Abstract—Most of the countries use bi-script documents. This is because every country uses its own national language and English as second/foreign language. Therefore, bi-lingual document with one language being the English and other being the national language is very common. Postal documents are a very good example of such bi-lingual/script document. This paper deals with word-wise handwritten script identification of bi-script documents written in Persian and Roman. In the proposed scheme, simple but fast computable set of 12 features based on fractal dimension, position of small component, topology etc. are used and a set of classifiers are employed for script identification experiments. We tested our scheme on a dataset of 5000 handwritten Persian and English words and 99.20% of correct script identification is obtained.

Keywords- Persian handwritten Recognition, Word-wise script identification, Fractal dimension.

I. INTRODUCTION

By advances in communication between countries, multi-lingual multi-script documents are media of transaction that may contain two or more scripts. Persian (Farsi) is one of the cursive/semi-cursive scripts in both printed as well as handwritten styles. It is mainly used in Iran, Afghanistan, Tajikistan and Uzbekistan. Persian is the national language of Iran and is employed as a medium of linguistic transaction by more than 120 million people [1]. The Persian script and other related scripts such as Arabic, Urdu, Pashto and Dari that are almost the same with minor differences in writing style, are also media of transactions amongst approximately 10% of the world population. Though Persian is the national language of Iran, Roman is also well accepted. So, bi-script document with one being the Roman and other being the local script is very common. See Figure 1(a-c) where examples of different bi-lingual documents written in Persian and Roman scripts are shown. From the first line of Figure 1(c) it can be seen that not only document is bi-script, a single line may be so.

There are two ways to deal with multi-script documents. One approach is to have an OCR system, which will treat multi-script document as a single script document where this script consists of all the properties of the individual scripts, which is very complex and difficult to implement as every language/script has its language specific rules and grammars. The other approach is to have a script identification module followed by script specific OCR system. For the development of a handwriting OCR, character segmentation plays a vital role, which needs script specific knowledge for proper segmentation. For example, while writing the Roman, characters touch each other mainly in the lower part whereas in some Indic scripts they touch in the upper part of the characters.
Moreover, there are many works reported on script identification in the literature and they are mainly for European scripts and there is also much progress in multi-script OCR of European languages, it is still to take full pace in Indian subcontinent [2-6]. Some works are available for printed Indian, Persian and Arabic scripts identification [7-15]. A few works for identification of handwritten Indian and Arabic scripts are also available in the literature [15-19]. However, to the best of our knowledge no work on word-wise script identification for handwritten Persian and Roman scripts is available in the literature. Hence in this paper we propose a script identification module, which deals with word-wise handwritten script identification from multi-script documents written in Persian and Roman. In the proposed scheme simple but fast computable set of 12 features, based on fractal dimension, position of small component, base line etc. are used for script identification. We have tested our scheme with classifiers like Neural Network [20], SVM [23], Nearest Neighbor[??], MQDF[24] and found that though results are comparable, in this case we got best results via SVM based classifier and the success rate was of 99.20% without any rejection. The proposed technique works for printed text as well.

Rest of the paper is organized as follows. Section 2 details Persian script. Pre-processing including data collection is described in Section 3. Section 4 deals with Feature extraction and word-wise script separation is described in Section 5. Experimental results are provided in Section 6 with conclusion in Section 7.

II. PERSIAN CHARACTERISTICS

The Persian alphabet set contains 32 basic shape (isolated) characters [21]. Figure 2 shows the 32 isolated printed Persian characters, which has 4 characters more when compared with Arabic characters. The characters which do not appear in Arabic are indicated by ‘*’. Each Persian character has between two and four shapes and the choice of which shape to use depends on the position of the character within a word or a sub-word. This makes the effective size of the alphabet about 115 characters. In the Persian, Arabic, Urdu and Pashto alphabet sets there are several letters that share the same basic form and differ only by a complementary part. The complementary part can be only one dot, a group of dots (2 or 3 dots) or a slanted bar (Figure 3-a to 3-c). The dot(s) can lie above, below or inside the letter. Another characteristic is the existence of a diacritical mark placed above the letter called “Gaf”. This diacritic changes whole meaning of the letter called “Kaf” (Figure 3-d). Therefore, any deletion of the dots/diacritic may result in a misinterpretation of a character/word.

Persian script is written from right to left and letters within a word are normally joined even in printed form. Letter shape and whether or not to connect is dependent on the letter and the other letters in its neighbors. Letters are connected at the same relative height. The “baseline” is the line at the height at which letters are connected and it is analogous to the line on which a Roman word sits. There is no connection between separate words/subwords and word/subwords are always separated by a little space between two words/subwords. There are seven characters (١، ۱، ٢، ٣، ٤، ۵، ٦، ٧) which can be connected only from right side. When they occur in the middle of a word, the word is divided into two or more subwords. Different Persian words with different number of subwords are shown in Figure 4. These special characteristics of Persian scripts make the Persian handwriting recognition a challenging problem.
III. PREPROCESSING

Document digitization for the present work has been done mainly from some real life data collected from a post-office (Cossipore post office of North Kolkata circle, West Bengal, India). The images are in gray tone and digitized at 300 dpi. As Persian script is not a very common and hence we did not find enough Persian script data from the postal documents so collected. Therefore, we have also collected some filled in form data for our proposed scheme. The handwritten word images were manually extracted and labeled. The dataset used for the present work consists of a total number of 5000 (2577 images of Persian and the rest are of Roman) handwritten words. The Persian (Roman) images include words with different number of sub-words (characters) varying from 1 (2) to 6 (15).

IV. FEATURE EXTRACTION

The feature selection is a very important task in any recognition scheme. They should be robust and easy to compute. The effectiveness of fractal dimension as a feature for script identification is proved in the work by Roy and Pal [17]. So the fractal dimension is chosen as one of our key features for script identification in the proposed scheme. Comparing Roman and Persian character set we found that there are a total of 28 small components present on the upper, lower or middle portion in all isolated characters of Persian character set compared to only 2 in upper part (in “i”, & “j”) in Roman. Therefore, they could serve as a clear mark of distinction. In addition, since these small components may be absent in some word images, some topological features like the baseline etc. are also considered. The features are described in the following subsections.

A. Fractal based feature

A fractal [22] is defined as a set for which the Hausdorff-Besikovich dimension is strictly larger than the topological dimension. The fractal dimension is a useful method to quantify the complexity of feature details present in an image. The fractal dimension is an important characteristic of the fractals because it contains information about their geometric structures. By employing fractal analysis, researchers typically estimate the dimension from an image.

The fractal dimension of continuous object is an entity specified in terms of well-defined mathematical limiting processes. A fractal is an irregular geometric object with an infinite nesting of structure at all scales (self-similarity). The fractal theory developed by Mandelbrot and Van Ness was derived from the work of mathematicians Hausdorff and Besikovich. The Hausdorff-Besikovich dimension (\(D_h\)) is defined as:

\[
D_h = \lim_{\varepsilon \to 0} \frac{\ln N_\varepsilon}{\ln 1/\varepsilon}
\]

where \(N_\varepsilon\) is the number of elements of \(\varepsilon\) diameter required to cover the object. Mandelbrot defines a fractal as a set for which the Hausdorff-Besikovich dimension strictly exceeds the topological dimension.

When working with discrete data, one is interested in a deterministic fractal and the associated fractal dimension (\(D_f\)) which can be defined as the ratio of the number of self-similar pieces (\(N\)) object–objects whose dimensionality is integer valued. However, the surfaces of many objects cannot be described with an integer value. These objects are said to have a “fractional” dimension. The magnification factor (1/\(r\)) into which an image may be broken. \(D_f\) is defined as:

\[
D_f = \frac{\ln N}{\ln 1/r}
\]

\(D_f\) may be a non-integer value, in contrast to objects lying strictly in Euclidean space, which have an integer value. However, \(D_f\) can only be directly calculated for a deterministic fractal. There are varieties of applicable algorithms for estimating \(D_f\), and we have used Box-counting algorithm for the same. Here we have used the fractal dimension of the full image, its contour, upper half and lower half of the contour of the image constituting 4 features based on fractal dimension. We noticed that fractal feature plays an important role in our script separation scheme.

B. Small component based feature

Here we take all the components whose height and width is less than twice of the stroke width (statistical mode of the horizontal run-length values of the image) and compares their position with respect to the base line. If such components lie completely above or below the base line then number of those components is used as feature. This feature is selected on the basis that, compared to Roman we find the number of characters having multiple dots in upper or lower part is very common (19 of the 32 isolated characters contain dots in upper or the lower part or both upper and lower) in Persian script. We have used a total of three features based on position of the small components in respect to the word image.

C. Topological features

The topological features considered here used in the present work are: area of loop, position of maximum length of the horizontal black run (i.e. the baseline), the statistical modes of horizontal, vertical black run, etc. We have used 5 features based on topology of the image.

Considering the above-mentioned features a feature set of 12 features is generated. To normalize the features, all the extracted features are divided to appropriate parameters. The advantages of these features are that they are simple and it is easy to compute them.
V. IDENTIFICATION PROCEDURE

Based on the above-normalized features, we employed Multi-layer Perceptron (MLP) Neural Network [20], Support Vector Machine (SVM) [23], k-Nearest Neighbor (k-NN) (with the Euclidian and the City-block) & MQDF [24] based classifier for identification of handwritten Persian and Roman script.

A. MLP Network

The MLP Network is, in general, a layered feed-forward network, pictorially represented with a directed acyclic graph. Each node in the graph stands for an artificial neuron of the MLP, and the labels in each directed arc denote the strength of synaptic connection between two neurons and the direction of the signal flow in the MLP. For pattern classification, the number of neurons in the input layer of an MLP is determined by the number of features selected for representing the relevant patterns in the feature space and output layer is chosen by the number of classes in which the input data belongs. The neurons in hidden and output layers compute the sigmoidal function on the sum of the products of input values and weight values of the corresponding connections to each neuron.

Training process of an MLP involves tuning the strengths of its synaptic connections so that it can respond appropriately to every input taken from the training set. The number of hidden layers and the number of neurons in a hidden layer require designing an MLP and also they should be determined during training process. Training process incorporates learning ability in an MLP [20]. Generalization ability of an MLP is tested by checking its responses to input patterns, which do not belong to the training set.

Since the number features is 12 and the number of possible classes in handwritten script for the present case is 2, the numbers of neurons in input and output layers of the Perceptron are set to 12 and 2, respectively. The number of hidden units (8), back Propagation learning rate and acceleration factor are set to suitable values, based on trial runs. Thus, finally a 12-8-2 MLP is designed for the present case.

B. SVM

The SVM has originally been defined for two-class problem and it looks for the optimal hyper-plane, which maximized the distance margin between the nearest examples of both classes, named Support Vector. The linear SVM can be extended to a non-linear classifier by using kernel functions like Polynomial/Gaussian kernels. For detail of SVM see the work of Vapnik [23]. In this context, we have employed SVM with different kernels and parameters such as Gaussian and polynomial kernels during our experiments and we have received the best result using Gaussian kernel.

C. K-NN

Nearest Neighbor is part of supervised learning that has been used in many applications in the field of data mining, statistical pattern recognition, image processing and many others. The term of nearest can be taken to mean the smallest Euclidean (City-block) distances in n-dimensional feature (12 here) space. This takes a test sample feature in vector form, and finds the Euclidean (City-block) distance between this and the vector representation of each training example. The training sample closest to the test sample is termed its Nearest Neighbor. This exploits the ‘smoothness’ assumption that samples near each other are likely to have the same class. The Euclidean distance between two points P and Q in the Euclidean n-space, is defined as:

\[ d(P, Q) = \sqrt{(p_1 - q_1)^2 + (p_2 - q_2)^2 + \ldots + (p_n - q_n)^2} \]

where P and Q can be two feature vectors of two patterns each with n features.

In the same way, the general formula for the City-block distance, which is also known as Manhattan distance, can be written as follow:

\[ d(P, Q) = |p_1 - q_1| + |p_2 - q_2| + \ldots + |p_n - q_n| \]

where the result is the absolute differences between coordinates of a pair of objects or the distance between two objects.

K-nearest Neighbor is an algorithm where the result of new instance query is classified based on majority of K-Nearest Neighbor category. The Nearest Neighbor (NN) and k-Nearest Neighbor (k-NN) using the Euclidian (City-block) distance with k=3, k=5 and k=7 are employed as classifiers for more experiment.

D. MQDF

Script likelihood identification is done by the following Modified Quadratic Discriminant Function [24].

\[ g(X) = \frac{1}{|X-M|^2} \sum_{i=1}^{k} \frac{\lambda_i}{\lambda_i + h^2} (\phi_i^T(X-M))^2 \cdot h^2 + \ln[h^{2(k-n)} \prod_{i=1}^{k} (\lambda_i + h^2)] \]

where \( X \) denotes the input feature vector, \( \hat{M} \) denotes the sample mean vector for each script class, and \( \lambda_i \) and \( \phi_i \) denote the eigenvalues and eigenvectors of the sample covariance matrix. Values of constants \( h^2 \) and \( k \) are selected experimentally to achieve the best performance. In the following experiments, \( k \) is set to 7 and \( h^2 \) to \( 3/8 \cdot \sigma^2 \), where \( \sigma^2 \) is the mean of eigenvalues \( \lambda_i \)'s over \( i \) and character classes.

VI. RESULT AND DISCUSSION

In our experiment, we have used a database of 5000 handwritten words (2577 Persian and 2433 Roman
handwritten words) for our experiments. Out of them 4000 (2000 each) are used for training of the proposed system and the rest are used for testing. Identification accuracy (R) and Rejection rate (Rej) are computed as follows. If out of N characters, x1 character are correctly recognized, x2 characters are rejected, and x3 are erroneous by the system then R = (x1/N) and Rej= (x2/N) (where N=x1+x2+x3). We have used different classifiers with different parameters for word-wise handwritten Persian-Roman script identification. SVMs with different kernels and parameters such as Gaussian and Polynomial kernels are employed during our experiments and SVM with Gaussian kernel gave better result than other SVMs. The results and the respective parameters using SVM are tabulated in Table 1. The confusion matrix of script identification result employing SVM with the Gaussian kernel with Lambda (\(\lambda\)) = 1 and Gamma (\(\gamma\))=24 on test data is given in Table 2. The Nearest Neighbor (NN) and k-Nearest Neighbor (k-NN) using the Euclidian and City-block distances with k=3, k=5 and k=7 are also employed as classifiers for more experiment. Experimental results of k-NN using the Euclidian and the City-block distance measures and different values for k are tabulated in Table 3. Detailed classification results with confidence, error, rejection, etc. using MLP neural network are given in Table 4 as well. The rejection rate on the Persian data was 0.4% while no Roman word is rejected by our system. We have also noticed that the identification confidence increases to 99.49% (99.89%) when rejection rate is 1.31% (9.55%). With MQDF classifier we obtained success rate of 98.79% with 7 Eigen values (k=7) & \(\lambda =0.2\).

TABLE 1. SVMs with Different Parameters

<table>
<thead>
<tr>
<th>Kernel</th>
<th>Degree</th>
<th>(C)</th>
<th>Correct Identification (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Polynomial</td>
<td>3</td>
<td>1</td>
<td>98.39</td>
</tr>
<tr>
<td>polynomial</td>
<td>3</td>
<td>0.1</td>
<td>97.83</td>
</tr>
<tr>
<td>polynomial</td>
<td>3</td>
<td>0.01</td>
<td>96.38</td>
</tr>
<tr>
<td>Polynomial</td>
<td>2</td>
<td>0.01</td>
<td>98.43</td>
</tr>
<tr>
<td>Polynomial</td>
<td>2</td>
<td>1</td>
<td>97.34</td>
</tr>
<tr>
<td>Polynomial</td>
<td>2</td>
<td>0.001</td>
<td>87.61</td>
</tr>
<tr>
<td>Polynomial</td>
<td>4</td>
<td>1</td>
<td>98.63</td>
</tr>
<tr>
<td>Polynomial</td>
<td>4</td>
<td>0.1</td>
<td>97.99</td>
</tr>
<tr>
<td>Polynomial</td>
<td>4</td>
<td>2</td>
<td>98.79</td>
</tr>
<tr>
<td>Gaussian</td>
<td>(\lambda =1)</td>
<td>(\gamma=24)</td>
<td>99.20</td>
</tr>
<tr>
<td>Gaussian</td>
<td>(\lambda =1e-1)</td>
<td>(\gamma=1)</td>
<td>97.66</td>
</tr>
<tr>
<td>Gaussian</td>
<td>(\lambda =1e-1)</td>
<td>(\gamma=2)</td>
<td>97.26</td>
</tr>
</tbody>
</table>

TABLE 2. Confusion Matrix of SVM Using the Gaussian Kernel.

<table>
<thead>
<tr>
<th>Script</th>
<th>Persian</th>
<th>Roman</th>
</tr>
</thead>
<tbody>
<tr>
<td>Persian</td>
<td>99.21%</td>
<td>0.79%</td>
</tr>
<tr>
<td>Roman</td>
<td>0.82%</td>
<td>99.18%</td>
</tr>
</tbody>
</table>
Some misrecognised and rejected script words are shown in Figure 5 (a-c). Most of the errors are happened because of cursiveness nature of Persian handwritten.

To the best of our knowledge, this is the first of its kind and hence we could not compare our results. Therefore, we have tried to compare the proposed system using different features. Therefore, we tried the same dataset on the 25 dimensional features described in [17] and 13 dimensional features described in [18]. From the experiment it can be noted that the result with current feature set is not only gives higher identification result, it is also very fast compared to them. For comparing results obtained by different feature sets, the results are shown in Table 6.

It is worth to mention that the propose scheme is independent to text size and there is no need of any normalization also.

Table 6. Script Identification result with different classifiers

<table>
<thead>
<tr>
<th>Method</th>
<th>Feature Dimension</th>
<th>Recognition rate</th>
<th>Rejection</th>
</tr>
</thead>
<tbody>
<tr>
<td>Roy and Pal [15]</td>
<td>25</td>
<td>98.44%</td>
<td>0.84%</td>
</tr>
<tr>
<td>Roy and Majumder [16]</td>
<td>15</td>
<td>96.01%</td>
<td>0.67%</td>
</tr>
<tr>
<td>Proposed Method</td>
<td>12</td>
<td>99.20%</td>
<td>0.00%</td>
</tr>
</tbody>
</table>

VII. CONCLUSION

In this paper a scheme for word-wise handwritten script identification from bi-script documents written in Persian & Roman is proposed. In the proposed scheme a small set of features based on fractal dimension, position of small component, base-line of text, etc. are computed and the best identification result is obtained using the SVM with Gaussian kernel as classifier. At present, we have used only 12 features and in future we plan to improve the accuracy of the system by minimizing the script dependency on the features. We are also planning to use this scheme as a general script separation module for the development of multi-script OCR for the multi-lingual country like India.

ACKNOWLEDGEMENT

The authors would like to thank Prof. F. Kimura of Japan for his help in getting MQDF result in this work.

REFERENCES