Paper

Off-line Persian Handwritten Recognition Using Hidden Markov Models

Member Ali Ahmadi (Osaka Prefecture University)
Member Sigeru Omatu (Osaka Prefecture University)
Member Michifumi Yoshioka (Osaka Prefecture University)

We present a system for recognition of Persian/Arabic handwritten scripts using Hidden Markov Models (HMMs). The text is segmented to words at first and from words to characters by an appropriate algorithm using the strokes and contour of word image. Then the feature vectors are extracted from sequential vertical frames of characters. Next, a Self-Organizing Map (SOM) is employed for clustering the features and reducing the size of inputs as well as smoothing the parameters of HMMs in classification phase. Finally, by using the HMM the characters are classified, and by concatenating the character HMMs, the word HMM is composed. The system is evaluated with five sorts of Persian handwritten data containing a number of 1,025 words, and the mean correct classification rate is 97% in word level.

Keywords: Persian, Arabic, HMM, SOM, Character Recognition, Word Recognition

1. Introduction

Until now much work has been concerned with the recognition of handwritten characters and different classification approaches are proposed. However, the performance of each of these individual classifiers does not meet the current real word applications. On the other hand, in Persian and Arabic script recognition field the progress is not satisfactory. This is mainly due to special characteristics of these languages. Persian writing which this paper addresses, like Arabic, is cursive both in handwriting and printed forms and is done horizontally from right to left. Moreover, the characters appear in different shapes depending on their position in the word. In addition, different characters may have exactly the same shape, and are distinguished from each other only by addition of a complementary sign which is usually a dot, a group of dots, or a zigzag-shape character that appears above, below, or on the baseline and is known as stress marks. This makes the recognition of Arabic characters much difficult specially in case of handwritten scripts. Further detail of Arabic scripts characteristics can be found in [1,2]. The difference between Persian writing and Arabic one, mainly comes from the number of characters (Persian alphabet has 32 main characters while for Arabic it is 28) and the way of combining characters. Different methods have been proposed on recognition of Arabic handwritten scripts either in word level or in character level. Also, various approaches were used in segmentation of cursive characters either in Arabic or English. Al-Sheik and Al-Taweel [3] assumed a reliable segmentation stage which divided letters into the four groups (initial, medial, final and isolated letters). The recognition system depended on a hierarchical division by the number of strokes. El-Emami [4] used a structural analysis to recognize postal address words after segmenting them into letters. Bouslama [5] produced a hybrid system that combined structural and fuzzy techniques. Structural analysis was used for discrimination between various letter classes, and fuzzy logic was used for discrimination of variability which exists within the same class. Alimi and Ghorbel [6] used template matching and dynamic programming to minimize errors in an on-line recognition system for isolated Arabic characters. El-Vakil [7] used stable features like number of dots, relative position of dots, and number of secondary strokes, to hierarchically reduce the number of letter class. Then a k-nearest neighbor classifier is used to determine the closest class. Alimi [8] set forth a complete system including a fuzzy radial basis function neural network that segmented letters according to their strokes which considered as 6 feature vectors.

Hidden Markov Models (HMMs) which are widely applied in recent years are qualified as an another suitable tool for cursive script recognition. Nevertheless, in Persian handwritten recognition there have been few approaches to HMMs. Dehghan [9] has presented a holistic system for recognition of handwritten Persian/Arabic words using HMM as the main classifier, and the histogram of chain-code directions of the image is used as feature vectors. Since the system is holistic, it is suitable only for limited dictionary applications.

Apart from this, HMMs are mostly used in recognition of Arabic/Persian printed texts, which basically can not be applied for handwritten script classification. Lu and Bazzi [10] have proposed an HMM based language-independent OCR which has been applied for Arabic printed texts as well. The system is based on extracting features from thin slices of line image and modeling each of characters by an HMM.

In this paper, we present a segmentation-based approach for recognition of Persian handwritten scripts which is based on assigning a discrete HMM to each character and by concatenating these sub-HMMs, the word HMM is composed. By means of this two-level approach the number of training samples available for each character is much larger than by a disjoint model for each individual word and therefore, the classification is more accurate. Our approach is similar to [11], however, it addresses the English writing, in aspect of being segmented-based and using the same way of training HMMs, but differs from it in the segmentation method and Self-Organizing Map (SOM) clustering. Our

segmentation method is well-matched to Persian characteristics, and SOM clustering is a step which we use prior to the HMM to reinforce its performance, as will be explained in follow.

The reason that HMMs usually have a good performance in recognition of cursive scripts, can be described as follow [11]: First, they are stochastic models that can cope with noise and pattern variations occurring in human handwriting. Next, the number of tokens representing an unknown input word may be of variable length. Moreover, for using HMMs there are standard algorithms for both training and recognition, which are fast and easy to implement.

In this work, at first all probable paths for segmentation of a word into characters are obtained. A new heuristic algorithm is applied to find the best segmentation paths considering the special properties of Persian characters and main rules of Persian writing. Then the features of segmented characters are extracted by using a sliding window technique to prepare a sequential form of observation for the HMM. A SOM is then employed for clustering the feature space and reducing the size of inputs. Finally, by classifying the characters and assigning the HMM to each of them, the whole word HMM will be combined. Our method is suitable for limited lexicon applications but can be used for usual handwritten scripts as well. Its main advantage is a new segmentation algorithm which makes this system much robust comparing to the holistic systems without character segmentation.

The organization of this paper is as follow: In Chapter 2 the preprocessing on the scanned text is described. In Chapter 3 segmentation of the text to words and then from words to characters is explained. Chapter 4 deals with image representation and feature extraction methods. Chapter 5 explains the SOM clustering. Chapter 6 describes the HMM classifier and the manner of training and recognition. The experimental results are given in Chapter 7. Chapter 8 concerns with discussion, and finally in Chapter 9 the conclusion is described.

2. Preprocessing

The preprocessing phase consists of the following steps:

2·1 Text digitization and noise removal The scanned text is saved as a GIF file due to its condensed format, and then is converted to a simple gray raw bitmap file which is an array of 0-1 pixels where 1 represents object and 0 background. A smoothing filter as well as a morphological closing-opening operation is used to remove the noises, which mainly appear as isolated black pixels over the background. There is some difficulty in this step due to the similarity between noises and stress marks, which are popular in Persian writing and come as isolated dots above or below of characters.

2·2 Skew correction There exists a wide variety of algorithms for skew correction in document images. Our approach is based on Hough transform [12]. After skew detection, a line segmentation process is done by using a Horizontal Projection Profile (HPP) algorithm.

3. Segmentation

Persian is cursive in the sense that words are composed of connected characters but the words are not usually connected to each other. For simplicity, in this work it is assumed that all words in the text are isolated, that is, there are recognizable spaces between the words. It can be asked from the writers to follow this rule. Moreover, we assume that there is not a significant vertical overlap between the characters within each word. As the stress marks used in Persian characters are so important in recognition procedure, it is also asked from the writers to care about putting such marks in the proper positions. Now under this assumptions we start the segmentation process which consists of two main steps: Word segmentation, and Character segmentation.

Word segmentation is done by searching for vertical gaps in the segmented lines. For this, a Vertical Projection Profile (VPP) is employed. We don't care of little skew existed in the lines, as it doesn't affect on the VPP results so much. The segmented words then are applied for character segmentation process.

The character segmentation process itself consists of following steps:

3·1 Skew correction and baseline detection In Persian writing the baseline of words usually can be detected from the Horizontal Projection Profile (HPP) with a good approximation. As shown in Fig. 1(a) the baseline has a maximum value of HPP. Usually, the baseline of word is near to the horizontal axis with little derivation. So by rotating the image with small degree of $\pm\theta$ and looking for the maximum value of HPP on the horizontal axis the baseline will be estimated and the skew image is corrected. In case of not a maximum value of HPP the middle line of cadre is considered as the baseline (See Fig. 1(b)).

3·2 Semi-word detection In Persian usually a word can be divided into smaller units called semi-words. Semi-word is a isolated part of word consisting one or more characters. Since determination of semi-words is so helpful for an accurate character segmentation, we do this step at first. For this purpose all connected components within an isolated word must be determined. The image of the word is considered as a 2-

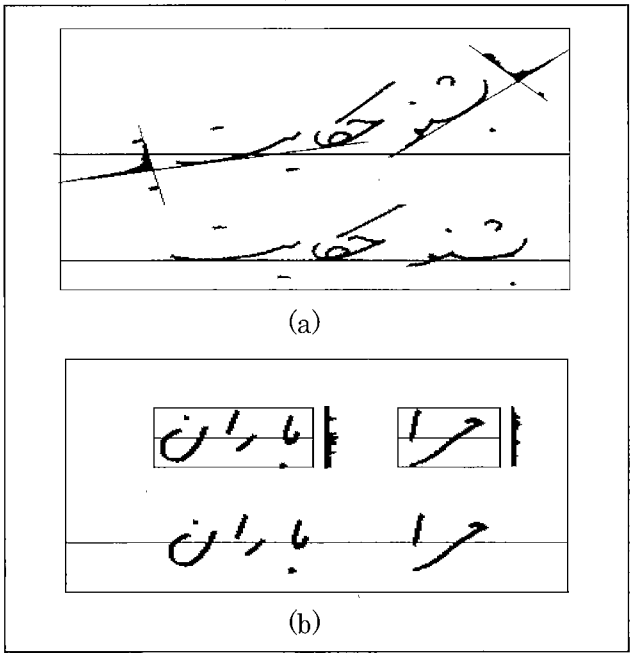


Fig. 1. Detection of word baseline. (a) Finding the line of maximum value of horizontal projection profile and then rotating word image within $\pm\theta$. (b) Middle line of image cadre is considered as the baseline.

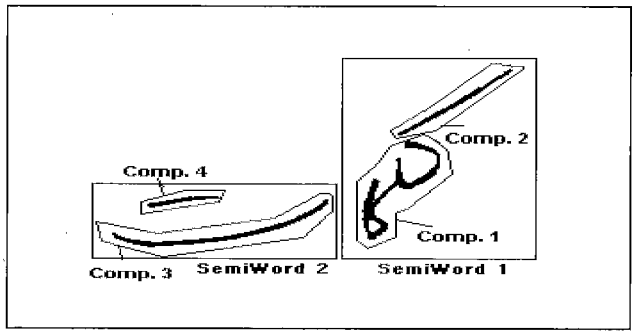


Fig. 2. An example of connected components and semi-words within a word.

dimensional array of black and white pixels then a connected component is a sort of connected black pixels. The top-right-most black pixel of region is considered as the start point from where we look for all connected neighbor black pixels successively. All found pixels are labeled and the process will continue until all connected pixels of this region will be labeled. Figure 2 shows connected components of a word.

A semi-word is usually consisted of a big connected component together with one or more small components as stress marks. For example, in Fig. 2 there are two semi-words, one is made with components 1 and 2, and another one is made with components 3 and 4, while the components 2 and 4 are stress marks. As mentioned before stress marks are so important in recognition of Persian characters as many characters are the same in the main components and differing only in stress marks. So one of our purposes in this step is to assign the stress marks to correct semi-words. In our algorithm stress marks are recognized from small size, distance to baseline, and less density in black pixels. The corresponding semi-word is found by measuring the vertical gaps within a word. The recognized semi-words of a word then should be put together again so that there are no vertical overlaps between them.

3·3 Thinning and stroke detection Thinning is done by removing all the boundary points to obtain the main skeleton of image and thereof the main strokes of the word is detected. These strokes associated with an image contour make us to have an accurate character segmentation. Thinning must not cause any disconnection between connected parts of the image. The algorithm used here is the same as described in [13]. The removing process is performed in several steps: At first, the points with no east neighbors will be removed. Next, those with no west neighbors, then those with no north neighbors and finally those whose south neighbors are zero will be removed. This will continue until the process makes no changes on the image.

The thinned image is then used for detecting the strokes and loop. The right-most point of each connected component is considered as a start point and moving ahead. All points located before a deviation angle whose value is over than a threshold, would be considered as a stroke region and approximated by a mean line which is the corresponding stroke. This procedure will continue until all points of word are processed.

During stroke extraction, we check the skeleton graph for loops. This is needed because loops carry important discriminatory information for recognition and usually each loop can be considered as the possible segmentation point.

It must be mentioned that we don't care of stress marks in the stroke and loop detection process. The result of stroke and loop extraction from Fig. 2 is shown in Fig. 3.

3·4 Contour detection To have a more accurate character segmentation we need the contour of word image to be employed together with strokes. Figure 4 shows an example of extracted contour.

3·5 Character segmentation process In this step we use both the strokes and the image contour of the word to generate two possible segmentation paths.

In the segmentation based on strokes, the points in which a new stroke starts, are considered as segmentation points.

Moreover, as it is shown in Fig. 5 each of the following appearances is a possibility for a segmentation point:

-loop

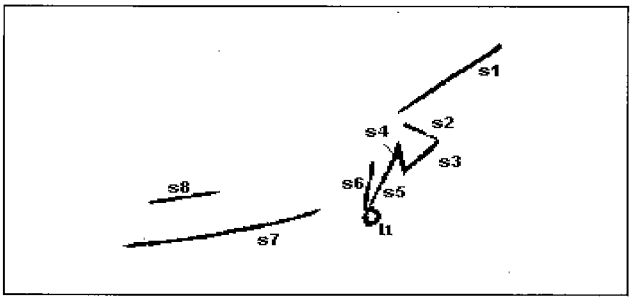


Fig. 3. Extracting the strokes and loops within a word.

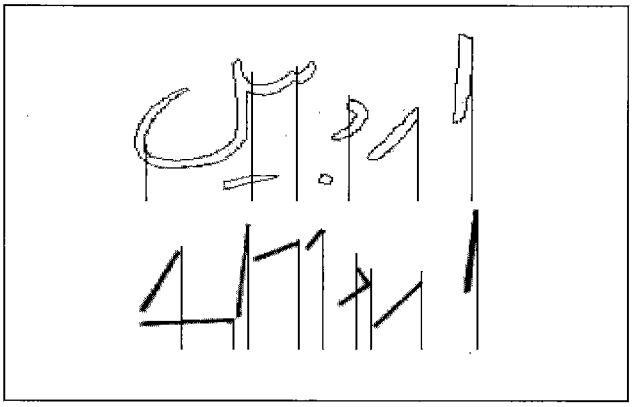


Fig. 4. Character segmentation based on contour and stroke map.

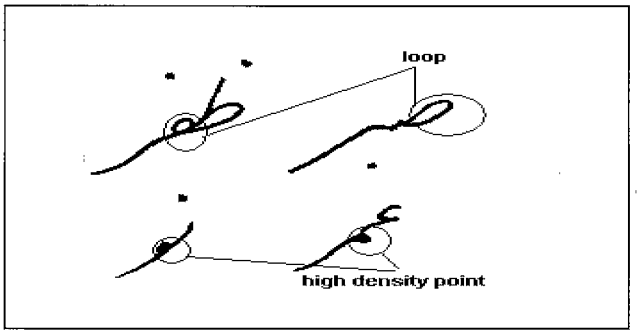


Fig. 5. Loop regions and high density points within a word are candidates for a segmentation point..

-high density of black pixels.

In the segmentation based on the image contour, we trace the chain code of image contour and detect the slop direction. The area in which the upper contour has a change in slop is considered as a possible break point (Fig. 4).

Then as it will be explained next, among these two paths, the one which gives minimum junk characters in classification stage, will be chosen as the final path.

4. Image Representation and Feature Extraction

To prepare a sequential form of observation for the HMM classifier we convert the word image to a sequence of vertical frames. A simple vertical slicing method is employed to segment all characters into a right-to-left sequence of overlapping vertical frames (Fig. 6). To normalize differences in size, each frame is computed as a narrow vertical strip whose height is equal to core-height of the character and width is relative to height ($0.2h$). The overlap between two frames is a system parameter currently taken to be 50% of the frame width. Depending on the character, the number of frames per character usually varies between 1 and 15. Each frame is then vertically divided into 15 equal overlapping cells.

After generating the frames, the features of each frame are extracted. The main goal of this step is to capture the most relevant and discriminant characteristics of the character to recognize. The main features we compute for each frame are: i) intensity (percentage of black pixels within each cell)

ii) vertical derivative of the intensity (across vertical cells)

iii) number of transition between black and white pixels in the vertical direction

iv) center of gravity of the frame (distance to baseline)

In this way each frame is represented as a 32-dimnesional feature vector.

5. SOM Clustering

The main aim of SOM clustering is quantization of feature vectors to limit the number of observation symbols in the discrete HMM. Moreover, since the amount of training data is usually limited, some kinds of parameter tying are required for the HMM classifier. Therefore, the SOM is employed as it preserves the neighborhood property of feature vectors across the result map which could be used for smoothing the parameters of the trained HMM as explained in next Chapter. By using the SOM we transform the multi-dimensional space of features to a 2-dimensional space of map in which each feature vector is represented by a position index (row, column) of a node which is the closest code book to the vector according to Euclidean distance. Thus, the HMM input will be a sequence of 2-dimensional vectors. The SOM nodes are used as the basis of the probability density functions (pdfs) of the states in HMMs.

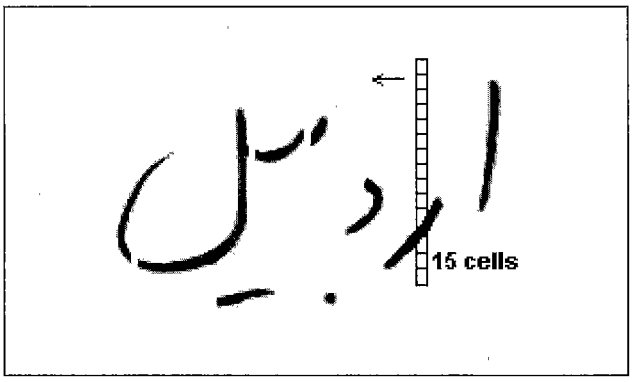


Fig. 6. Dividing each character into frames and each frame into equal overlapping cells.

Table 1. Parameters of SOM model.

Item	Value
Map size:	9x9
Topology:	Rectangle
Neighborhood:	Bubble
Learning:	Linear
Initial radius:	10

There are two kinds of approaches: One is using small separate SOMs for each category of feature vectors. This is more accurate and covers all different variations of input vectors. The second way is building a large SOM covering all feature vectors discarding their category. This is faster and more practical, however, it may not be so accurate to sense all vector variations. In this paper we use the latter way for simplicity. So a map of 9x9 units is used to cluster a number of 160,000 feature vectors extracted from all character image frames. The parameters used for the SOM program [14] are as shown in Table 1.

6. Hidden Markov Model Classifier

HMMs are models for sequential stochastic data. An HMM formally is defined a triple $\lambda = (\pi, A, B)$ [15] where,

$\pi_i = P(q_1 = S_i)$

$a_{ij} = P(q_t = S_j \mid q_{t-1} = S_i)$

$b_{j,k} = b_j(v_k) = P(O_t = v_k \mid q_t = S_j)$.

The number of states for each HMM considered as N and depends on the minimum frames extracted from the character. The number of distinct observation symbols per state is M which is equal to the SOM size (9x9), so in the above equations:

$1 < i, j < N$ and $1 < k < 81$.

As mentioned before we consider a separate right-to left linear HMM for each segmented character and then by concatenating these sub-HMMs the main HMM for whole word is combined. The feature vectors extracted from the image frame are used as sequential observation for the HMMs. We consider totally 84 classes for different types of Persian characters including different forms of all letters as well as digits. Therefore, 84 sub-HMMs are taken for classification. Figure 7 shows a simple schema of HMMs for characters and words. It should be noticed that usually we take two HMMs per character (one for the upper case and one for lower case) regardless of the number of different occurrences of a characters within a word or in the dictionary but in some cases since the different forms of a character are quite different in shape and size (for example, ع ـع ـعـ عـ), a number of 3 or more models are taken. Since the number of frames in different samples of a character is variable, usually we set the number of sub-HMM states equal to the minimum number of feature vectors that can be expected for a character. It could be happened that a sequence of

observation symbols is rejected by the correct HMM because the number of symbols is less than the number of states since the sub-HMMs are supposed to be linear straight-forward type.

In Table 2, the number of states for models corresponding to the main Persian characters is given.

Preceding to recognition, all sub-HMMs are trained independently by *Baum-Welch algorithm* [15] to find the model parameters which maximize the probability of the observation sequences. The segmentation algorithm described in Chapter 3, gives all probable paths for segmenting the characters. The best one is selected regarding to minimum number of junks within the segmented parts. Then the characters are labeled manually to one of the 84 classes. The feature vectors extracted from all image frames are used as the input of the SOM to make the clustering map. According to the fact explained in Chapter 5 a sequence of 2-dimensional SOM codebooks (nodes), representing a sequence of feature vectors are used as sequential observations for training each character HMM. Next to training the sub-HMMs, the HMM for each word of the dictionary is composed by concatenating these submodels and is trained in the same way.

Finally, in the recognition phase again we use segmentation process but just to remove the vertical overlap between characters within the word. Then a sequence of feature vectors clustered by the SOM and corresponding to all letters of the word is inputted to each word HMM which has been already created in the training step. If λ_l is the HMM corresponding to ℓ -th word of dictionary and F presents the observation sequence of the unknown word, then F is classified as the dictionary word ℓ with the maximum a posteriori probability $p(\lambda_l | F)$. *Viterbi algorithm* [15] is used to compute this probability.

Basically, the main problem in the HMM parameter estimation is limitation of training data which causes the recognition rate become degraded with even a slant variation in test data. To overcome this problem, an appropriate smoothing of estimated observation probabilities of the word HMM is applied based on the neighborhood information preserved in the SOM codebook. Our approach is similar to [9] but differs in the neighborhood function. After determining the word HMMs in training phase, the value of observation probabilities in each state will be adjusted by the probabilities of neighboring nodes on the SOM as follow:

$$b_j^{new}(k,l) = b_j^{old}(k,l) + \sum h_{(k,l)(p,q)} b_j^{old}(p,q)$$

where $h_{(k,l)(p,q)}$ is a function of the distance between two nodes (p,q) and (k,l) in the map defined as

$$h_{(k,l)(p,q)} = s \exp\left(\frac{-d^2_{(k,l)(p,q)}}{\lambda}\right).$$

Here, λ is a parameter proportional to neighborhood density around each node but for simplicity chosen to be 100 in all cases, s is smoothing factor, and d is distance between two nodes with the coordinates *(k,l)* and *(p,q)* on the map.

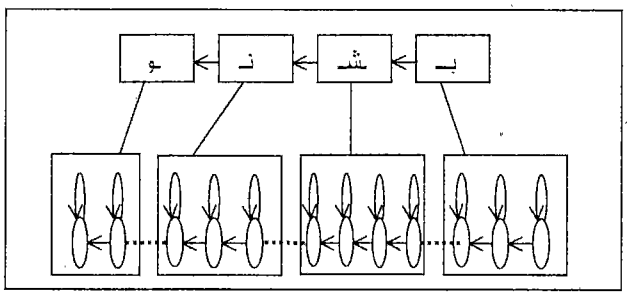


Fig. 7. Simple graph of model building. The word HMM is built from concatenating of character models in the training.

Table 2. The number of states for each letter model.

Letter model	No of states	Letter model	No of states	Letter model	No of states	Letter model	No of states
ا	1	خ	4	ص	5	ک	5
ب	3	د	2	ض	5	گ	5
پ	3	ذ	2	ط	3	ل	3
ت	3	ر	2	ظ	3	م	3
ث	3	ز	2	ع	3	ن	3
ج	4	ژ	2	غ	3	و	2
چ	4	س	3	ف	4	ه	4
ح	4	ش	4	ق	4	ی	4

7. Experimental Results

In order to train and evaluate our system, we use five sets of Persian handwritten texts. A number of 5,125 images of 205 city names of Iran, written by five writers were gathered. We asked the writers to write each of city names five times so for each city name we had totally 25 samples. To limit the variation in writing, we asked the writers to adhere as closely as possible to standard writing style, and not use complicated writing manner. Also they had been asked to use the usual size for the words. A sample of data written by one of the writers is shown in Fig. 8.

In order to have more effective learning, the K-fold algorithm was used. The entire data set is randomly divided into a training set (80%) and test set (20%) five times and for each time the HMMs are trained and evaluated by the corresponding data sets. Thus, for each word we had statistically a number of 100 (25x5x0.8) training and 25 (25x5x0.2) test samples. Also for each character according to the number of occurrences within the words we had different numbers of samples, for example, in case of " ا " the number of occurrences within the words is 58 so the numbers of training and test samples are 5,800 (58x100) and 1,450 (58x25), respectively. In case of character " ب " with 7 occurrences these values are 700 and 175. The SOM program was used for clustering of more than 160,000 feature vectors extracted from vertical features. At first a copy of whole data set was applied and after character segmentation, the HMMs for

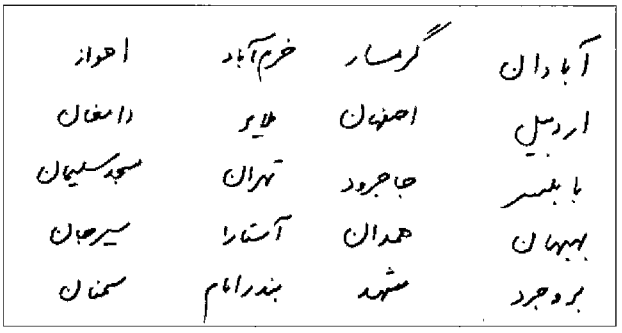


Fig. 8. City names of Iran written by one of the writers.

characters and thereby the word HMMs were built. Then the above training and test data sets were used for training and evaluating the word HMMs. The recognition rate is calculated by averaging the classification results in different sorts of test data as shown in Table 3.

If we suppose that correct classified words are among the top k candidates (two or five) then the recognition rate can be more increased. Moreover, as mentioned before, to overcome the problem of insufficient data for training, we smoothed the HMM parameters by a smoothing factor (s) based on the neighborhood information of the SOM which has been already described in Chapter 6.

Table 3. Results of recognition rate for test data sets (%).
(allowing correct word be only the **first** candidate)

Writer	Test Set A	Test Set B	Test Set C	Test Set D	Test Set E	Avrg.
1	44.9	47.1	47.2	41	49.9	46.02
2	48.21	53.21	50.13	55.1	54.2	52.17
3	43.62	37.13	65.7	34.26	67.9	49.72
4	59.93	49.35	55.26	52.19	50.9	53.52
5	52.2	49.21	53.21	56.7	57.6	53.78
Avrg.	49.7	47.2	54.3	47.85	56.1	51.03

$$b_1^{OLD} = \begin{pmatrix} 0.010 & 0 & 0 & 0 & 0 & 0.010 & 0.020 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0.075 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0.464 & 0.421 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

$$b_1^{NEW} = \begin{pmatrix} 0.010 & 0 & 0 & 0 & 0 & 0.011 & 0.020 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0.095 & 0.020 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0.474 & 0.431 & 0.006 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0.014 & 0.014 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

Fig. 9. The observation probability values for state one of HMM represent for word اردبیل before and after smoothing (smoothing factor s=0.04)

Table 4. Results of recognition rate for test data sets after smoothing (s=0.04, allowing correct word be among the two candidates)

Write	Test Set A	Test Set B	Test Set C	Test Set D	Test Set E	Avrg.
1	86.15	87.1	92.7	93.14	94.9	90.8
2	90.91	93.1	91.1	97.21	93.7	93.2
3	87.6	82.1	96.2	92.61	96.9	91.08
4	95.14	89.6	93.1	97.1	90.61	93.11
5	90.7	92.15	95.4	97.14	94.64	94
Avrg.	90.1	88.81	93.7	95.44	94.15	92.44

Table 5. Recognition rates before and after smoothing
(Rec(n) means that correct word is among top n candidates)

Top-k	Rec(1)	Rec(2)	Rec(5)
Average Rec. Rate (before smoothing)	51.03	60.10	80.75
Average Rec. Rate (smoothing with s=0.01)	80.9	85.1	92.4
Average Rec. Rate (smoothing with s=0.04)	84.1	92.44	97.5

A sample of smoothed parameter values (observation probability data b_j) for the HMM related to city word (containing 6 sub-HMMs and totally 12 states) is shown in Fig. 9. Also the new results of recognition obtained after this smoothing procedure are shown in Table 4 where s is taken 0.04. Finally, Table 5 shows an overview on all recognition results. As it can be seen the recognition rates are much improved after smoothing.

8. Discussion

As we explained before, our system is proposed for a limited variation vocabulary. However, the writers were asked to follow a sort of standard in writing, the differences in writing styles were quite significant. A detailed error analysis done on the misclassified cases, showed that the reason of misclassification mainly comes from the complex and specific characteristics of Persian writing. For example, as we mentioned before, the stress marks and their mispositions cause the most misclassification cases, though we tried to reduce this rate by the proposed segmentation method. Table 6 shows the results of error analysis on the experimental data. As a solution to decrease the misclassification rate of system still more, we guess that a scale normalization of the word images would be helpful. Moreover, currently we consider a total number of 84 sub-HMMs for all characters. But it seems that by increasing this number and considering more HMMs for different occurrences of each character, the misclassifications risen by the character variation would be decreased, although it would make the system more complex. On the other hand, the number of states for each character HMM in Table 2 is taken through trial and error considering the shape characteristic of character. Normally, increasing the number of states in the HMM heightens the accuracy but due to the speed limitations we tried to select the minimum number of states for each model.

Actually, it is hard to give a comparative result with all other Persian/Arabic handwritten recognition works because of few works and unavailability of such systems. But comparing to the methods already described in the introduction of this paper, we can claim that our proposed method is free of the main common shortcoming of such systems ,i.e., case-dependence or in other term no generalization. That is, the proposed system can be used confidentally in a dictionary with specific number of handwritten words giving an acceptable recognition rate, and due to the robust segmentation method we employed, the classification result is much accurate comparing to holistic methods without the segmentation step. For example, compared to the work [9] which is a similar work on a limited vocabulary of Persian words but without segmentation, our method shows an increasing in average recognition rate almost more than 10%.

Table 6. The misclassification analysis of test data.

Misclassification Reason	No of Occurrences	Percent
Stress marks ـً ⇒ ـَ	60	37%
Similarity between characters ب ⇒ ر	45	27.5%
Variation in word length	12	7.5%
Lengthening character ـــــ	31	19.2%
Others	14	8.8%
Total	162	100%

9. Conclusion

In this paper, we have presented an off-line HMM based system for recognition of Persian handwritten texts which is suitable for using in limited vocabulary of data. Using a sort of good quality data obtained from five different writers, we could get a recognition rate of 92% in word level which is quite satisfactory regarding to complex characteristic of Persian scripts like cursiveness and dependence of characters to stress marks. Although, still there are many open problems like large variance in data or overlaps between characters or also substitute errors caused by stress marks and similar characters, but due to the high potential of the HMM classifier and the proposed segmentation method, we believe that the system can be improved for using in recognition of large vocabularies with more variations in writing manner.

(Manuscript received February 15, 2002, revised August 22, 2002)

References

(1) M.Fakir, M.M.Hassani, and C Sodeyama: "Recognition of Arabic Characters Using Karhunen-Loeve Transform and Dynamic Programming," *Proc. IEEE Int'l Conference on Systems, Man & Cybernetics*, Vol VI, pp.868-873, Tokyo, Japan (1999)

(2) B.Al-Badr, S.A Mahmoud:"Survey and Bibliography of Arabic Optical Text Recognition," *Signal Processing*, Vol. 41, pp.49-77 (1995)

(3) T.S.El-Sheik, and S G El-Taweel: "Real-Time Arabic Handwritten Character Recognition" ,*Pattern Recognition*, Vol. 23 , No 12 , pp.1323-1332 (1990-12)

(4) S.Al-Emami, M.Usher. "On-Line Recognition of Handwritten Arabic Characters", *IEEE Transactions on Pattern* Analysis and Machine Intelligence, Vol. 12, No. 7, pp.704-710, 1990 (1990-7)

(5) F.Bouslama and A.Amin: "Pen-based Recognition System of Arabic Character Utilizing Structural and Fuzzy Techniques", *Proceeding of Second Int'l Conf. on Knowledge-Based Intelligent Electronic Systems*, pp 76-85, Australia (1998-4)

(6) A.Alimi and O.Ghorbel: "The Analysis of Error in an On-Line Recognition System of Arabic Handwritten Characters", *Proceedings ICDAR'95*, pp 890-893, Montreal, Canada (1995-8)

(7) M.El-Vakil and A.Shoukry: "On-Line Recognition of Hand-written Isolated Arabic Characters", *Pattern Recognition*, Vol 22, No. 2, pp.97-105 (1989-2)

(8) A Alimi: "An Evolutionary Neuro-Fuzzy Approach to Recognize On-Line Arabic Handwriting", Proceedings of the 4th Int'l Conf. Document Analysis and Recognition (ICDAR '97), pp.382-386 (1997)

(9) M Dehghan, K.Faez, M.Ahmadi, and M.Shridhar. "Holistic Handwritten Word Recognition Using Discrete HMM and Self-Organizing Feature Map," *Proce. IEEE* Int'l *Conference on Systems, Man & Cybernetics*, pp. 2735-2738, Nashville, USA (2000)

(10) Z.Lu, I.Bazzi, A Kornai, and J.Makhoul: Natarajan P., Schwartz R., "Robust language-independent OCR system", *Proc. SPIE*, Vol. 3584, pp 96-104, 27th AIPR Workshop (1999)

(11) H.Bunke, M.Roth, and E.G.Schukat-Talamazzini: "Off-line Cursive Handwritten Recognition using Hidden Markov Models," *Pattern Recognition*, Vol. 28, No. 9, pp 1399-1413 (1995-9)

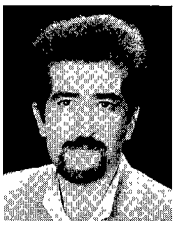
(12) R.Duda and P.Hart: "Using of the Hough Transform to Detect Lines and Curves in Pictures," *Commun. ACM*, Vol. 15, No. 1, pp.11-15 (1972-1)

(13) A.Rosenfeld and A.C.Kak: "Digital Pictures Processing", Vol. 2, pp.232-240, Academic Press, 2ed (1982)

(14) T.Kohonen, J.Hynninen, J.Kangas, and J.Laaksonen: "SOM-PAK: The Self-Organizing Map Program Package, Version 3.1" Helsinki University of Technology, Finland (1995)

(15) L R Rabiner: "A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition," *Proce. IEEE*, Vol. 77, No. 2, pp.257-286 (1989-2)

Ali Ahmadi



(Member) received B.Sc. in Electrical Engineering from AmirKabir University, Tehran, Iran and M.Sc. in Image Processing from Osaka Prefecture University, Japan in 1991 and 2001 respectively. He was working in field of computer programming and system designing from 1992-1998. Now he is studying in the field of Image Processing and Neural Networks as PHD candidate on Osaka Prefecture University. His research interests are Image Processing, Text Processing, Pattern Recognition and Neural Networks.

Sigeru Omatu

(Member) received the B.E. degree in Electrical Engineering from the Ehime University in 1969 and the M.E. and Ph.D. degrees in Electronics Engineering from Osaka Prefecture University in 1971 and 1974, respectively. He was with the Tokushima University as a research associate since 1974, a lecturer since 1975, an associate professor since 1980, and a professor since 1988. Then he joined the Osaka Prefecture University in 1995, where he is currently a professor in the Graduate School of Engineering.

Michifumi Yoshioka

(Member) received the B.E., the M.E, and Ph.D. degrees in Geo System Engineering from University of Tokyo, Japan, in 1991, 1993, and 1996, respectively. Since 1996, he has been a research associate and from 1998 lecturer at Osaka Prefecture University. His current interests center on image processing methods using neural networks.