vectors including {0-30, 30-60, 60-90... 330-360}, which are located uniformly around a circle. Fig. 2 shows twelve intervals of gradient.

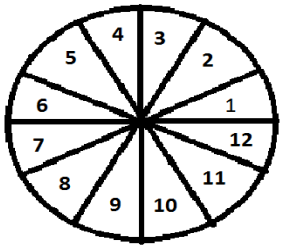


Figure 2. The twelve gradient intervals

C. Spatial Matching Algorithm

In this section we compare the image of input character with all saved images in the database and recognize which of the images in the database is corresponds to the input image. First we chose n boundary points of input image and then we compute the Euclidean distance between them and store it in a n ×n matrix (Q). The (i,j) index of this matrix indicates the Euclidean distance between points i and j. The Euclidean distance between these points $p_1(x_1,y_1)$ and $p_2(x_2,y_2)$ is described as equation (4).

$$d = \sqrt{(P(X_1) - P(X_2))^2 + (P(Y_1) - P(Y_2))^2} \quad (4)$$

To decrease the computation time during matching process, the Euclidean distance between different points is divided logarithmically to five different intervals, where the minimum and maximum distance are 1.3335 and M respectively. For example if we suppose M = 100, the logarithmic intervals are y= [1.3335 3.9242 11.54 33 100]. In this case, each Euclidean distance locates in one of the defined intervals. Therefore, for each sample point a Spatial Gradient Difference

Descriptor (SGDD) is defined. In addition to Euclidean distance, respective slope is defined between two points p (x_1,y_1) and p (x_2,y_2). Respective slope is described in equation (5).

$$\Delta G_{12} = \frac{P_{1y}-P_{2y}}{P_{1x}-P_{2x}} \quad (5)$$

As we divided gradient vector to twelve intervals, the respective slope is divided to twelve intervals, too. Each sample point is located in five Euclidean distance intervals and twelve slope intervals respect to other points. It means for each point we will have a descriptor vector SGDD with length of 60 (5 ×12 = 60), which is illustrated graphically in Fig (3).

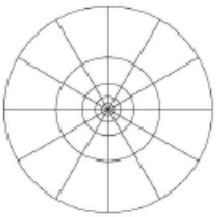


Figure. 3 The graphically representation of SGDD

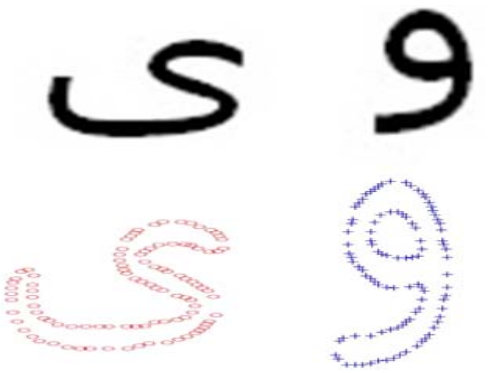
III. EXPERIMENTAL RESULTS

To evaluation our approach a dataset including 20 different widely used Farsi fonts was constructed. It should be noted that the size of letters were different to show that algorithm is time invariant. After extracting a disconnected letter of text, if it has one or more dots, its dots are deleted. Then for this letter and all dataset symbols, n boundary points are extracted; and for each point SGDD descriptor are calculated. Then SGDD descriptor vector of points of document letter is compared with SGDD descriptor vector of points of all dataset symbols. There are 2 criteria in our method that we use them in font recognition. First criterion is the value of K. K is the number of matched points while comparing boundary points of two letters. To normalize this factor, we use confidence term, C :

$$C=k/n \quad (6)$$

Second criterion is SSD. When two dissimilar letters are compared, value of C is low but comparing two similar letters leads to high value for C. Analyzing only C is enough to distinguish two different letters. For example to distinguish between dissimilar letters such as ا , ب or ک , س analyzing only C is enough. But while comparing to similar letters that are even in different fonts, analyzing only C isn't efficient enough. Because values of C even in similar letters with different fonts are close to each other whereas values of SSD are different from each other and play important role in font recognition. When we calculate SSD between query letter and a symbol of dataset, whatever SSD be lower, similarity between them is more.

While testing, observed that this approach acts successfully in distinguishing dissimilar letters. For example, spatial distribution of boundary points of dissimilar letters such as ا , ب or و , ی aren't similar therefore number of matched points are little rather than similar letters; consequently value of K and C is low. In fig. 4 distribution of boundary points of letters و and ی are showed.



Fiugure. 4 Distribution of boundary points of letters **و** and **ی**

As seen in this figure distribution of boundary points of dissimilar letters are completely different, therefore they have a few matched points thus value of C for these cases are low and mainly lower than 80%. In our approach to find font of a letter, we first consider C between query letter and all symbols of dataset. Symbols of dataset that for them C is higher than 80%, are desired candidates. To exact recognition of font, SSD is considered. In table 2, C and SSD resulted of comparing query letter و in font 'Badr', with letter و in 10 fonts of dataset is presented.

TABLE II: SSD AND C IN 10 CASES FOR LETTER "**و**"

و			
Font name	**font**	**SSD**	**C**
Tahoma	و	**0.0260**	**87%**
Ferdosi	و	**0.0269**	**95%**
Baran	و	**0.0232**	**89%**
Bardiya	و	**0.0363**	**87%**
Farnaz	و	**0.0344**	**84%**
Badr	و	**0.0084**	**97%**
Kamran	و	**0.0270**	**90%**
Mashhad	و	**0.0232**	**91%**
Mitra	و	**0.0279**	**91%**
B Nazanin	و	**0.0271**	**92%**

TABLE III SSD AND C IN 10 CASES FOR LETTER "ی"

ی			
Font Name	**Font**	**SSD**	**C**
Bardiya	ی	**0.0287**	**81%**
Farnaz	ی	**0.0319**	**79%**
Homa	ی	**0.0220**	**86%**
Lotus	ی	**0.0241**	**92%**
Tahoma	ی	**0.0134**	**97%**
Mitra	ی	**0.0236**	**94%**
B Nazanin	ی	**0.0223**	**95%**
Titr	ی	**0.0235**	**92%**
Yaghut	ی	**0.0225**	**92%**
Zar	ی	**0.0226**	**93%**

It is seen from table 2, that C for all cases is higher than 80%; value of C for 'Badr' font is highest(97%) and SSD is the lowest(0.0084); thus we can certainly say that font of query letter is 'Badr'. In table 3 query letter ی is in font 'Tahoma'. Table 3 results show that highest C (97%) and lowest SSD (0.0134) are obtained when query letter compared with letter ی in 'Tahoma' font.

IV. CONCUSION

In this paper a new approach for Farsi font recognition is presented. In this method is said that all 32 letters bodies of Farsi language are describable with 18 symbols. So, for 20 widely used Farsi fonts their 18 symbols were constructed and reserved in a dataset. This work can be done for more fonts if necessary. To recognize font of a query letter, boundary points of query letter and dataset symbols are extracted and spatial matching between boundary points of query letter and all dataset symbols are done. Finally, analyzing C and SSD determines the font of query letter. This approach presents recognition rate higher than 94% and is applicable on other languages especially languages similar to Farsi such as Arabic. A disadvantage of this approach is its low speed. This problem is a result of high computational operations.

REFERENCES

[1] M. Khorsheed, "Off-line Arabic Character Recognition a Review" Pattern Analysiss & Applications, vol. 5, pp. 31-45, 2002.

[2] N. Amara, F. Bouslama, "Classification of Arabic script using multiple Sources of information" International Journal on Document Analysis and Recognition, vol. 5, pp. 195-212, 2005.

[3] S. Ouchtati, L. Bennacer, and M. Bedda, "An Offline System for the Arabic Handwritten Words Recognition" International Review on Computers and Software (IRECOS), vol. 3. n.6 , pp. 579-585, November 2008.

[4] R. Azmi, and E. Kabir, "A New Segmentation Technique for Omnifont Farsi Text" Elsevier Pattern Recognition Letters, vol. 22 ,n. 2, pp. 97-104 , February 2002.

[5] H. Khosravi, and E. Kabir "A very large database of handwritten Farsi digits and a study on their varieties" Pattern Recognition Letters, vol. 28, issue. 10, pp. 1133-1141, July 2007.

[6] B. Parhami, and M. Taraghi, "Automatic Recognition of Printed Farsi Text" Pattern Recognition Letters, vol. 14, pp. 395 – 403, 1981.

[7] H. Khosravi, E. Kabir, "Farsi font recognition based on sobel-roberts features" Pattern Recognition Letters, vol. 31, pp. 75-82, 2010.

[8] F. Slimane, S. Kanoun, A. M. Alimi, R. Ingold, J. hennebert "Gaussian Mixture Models for Arabic Font Recognition" International Conference on Pattern Recognition, 2010.

[9] A. Borji, M. Hamidi "Support Vector Machine for Farsi font recognition" Journal of Word Academi of Science, Engineering and Technology. Vol. 28, 2007.

[10] M. H. Shirali-shahreza, S. Shirali-shahreza, "Farsi/Arabic text font estimation using dots" IEEE International Symposium Signal Processing and Information Technology, 2006.

[11] R. Mehran, H. Pirsiavash, F. Razzazi, "A Font-End OCR for Omni-Font Persian/Arabic Cursive Printed Documents" Proceedings of Digital Image Computing, Techniques and Applications (DICTA 05), pp. 385-392, 2005.