Journal of Ambient Intelligence and Humanized Computing (2024) 15:943–960
https://doi.org/10.1007/s12652-018-1052-4

ORIGINAL RESEARCH

A comprehensive handwritten *Indic* script recognition system: a tree-based approach

Pawan Kumar Singh[1] · **Ram Sarkar**[1] · **Vikrant Bhateja**[2] · **Mita Nasipuri**[1]

Received: 18 March 2018 / Accepted: 15 September 2018 / Published online: 26 September 2018

Abstract
A noteworthy achievement has been accomplished in developing optical character recognition (OCR) systems for different *Indic* scripts handwritten document images. But in a multi-script country like India, this cannot serve the entire purpose of document digitization when such multi-script document images need to be converted into machine readable form. But developing a script-invariant OCR engine is almost impossible. Therefore, in any multi-script environment, a complete framework of script identification module is very essential before starting the actual document digitization through OCR engine. Keeping this research need in mind, in this paper, we propose a novel handwritten script recognition model considering all the 12 officially recognized scripts in India. The classification task is performed at word-level using a tree-based approach where the Matra-based scripts are firstly separated from non-Matra scripts using distance-Hough transform (DHT) algorithm. Next, the Matra and non-Matra based scripts are individually identified using modified log-Gabor filter based features applied at multi-scale and multi-orientation. Encouraging outcomes establish the efficacy of the present tree-based approach to the classification of handwritten *Indic* scripts.

Keywords Handwritten text · Script recognition · *Indic* script · Tree-based approach · Distance-Hough transform algorithm · Modified log-Gabor filter transform · Statistical significance test

1 Introduction

The term "script recognition" can be defined as a process of recognizing the scripts of the texts, printed or handwritten, in any multi-script or multilingual milieu. Script recognition becomes an essential step before running any optical character recognition (OCR) module. This is because the process of script recognition is designed to categorize the printed or handwritten texts of a specific script type of the underlying document (Pal and Chaudhuri 2001). It is to be noted that the conversion of these texts into machine editable form written in various distinct scripts using a particular OCR engine is almost unfeasible. This problem could possibly be solved by using an assembly of OCR modules where the classified scripts will be further processed by the respective OCR engine (Singh 2013). Till the early years of this decade, researchers have paid almost no attention to the issue of language/script recognition. But, the need of automatic script classification is becoming more pertinent as the increasing demand for document processing in our day-to-day life makes the people to face situations where the task of script identification becomes unavoidable. This is more compulsory in a country like India, where multi-script is ubiquitous. Hence, researchers belonging to the OCR community have started to accept the requirement of a feasible solution of the script identification problem. Therefore, this necessitates in developing a pre-OCR script classification model which would help to categorize the script in which the input document is written. This would, in turn, facilitate to choose the specific OCR tool for the classified scripts. The

✉ Pawan Kumar Singh
pawansingh.ju@gmail.com

Ram Sarkar
raamsarkar@gmail.com

Vikrant Bhateja
bhateja.vikrant@gmail.com

Mita Nasipuri
mitanasipuri@gmail.com

[1] Department of Computer Science and Engineering, Jadavpur University, Kolkata, West Bengal, India

[2] Department of Electronics and Communication Engineering, Shri Ramswaroop Memorial Group of Professional Colleges (SRMGPC), Lucknow, India

applications of a script recognition system are as follows: (a) sorting of the document images script-wise, (b) selecting appropriate script-specific OCR module, (c) search online archives of document images for those containing a particular script and also processing large volumes of documents of unknown origin and (d) design a multi-script OCR system.

The solution to the dilemma of script classification is addressed at page-level, text line-level, and word-level (Singh et al. 2015a, b, c). Of these, the classification of the scripts at word-level in a multi-script scenario is usually more complicated and strenuous. At word-level, the number of candidate characters found in an individual text word may not be always sufficient for the script classification purpose. But recognition of scripts at word-level becomes an ideal and ultimate choice in a mixed-script scenario where a particular handwritten script document consists of words in multiple scripts. Keeping this fact in mind, in this research work, a handwritten script classification methodology is developed at word-level.

Despite the fact that the researchers have tried to solve the problem of handwritten script classification by considering multiple scripts, mostly the testing is measured, in bi-script and tri-script scenarios only fulfilling the partial requirement of the aforementioned problem. Limited research is a bare restriction in an Indian subcontinent, where 12 official scripts are prevalent. While considering a number of research articles on handwritten script recognition, we have seen numerous features used in the literature, and most of them have enormous computational complexities. If these proposed features are applied for the recognition of all the 12 *Indic* scripts considered altogether at a time, the computational complexity of the whole system becomes very high. Instead of solving this problem using orthodox (or brute force) methods, a hierarchical methodology of tree structure is considered as an alternative solution. The formation of hierarchical tree structure allows us to group analogous scripts at one level and then focus on the classification among the next analogous scripts at the next level leading to a model with improved recognition accuracy. The grouping of the scripts at each level of the tree structure is based on some *inter-script* specific features. Finally, each group of scripts is identified based on unique *intra-script* features. The script classification techniques, proposed earlier, (i.e., brute force method of pattern recognition) considered the *Indic* scripts concurrently; whereas it would always be a better option to classify them at different levels depending on their individual script dependent characteristics. That is why the present approach reduces the chance of misclassifications which, in turn, will augment the recognition accuracy of the whole model. The proposed tree-based approach for tackling the problem of handwritten script recognition is not found in the literature irrespective of the fact that it reduces the complexity of the overall classification model. The main highlighting issue of this paper is two-fold: (a) design of a novel tree-based approach for solving handwritten script recognition problem which has not been taken into account till date and (b) consideration of all the official scripts used in Indian sub-continent.

The organization of the paper is as follows: Sect. 2 describes the previous state-of-the-art research done for solving the problem of script recognition whereas a brief outline to the official scripts used in Indian sub-continent is explained in Sect. 3. Section 4 reports the proposed tree-based approach used for the classification of *Indic* scripts whereas Sect. 5 presents the detailed experimental evaluation of the proposed methodology. Finally, Sect. 6 concludes our paper.

2 Literature study

A comprehensive literature survey on *Indic* script classification illustrated by Singh et al. (2015a, b, c) depicts that a noteworthy research work has been carried out at word-level (Pal and Chaudhuri 1997; Patil and Subbareddy 2002; Dhanya et al. 2002; Sinha et al. 2004; Dhandra et al. 2006; Chanda et al. 2008, 2009; Padma and Vijaya 2010a, b; Das et al. 2011; Chanda and Pal 2005; Pati and Ramakrishnan 2008; Chaudhuri and Gulati 2016; Sarkar et al. 2010; Singh et al. 2013, 2014, 2015; Hangarge et al. 2013; Pardeshi et al. 2014; Obaidullah et al. 2017a, b; Sahare et al. 2018). But most of these works have been performed on printed text words (Pal and Chaudhuri 1997; Patil and Subbareddy 2002; Dhanya et al. 2002; Sinha et al. 2004; Dhandra et al. 2006; Chanda et al. 2008, 2009; Padma and Vijaya 2010a, b; Das et al. 2011; Chanda and Pal 2005; Pati and Ramakrishnan 2008; Chaudhuri and Gulati 2016) whereas a few researchers have considered the handwritten case (Sarkar et al. 2010; Singh et al. 2013, 2014, 2015, 2017; Hangarge et al. 2013; Pardeshi et al. 2014; Obaidullah et al. 2015, 2017a, b; Sahare et al. 2018). Sarkar et al. (2010) present a system, which separates the scripts of handwritten words from a document, written in *Bangla* or *Devanagari* scripts mixed with *Roman* script using multi layer perceptron (MLP) classifier. A set of eight different word-level holistic features is extracted from the word images. Singh et al. (2013) report an intelligent feature based technique, which identifies the scripts of handwritten words from a document page, written in *Devanagari* script mixed with *Roman* script using MLP classifier. A set of 39 distinctive features is designed combining 8 features described in Sarkar et al. (2010) and the rest 31 based on convex hull of each word image. Singh et al. (2014a, b) describe a robust word-wise script identification scheme for five different scripts written in *Bangla, Devanagari, Malayalam, Telugu* and *Roman*. A combination of shape based and texture based features

are extracted which is then classified by using MLP classifier. Hangarge et al. (2013) use directional discrete cosine transform (D-DCT) based word level handwritten script identification. This is done in order to capture directional edge information, one by performing 1D-DCT along left and right diagonals of an image, and another by decomposing 2D-DCT coefficients in left and right diagonals. The classification is performed using linear discriminant analysis (LDA) and *k*-nearest neighbor (*k*-NN) classifiers. Pardeshi et al. (2014) present a word-level handwritten script identification technique for 11 different major Indian scripts (including *Roman*) in bi-script and tri-script scenarios. The features are extracted based on the combination of Radon transform, discrete wavelet transform, statistical filters and DCT which are tested using support vector machine (SVM) and *k*-NN classifiers. An automatic handwritten script identification technique for document images of six popular *Indic* scripts *namely, Bangla, Devanagari, Malayalam, Oriya, Roman* and *Urdu* is described by the authors in Obaidullah et al. (2015). At first, a 34-dimensional feature vector is constructed using a combination of Radon transform (RT), DCT, fast Fourier transform (FFT) and distance transform (DT). Finally, a greedy attribute selection method is applied to choose 20 attributes and the classification is done at block-level using MLP classifier.

Very recently, a novel approach for separating *Indic* scripts with 'Matra' as the discrimination aspect is proposed by Obaidullah et al. (2017a). This approach uses the combination of an optimized fractal geometry analysis and Random forest classifier is applied for the improvement of the performance of handwritten script recognition from document images. In one of the works reported by Obaidullah et al. (2017b), a multi-level script classification research is carried out to choose the optimal level of work at which the task of handwritten script identification has been performed successfully. This work mainly deals with two distinct kinds of features: script dependent and script independent. A qualitative measure of these two feature sets is then evaluated to classify the input scripts at different levels. Singh et al. (2015a, b, c) present a word-level script classification procedure for seven *Indic* scripts *namely, Bangla, Devanagari, Gurumukhi, Malayalam, Oriya, Telugu* and *Roman* are using both elliptical and polygon approximation-based techniques. Sahare et al. (2018) propose a new scheme for script identification from word images using skew and scale robust log-polar curvelet features. These word images are first extracted in the form of text-patches from documents using Gaussian filtering. Thereafter, texture features are calculated using curvelet transform in log-polar domain and *k*-NN classifier is used to classify the input script word images. Recently, Singh et al. (2017) propose a two-stage word-level script identification technique for eight handwritten popular scripts *namely, Bangla, Devanagari, Gurumukhi, Oriya, Malayalam, Telugu, Urdu* and *Roman*. In the first stage, Discrete Wavelet Transform (DWT) is applied on the input word images to extract the most representative information whereas in the second stage, RT is applied to the output of the first stage. Finally, a set of 48 statistical features is computed from each word image which is classified using SVM classifier.

3 Outline of the Indian scripts

India is considered to be a multi-script homeland where 12 major scripts are written using 23 constitutionally acknowledged languages. The 12 official scripts used in Indian subcontinent are: *Oriya, Malayalam, Telugu, Bangla, Kannada,*

Table 1 General information regarding 12 official scripts used in India

Script	Origin	Number of basic characters		Number of native speakers (Millions) (Languages of India 2017)	Used to write language(s) (Ancient Scripts 2017)
		Vowels	Consonants		
Devanagari	*Brahmi*	15	33	366	*Hindi, Nepali, Marathi, Konkani, Sindhi* etc
Bangla		11	39	207	*Assamese, Manipuri* etc
Telugu		16	37	69.7	*Telugu*
Tamil		12	18	66.0	*Tamil, Sanskrit, Saurashtra, Badaga, Irula, Paniya* etc
Gurumukhi		12	30	57.1	*Punjabi, Sanskrit, Sindhi, BrajBhasha, Khariboli* etc
Gujarati		12	36	46.1	*Gujarati, Sanskrit, Kiutchi, Avestan* etc
Malayalam		13	36	35.71	*Malayalam*, etc
Kannada		13	36	35.3	*Kannada, Konkani, Tulu* etc
Oriya		14	38	32.3	*Oriya, Sanskrit* etc
Manipuri		6	15	24.20	*Meithei, Manipuri*,, *Kabui* etc
Urdu	*Persian*	10	28	60.3	*Urdu, Bati, Burushaski* etc
Roman/Latin	*Greek*	5	21	341	*German, English, Spanish, Indonesian, Italian* etc

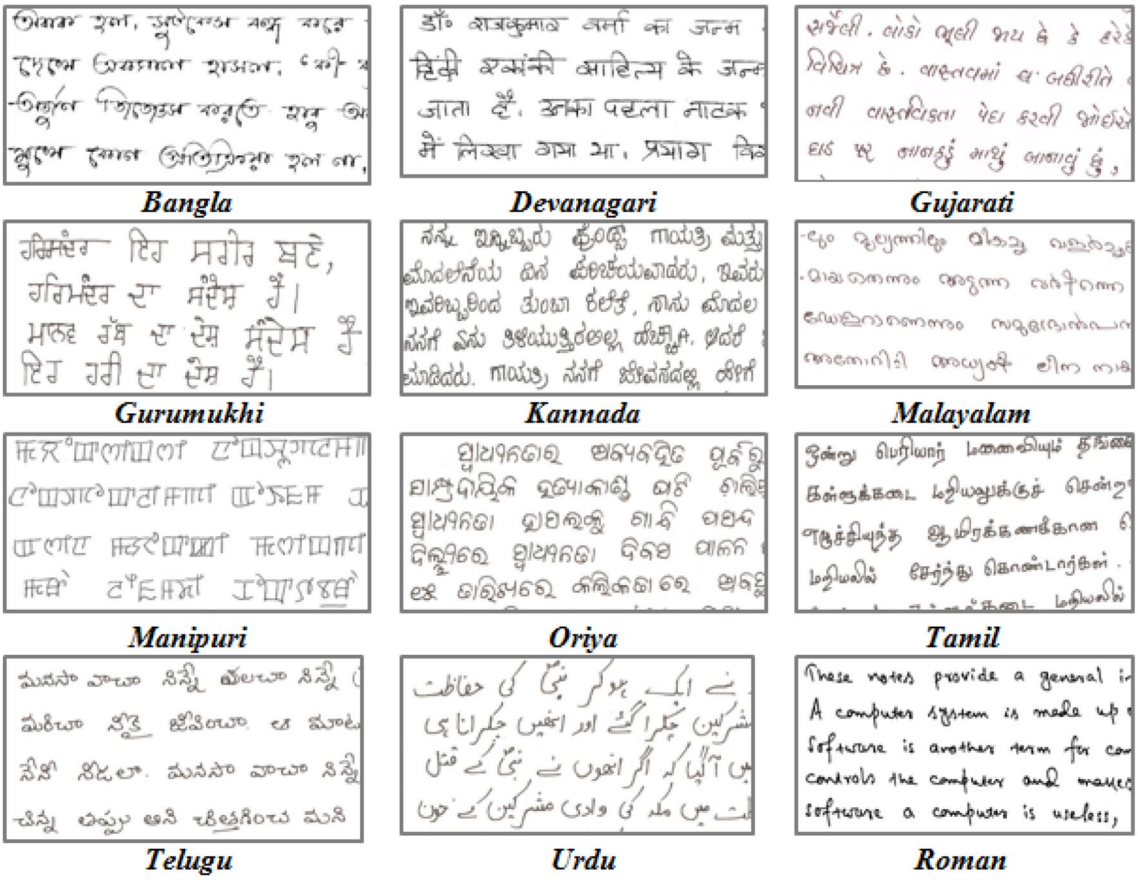

Fig. 1 Samples of handwritten texts in 12 different Indian scripts

Manipuri, Tamil, Gurumukhi, Urdu, Devanagari, Gujarati and *Roman*. Among these 12 scripts, the origin of *Urdu* and *Roman* scripts is the *Persian* and *Latin* scripts respectively. Apart from *Urdu* and *Roman* scripts, the origin of remaining scripts is the premature *Brahmi* script and is known as *Indic* scripts (Singh et al. 2017). The general information (such as the originality, number of basic characters present, number of native speakers and the languages written using a particular script) about the said scripts has been analyzed for non-Indian readers and detailed in Table 1. Figure 1 shows snapshots of handwritten texts written in 12 *Indic* scripts.

Among the 12 scripts, *Bangla, Devanagari, Oriya, Gujarati, Manipuri* and *Gurumukhi* are referred to North-*Indic* or *Indo-Aryan* scripts whereas the remaining four scripts *viz*., *Tamil, Telugu, Kannada*, and *Malayalam* are referred to as South-*Indic* or *Dravidian* scripts. The characters of some *Indo-Aryan* scripts (for example, *Devanagari, Manipuri, Bangla, Gurumukhi*) are found to possess a horizontal stroke over their top called "Matra"/"Shirorekha". "The Matra of one character touches the next character is forming a word in order to generate a common Matra. On the other hand, some scripts *viz*., *Gujarati, Urdu, Malayalam* and *Roman* etc. do not contain Matra but they include some vertical strokes" (Singh et al. 2015a, b, c). Table 2 represents the occurrence of stokes at various orientations for all 12 official Indian scripts as examined from the nature of their structures. Such orientations of strokes for these scripts are calculated approximately since the same are predicted from

Table 2 Presence of approximated strokes or line segments found at different orientations for 12 *Indic* scripts

Script	Orientations
Devanagari	$0^0, 90^0$
Bangla	$0^0, 45^0, 135^0$
Telugu	$0^0, 30^0, 90^0, 120^0, 150^0$
Tamil	$0^0, 75^0, 90^0, 120^0, 150^0$
Gurumukhi	$0^0, 90^0, 120^0$
Gujarati	$60^0, 90^0, 120^0$
Malayalam	$0^0, 45^0, 90^0, 120^0, 150^0$
Kannada	$0^0, 15^0, 90^0, 120^0, 150^0$
Oriya	$45^0, 60^0, 90^0, 120^0$
Manipuri	$0^0, 75^0, 90^0$
Urdu	$60^0, 90^0$
Roman/Latin	$30^0, 90^0$

Fig. 2 Figure showing the strokes (highlighted in red) for different script handwritten word images (the first two scripts are Matra-based whereas the rest two are non-Matra based). (Color figure online)

the *visual appearances* of different handwritten script word images. Figure 2 illustrates various strokes found in distinct script text words. Here, the study also examines that though handwritten words may be slanted and/or skewed in some cases, but we assume that generally the writers follow these basic traits during writing unless (s)he has an emergency or bad mood at the time of penning.

4 Proposed methodology

The proposed script classification framework comprises of a tree-based approach. In this approach, all the 12 scripts are initially classified using MLP classifier and the confusion matrix is then examined. Based on the data of the *confusion matrix* with a minor biasness to the *visual appearances* of the scripts (Hiremath and Shivashankar 2008), it is inferred that the Matra-based and non-Matra based scripts could be formed two separate clusters. So, at the first level, all the 12 scripts are pigeonholed into two sub-groups *i.e.*, "**G1**: *Gurumukhi, Devanagari, Manipuri, Bangla*, and **G2**: *Gujarati, Oriya, Telugu, Kannada, Tamil, Malayalam, Urdu* and *Roman*". The feature extraction for this classification is done using our Distance-Hough transform (DHT) algorithm which combines the two significant concepts *namely*, Distance Transform (Rosenfeld and Pfaltz 1966) and Hough Transform (Duda and Hart 1972) to estimate a collection of distinct features from the handwritten script words. "The main advantage of the Hough Transform technique is that it is tolerant to gaps while approximating the boundary descriptions of the handwritten word images and is relatively unaffected by noise and occlusion, unlike other edge detectors" (Can and Duygulu 2011; Gonzalez and Woods 1992). In order to get an idea about the other kind of shapes in the word images, we have chosen Distance Transform on the word images. It is worth mentioning that this transform plots different script words, without considering their shape, into different stroke elements. Consequently, the application of Hough Transform is repeated over the word images obtained as output from the Distance Transform. This is done in order to acquire distinct feature vectors extracted from the shape variations of different script words. The DHT algorithm competently unifies two entirely dissimilar transforms by perceiving the benefits of both. The output of DHT algorithm on some Matra and non-Matra based scripts are shown in Fig. 3. The steps of implementation of DHT algorithm are described in Algorithm 1. Thus, using Algorithm 1, a 36-dimensional feature vector is estimated as Level-1 features.

Algorithm 1: DHT Algorithm	
Step 1:	Convert the RGB word image into a gray scale image.
Step 2:	Apply image smoothing operation using Gaussian low-pass filter .
Step 3:	Perform image binarization operation with a global threshold value α.
Step 4:	Implement Euclidean Distance Transform to the output word image from Step 4. Scale the image by a factor of 8. Here, large pixel values indicate large distance which helps to analyze the shapes. Except these high valued pixels, mark the rest as '0'.
Step 5:	Perform image thinning operation to extract the exact shape.
Step 6:	Implement Hough transform along 18 different orientations ($\theta = -90^0, -80^0, -70^0, \ldots\ldots, 0^0 \ldots., 80^0$), with ρ resolution taken as 1 pixel.
Step 7:	Compute the ρ-value corresponding to the maximum accumulator value along each orientation.

Fig. 3 Output transformed images (shown at the right-hand side) obtained after applying our DHT algorithm on some handwritten script word images (shown on left side)

Modified log-Gabor (MLG) filter based features, reported in Singh et al. (2015a, b, c), had performed well in the script classification task and therefore is chosen as the main feature descriptor in the present work to classify the script word images at the remaining two levels of the proposed tree architecture. For proper texture analysis of text word images written in different scripts, this feature has been proven very useful in the past. Recently, Singh et al. (2018) apply MLG filter as feature extractor and MLP classifier as classification methodology for solving the problem of handwritten script recognition at word-level

In order to preserve the spatial information, a windowed Fourier transform (WFT) is considered in the present work. WFT involves multiplication of the image by the window function and the resultant output is followed by applying the Fourier transform. WFT is basically a convolution of the image with the low-pass filter. Since for texture analysis, both spatial and frequency information are preferred, the present work tries to achieve a good tradeoff between these two. Gabor transform uses a Gaussian function as the optimally concentrated function in the spatial as well as in the frequency domain (Gonzalez and Woods 1992). Due to the convolution theorem, filter interpretation of the Gabor transform allows efficient computation of the Gabor coefficients by multiplying the Fourier transformed image with the Fourier transform of the Gabor filter. The inverse Fourier transform is then applied on the resultant vector to get the output filtered images. The algorithm for applying MLG filter transform is described in Algorithm 2.

Algorithm 2: Modified log-Gabor filter

Function ModifiedLogGabor(IMG)

[Image IMG has r rows and c columns]

[Number of scales:$nscale$ and Number of orientations:$norient$]

1. Set $theta = \frac{\pi}{norient * sigma}$.
2. Set $ifft = Fourier\ transform(IMG)$
3. Create Z=array of zeros having r rows and c columns.
4. Set $gaborSquareEnergy = Z$
5. Set $gaborMeanAmplitude = Z$
6. Set $EO = cell(nscale, norient)$ where EO is a cell array of convolution results.
7. Set $iftFiltArr = cell(1, nscale)$ where, $iftFiltArr = cell$. This corresponds to the array of inverse Fourier transforms of filters.
8. Apply$lp = lowpassfilter$ $([r, c], 0.4, 10)$
9. Set $logGabor = cell(1, nscale)$
10. Repeat 11 to 15 while $s <= nscale$
11. Perform $wavelength = minwavelength * (mult)^{s-1}$
12. Use$fo = 1/wavelength$
13. Apply $logGabor(s) = \frac{e^{-\log\left(\left(\frac{radius}{fo}\right)^2\right)}}{2*(thetasigma)^2}$
14. Apply $logGabor(s) = (logGabor(s)) * lp$
15. Increment$s = s + 1$
16. For each point in the filter matrix, calculate the angular distance from the specified filter orientation.
17. Repeat steps 18 to26 while $o <= norient$
18. Repeat steps19 to 25 while $s <= nscale$
19. Set $filter = logGabor(s) * spread(o)$
20. Apply $ifftFilt = Real(Fourier\ transform(filter)\ * \sqrt{r * c}$
21. Set $An = abs(EO(s, o))$
22. Calculate $gaborSquareEnergy(count) = sum(sum(An^2))$
23. Calculate $gaborMeanAmplitude(count) = mean(An)$
24. $count = count + 1$
25. Increment$s = s + 1$
26. Increment$o = o + 1$
27. Return $[gaborSquareEnergy, gaborMeanAmplitude]$
28. End

The default value of *sigma* is taken as 0.75 in Algorithm 2. The feature extraction procedure as described in Algorithm 2 is repetitively used at multiple scales and orientations in order to capture discriminative information (i.e., *gaborSquareEnergy*) for the classification of script word images at the remaining two levels

4.1 Level-2 features

At the second level, features based on MLG filter transform are estimated for 5 different scales (*nscale* = 1, 2, 3, 4, and 5) and 6 different orientations (*norient* = 0°, 45°, 75°, 90°, 120° and 135°) for the *intra-script* identification for the words belonging to **G1**. Here, the convolution of all filtered images is performed with the input word images to obtain 30 (i.e., 5×6) distinct features for a particular script text word. This results in generation of feature vectors in the form of matrices. The application of the present approach for handwritten *Devanagari* and *Gurumukhi* script words have been depicted in Fig. 4a, b respectively. The group **G2** is again classified into four distinct sub-groups *i.e.*, **G2.1**: *Gujarati, Oriya*; **G2.2**: *Kannada, Malayalam, Tamil, Telugu*, **G2.3**: *Urdu* and **G2.4**: *Roman*. This classification is done using MLG based

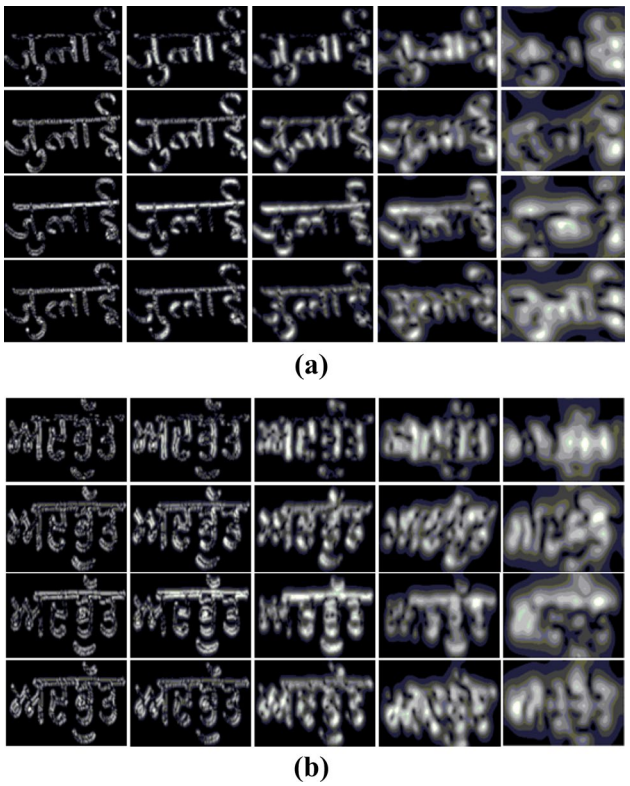

Fig. 4 Output word images written in: **a** *Devanagari*, and **b** *Gurumukhi* after applying MLG filter transform in 5 scales and 4 orientations (first, second, third and fourth rows show the output for *norient* = 0°, 30°, 60°, 90° and its corresponding five scales respectively)

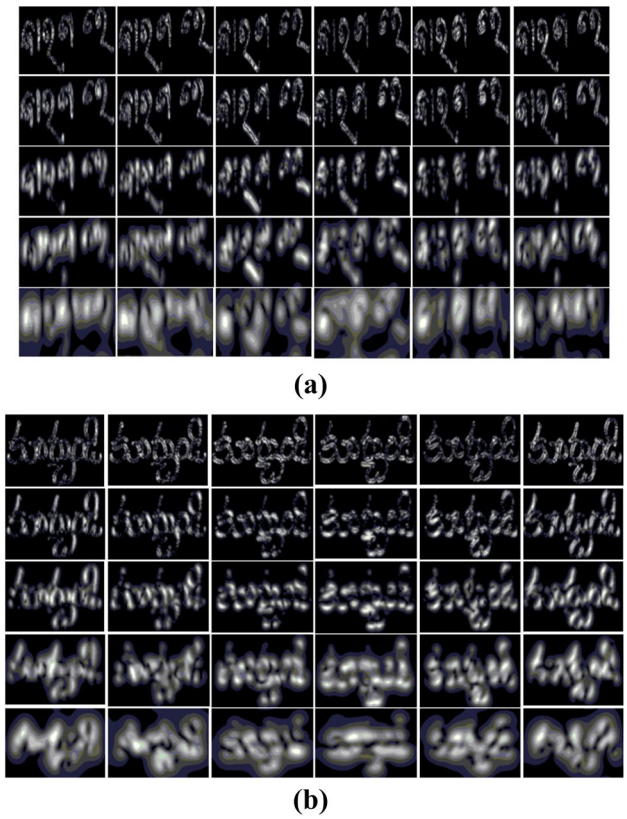

Fig. 5 Output word images written in: **a** *Oriya*, and **b** *Telugu* after applying MLG filter transform in 5 scales and 6 orientations (first, second, third, fourth and fifth rows show the output for *nscale* = 1, 2, 3, 4, 5 and its corresponding six orientations respectively)

feature vectors which are produced from 5 different scales (*nscale* = 1, 2, 3, 4, and 5) and 6 orientations (*norient* = 0°, 30°, 60°, 90°, 120°, and 150°) to obtain 30 different feature elements for a given input image. Therefore, level-2, the 6 scripts *namely, Gurumukhi, Devanagari, Manipuri, Bangla, Roman* and *Urdu* are identified by the proposed approach. Filtered word images generated after performing the present technique on sample *Oriya* and *Telugu* scripts are illustrated in Fig. 5a, b respectively.

4.2 Level-3 features

At this level, two scripts *namely, Gujarati*, and *Oriya* from the first subgroup, i.e., **G2.1** are finally identified based on MLG filter transform which are obtained from 5 different scales (*nscale* = 1, 2, 3, 4, *and* 5) and 4 orientations (*norient* = 45°, 60°, 90° and 120°) to get 20 different feature elements. Four South-*Indic* scripts under the second sub-group, **G2.2** are also identified based on MLG which are estimated from 5 different scales (*nscale* = 1, 2, 3, 4, *and* 5) and 6 orientations (*norient* = 15°, 30°, 45°, 75°, 90°, and 120°) to obtain 30 distinct feature elements for a sample script word

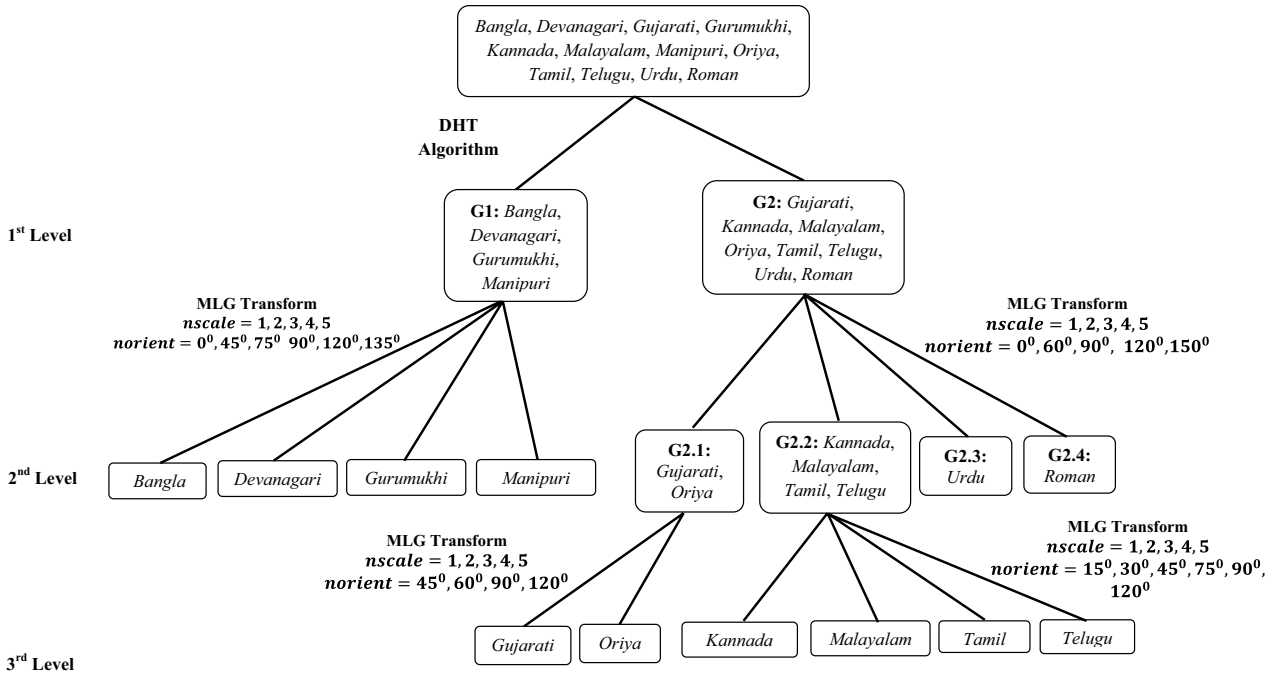

Fig. 6 Diagrammatic representation of the designed tree-based architecture for word-level handwritten script classification methodology

image. Finally, at level-3, the remaining 6 scripts *namely, Gujarati, Oriya, Kannada, Malayalam, Tamil*, and *Telugu* are classified.

The three-level tree architecture used for handwritten script identification of 12 official *Indic* scripts is illustrated in Fig. 6. It can be noticed from Fig. 6 that at the first level, the Matra based scripts are separated from the non-Matra based scripts using DHT algorithm. In the second level, all the four Matra based scripts such as *Bangla, Devanagari, Gurumukhi, Manipuri* are classified to their respective script classes using MLG filter transform based features. Among the non-Matra based scripts, the two scripts *namely, Urdu* and *Roman* are also successively classified. As a result, a total of six scripts are classified in the second level. The remaining six non-Matra based scripts *namely, Gujarati, Oriya, Kannada, Malayalam, Tamil, Telugu* are finally classified at the third level of tree classification. It is to be noted that the first level uses DHT algorithm as feature extractor whereas the classification of scripts at the rest two levels are done using MLG filter transform based features.

5 Classification results and analysis

Since, there is no standard benchmark database consisting of handwritten *Indic* scripts available in public domain, it became mandatory to collect handwritten word samples for the evaluation of the proposed work. A large number of people with varying sex, age, educational qualification etc have been involved in the collection of the database. Here, each person was asked to write at most 10–15 text words per script. The purpose for this is to achieve maximum veracity. A huge database consisting of 18,000 handwritten word images written in said 12 official *Indic* scripts have been collected for the evaluation of the proposed methodology. Here, each script contributes to exactly 1500 text words. These word images are scanned using a flat-bed HP scanner at a resolution of 300 dpi and stored as gray tone images. The pre-processing described in Nomura et al. (2009) is applied for removing noisy pixels from the scanned images. Few samples of handwritten word images written in 12 different scripts taken from our database are shown in Fig. 7. In order to evaluate our proposed tree-based approach, a total of 12,000 text words (i.e., 1000 words per script) has been randomly selected for designing a training set whereas the remaining 6000 text words (i.e., 500 words per script) has been considered for the testing purpose. The performances of three popular supervised classifiers *viz*., MLP (Basu et al. 2005), SVM (having polynomial kernel) (Cristianini and Shawe-Taylor 2000) and random forest (Breiman 2001) have been applied and compared over the tree structure. A brief discussion related to these classifiers is given in the following subsection.

Fig. 7 Handwritten word images in 12 *Indic* scripts

5.1 Design of classifiers

5.1.1 MLP classifier

MLP, a special kind of artificial neural network (ANN), is a feed-forward layered network of *artificial neurons* (Basu et al. 2005). Each artificial neuron in the MLP computes a *sigmoid function* of the weighted sum of all its inputs. An MLP consists of one *input layer*, one *output layer* and a number of *hidden* or intermediate *layers*. The numbers of neurons in the input and the output layers of MLP are mainly chosen as the number of features extracted for the given problem and the number of pattern classes used for the experiment respectively. The number of neurons in other layers and the number of layers in the MLP are determined by a trial and error method at the time of its *training*. For designing the MLP based classifiers, several runs of back propagation learning algorithm with two user-defined parameters *namely*, learning rate (η) and momentum term (α) are executed for different number of neurons in its hidden layer.

5.1.2 SVM classifier

SVM classifier effectively maps feature vectors to a high dimensional feature space where an 'optimal' separating hyperplane (the *maximal margin* hyperplane) is constructed (Cristianini and Shawe-Taylor 2000). To construct an optimal hyperplane, SVM employs an iterative training algorithm to minimize an error function which can be defined as:

Find $W \in \mathbb{R}^m$ and $b \in \mathbb{R}$ to

$$\text{Minimize } {}^1\!/_2 W^T W + C \sum_{i=1}^{N} \xi_i$$

subject to constraints $y_i\left(W^T \phi\left(x_i\right) + b\right) \geq 1 - \xi_i,\ i = 1, 2, \ldots, N$

$\xi_i \geq 0,\ i = 1, 2, \ldots, N$

where C is the capacity constant, w is the vector of coefficients, b is a constant, and ξ_i represents parameters for handling non-separable data (inputs). The index i labels the N training cases. Note that $y \in \pm 1$ represents the class labels and x_i represents the independent variables. The kernel ϕ is used to transform data of the input pattern (independent) to the feature space. It should be noted that the larger the C, the more the error is penalized. Thus, C should be chosen with care to avoid over fitting. There are number of kernels that can be used in SVM models. These include linear, polynomial, radial basis function (RBF) and sigmoid. For the present work, we have applied polynomial kernel which can be written as:

$$K\left(X_i, X_j\right) = \left(\gamma X_i . X_j + C\right)^d \tag{1}$$

where $K\left(X_i, X_j\right) = \phi\left(X_i\right).\phi\left(X_j\right)$ is the kernel function which represents a dot product of input data points mapped into the higher dimensional feature space by transformation ϕ.

5.1.3 Random forest classifier

A collection or ensemble of simple tree predictors constitute a Random Forest, each capable of producing a response when presented with a set of predictor values. For

classification problems, these responses acquire the type of a class membership, which relates, or classifies, a set of independent predictor values with one of the categories present in the dependent variable. The response of each tree depends on a set of predictor values selected independently (with replacement) and with the similar distribution for all trees in the forest, which is a subset of the predictor values of the original data set. The optimal size of the subset of predictor variables is given by log_2^{M+1}, where M is the number of inputs.

Given a set of simple trees and a set of random predictor variables, the Random Forest classifier defines a margin function that computes the extent to which the average number of votes for the correct class surpasses the average vote for any other class present in the dependent variable. This measure provides us with a suitable way of making predictions as well as provides a confidence measure with those predictions. The primary advantage of this classifier is that it can flexibly incorporate missing data in the predictor variables. When missing data are encountered for a particular observation (case) during model building, the prediction made for that case is based on the last preceding (non-terminal) node in the respective tree. Thus, missing data for some of the predictors do not necessary require removing cases from the analysis or computing proxy split information. For the present work, we have implemented the Random Forest classifier as described by Breiman (2001).

5.2 Justification of using tree-based approach

Before detailing the outcomes of the experiments, a justification is presented regarding the grouping of the scripts used in tree-based classification. Initially, the DHT algorithm is applied on all 12 *Indic* scripts and the obtained feature values are fed to different supervised classifiers. Since, MLP classifier produces the best recognition accuracy for the present work, so, we have considered MLP as the final classification in our case. The *confusion matrix* generated using MLP classifier on 12 said scripts are carefully examined to perform the best grouping of the undertaken scripts. This *confusion matrix* is illustrated in Table 3. It can be witnessed from Table 3 that the four Matra-based scripts are confused among each other whereas the remaining eight non-Matra based scripts are also confused among each other. So, the

Table 3 Confusion matrix outputted by MLP classifier on 12 *Indic* scripts at first level (*A* = *Bangla, B* = *Devanagari, C* = *Gujarati, D* = *Gurumukhi, E* = *Kannada, F* = *Malayalam, G* = *Manipuri, H* = *Oriya, I* = *Tamil, J* = *Telugu, K* = *Urdu* and *L* = *Roman*)

Script	*A*	*B*	*C*	*D*	*E*	*F*	*G*	*H*	*I*	*J*	*K*	*L*
A	**989**	153	10	144	9	5	125	36	7	6	3	13
B	112	**988**	16	241	11	3	93	10	12	0	0	14
C	14	18	**1059**	17	6	12	16	277	18	23	16	24
D	123	231	28	**901**	12	3	155	7	15	5	2	18
E	16	15	14	15	**898**	141	18	16	135	217	3	12
F	14	12	18	12	190	**850**	19	17	164	176	18	10
G	96	78	20	114	3	7	**1131**	9	18	11	5	8
H	24	19	253	18	14	23	9	**1065**	20	16	17	22
I	17	4	28	8	226	159	21	9	**828**	165	18	17
J	13	7	20	6	232	175	15	19	100	**883**	9	11
K	0	0	4	1	2	13	22	1	16	13	**1418**	10
L	12	11	13	8	14	3	15	14	5	8	3	**1394**

Bold values indicate the number of correctly recognized script word images

Table 4 Confusion matrix outputted by MLP classifier on 8 *Indic* scripts at second level

Script	*C*	*E*	*F*	*H*	*I*	*J*	*K*	*L*
C	**868**	7	8	146	6	9	5	10
E	16	**700**	57	5	39	73	2	6
F	11	23	**746**	3	125	29	4	9
H	139	13	5	**880**	8	10	3	7
I	9	34	11	18	**712**	23	6	15
J	16	28	51	9	40	**723**	8	9
K	3	8	6	2	5	3	**1385**	6
L	11	13	12	15	22	9	5	**1307**

Bold values indicate the number of correctly recognized script word images

Table 5 Comparison of results of the tree-based classification approach with individual feature extraction approaches (taking one at a time as well as their combination) using MLP classifier on 12 *Indic* scripts

Script classification methodologies	Size of feature vector	Classification accuracy (%)
DHT algorithm	36	70.94
MLG filter transform	90 (5 scales and 18 orientations)	85.65
Combination of DHT algorithm and MLG filter transform	126	90.05
Proposed tree-based approach	**Variable**	**94.23**

group is formed considering the Matra at the first level. At the second level, the feature extraction procedure is done with the help of MLG upon non-Matra based scripts and the *confusion matrix* obtained (shown in Table 4) is again scrutinized to locate the best grouping among these scripts. It can be noted from Table 4 that the two scripts such as *Gujarati* and *Oriya* get misclassified between each other whereas a reasonable amount of confusion is also created among the four South-*Indic* scripts. In case of *Urdu* and *Roman* scripts, a lesser amount of confusion is noticed with any other scripts. So, it is better to pool them in two distinct individual sub-groups. The experiment conducted also confirms our choice of grouping of scripts which helps in designing a tree-based structure. At the first level of *inter-script* classification, the four Matra-based scripts are separated from the eight non Matra-based scripts and the MLP classifier scores a classification accuracy of 94.39%. At the second level, the *intra-script* classification of the four Matra-based scripts as well as two non-Matra based scripts (such as *Urdu* and *Roman* scripts) produces 93.3% accuracy. Again, at the final level of *intra-script* tree classification, the remaining six scripts *namely, Gujarati, Oriya, Kannada, Malayalam, Tamil, Telugu* are recognized giving an accuracy of 95% using MLP classifier.

Furthermore, we have experimented with the best performing classifier (here it is MLP) to classify 12 *Indic* script classes using both DHT algorithm and MLG filter transform. It can be observed that DHT algorithm and MLG based features attain individual script classification accuracies of 70.94% and 85.65% respectively. In addition, the combination of these two feature vectors produces an identification accuracy of 90.05% which is less than the accuracy of the proposed tree-based approach. The comparison results are also shown in Table 5. This states the benefit of using a hierarchical strategy for word-level handwritten script classification.

5.3 Performance evaluation

The second part of the experiment is related with measuring the classification accuracy of the proposed script recognition schema. The performance of the developed approach is measured in terms of some popular statistical factors. The definitions regarding these parameters are already described in one of our previous works (Singh et al. 2017). A brief description of these parameters is also presented below for the general readers. These parameters are as follows: AAR, true positive rate (TPR), false positive rate (FPR), precision, recall, F-measure and area under ROC (AUC).

(a) AAR: AAR is used as assessment criteria for measuring the recognition performance of the proposed system which can be expressed as follows:

$$AAR = \frac{\#Correctly\ classified\ words}{\#Total\ words} \times 100\%. \tag{2}$$

Let us define the following notations:

True positive (TP): the number of correctly identified word images,

False positive (FP): the number of incorrectly identified word images,

True negative (TN): the number of correctly misclassified and.

False negative (FN): the number of incorrectly misclassified word images.

Now, it is obvious that the classification rule is better for higher values of TP and TN, and lower values of FP and FN.

(b) TP rate (TPR): This is defined as the proportion of positive cases that are correctly identified as positive. It is also known as *recall* or *sensitivity* and can be written as:

$$TPR = \frac{TP}{TP + FN}. \tag{3}$$

(c) FP Rate (FPR): This is defined as the proportion of negative cases that are incorrectly classified as positive. It can be written as:

$$FPR = \frac{FP}{FP + TN}. \quad (4)$$

(d) Precision: This is defined as the proportion of the predicted positive cases that are correct. It is also known as *consistency* or *confidence* and can be written as:

$$Precision = \frac{TP}{TP + FP}. \quad (5)$$

(e) F-Measure: This measure has been widely employed in information retrieval and is defined as the harmonic mean of recall and precision.

$$F - Measure = \frac{2.recall \times precision}{recall + precision} \quad (6)$$

(f) AUC: A receiver operating characteristic (ROC), or ROC curve, is a graphical plot that illustrates the performance of a binary classification system as its discrimination threshold is varied (Gonzalez and Woods 1992). The curve is created by plotting the TPR against the FPR at various threshold settings. The AUC of a

Table 6 Performance results of MLP classifier (along with the averages) while identifying 12 *Indic* scripts

Script	Statistical performance measures						
	AAR	TPR	FPR	Precision	Recall	F-measure	AUC
Bangla	89.1	0.891	0.017	0.946	0.891	0.918	0.937
Devanagari	96.3	0.963	0.015	0.956	0.963	0.960	0.974
Gujarati	95.3	0.953	0.041	0.958	0.953	0.956	0.989
Gurumukhi	94.3	0.943	0.029	0.915	0.943	0.929	0.957
Kannada	95.4	0.954	0.011	0.967	0.954	0.960	0.994
Malayalam	94.3	0.943	0.024	0.930	0.943	0.936	0.992
Manipuri	94.5	0.945	0.025	0.926	0.945	0.935	0.960
Oriya	95.9	0.959	0.047	0.954	0.959	0.956	0.989
Tamil	96.7	0.967	0.018	0.947	0.967	0.957	0.995
Telugu	92.4	0.924	0.018	0.944	0.924	0.934	0.988
Urdu	94.2	0.942	0.002	0.989	0.942	0.965	0.999
Roman	92.4	0.924	0.014	0.905	0.924	0.915	0.955
Average	**94.23**	**0.942**	**0.021**	**0.945**	**0.942**	**0.943**	**0.977**

Table 7 Performance results of SVM (along with the averages) while identifying 12 *Indic* scripts

Script	Statistical performance measures						
	AAR	TPR	FPR	Precision	Recall	F-measure	AUC
Bangla	89.1	0.891	0.022	0.932	0.891	0.911	0.987
Devanagari	96.2	0.962	0.009	0.974	0.962	0.968	0.995
Gujarati	95.3	0.953	0.053	0.947	0.953	0.950	0.950
Gurumukhi	94.6	0.946	0.031	0.910	0.946	0.928	0.989
Kannada	91.9	0.919	0.017	0.948	0.919	0.933	0.951
Malayalam	93.2	0.932	0.019	0.942	0.932	0.937	0.956
Manipuri	93.1	0.931	0.028	0.916	0.931	0.924	0.987
Oriya	94.7	0.947	0.047	0.953	0.947	0.950	0.950
Tamil	95.3	0.953	0.039	0.890	0.953	0.920	0.957
Telugu	89.6	0.896	0.025	0.922	0.896	0.909	0.935
Urdu	94.7	0.947	0.006	0.957	0.947	0.952	0.996
Roman	92.6	0.926	0.014	0.903	0.926	0.914	0.989
Average	**93.35**	**0.933**	**0.026**	**0.933**	**0.933**	**0.933**	**0.970**

Table 8 Performance results of random forest (along with the averages) while identifying 12 *Indic* scripts

Script	Statistical performance measures						
	AAR	TPR	FPR	Precision	Recall	F-measure	AUC
Bangla	87.1	0.871	0.034	0.895	0.871	0.883	0.975
Devanagari	94.9	0.949	0.027	0.921	0.949	0.935	0.993
Gujarati	94.5	0.945	0.109	0.897	0.945	0.920	0.970
Gurumukhi	89.1	0.891	0.025	0.923	0.891	0.906	0.986
Kannada	91.2	0.912	0.053	0.851	0.912	0.880	0.977
Malayalam	83.2	0.832	0.040	0.875	0.832	0.853	0.956
Manipuri	93.5	0.935	0.032	0.907	0.935	0.921	0.987
Oriya	89.1	0.891	0.055	0.942	0.891	0.916	0.970
Tamil	90.0	0.900	0.058	0.837	0.900	0.867	0.974
Telugu	85.9	0.859	0.047	0.860	0.859	0.858	0.967
Urdu	84.3	0.843	0.004	0.972	0.843	0.903	0.985
Roman	88.6	0.886	0.080	0.887	0.886	0.884	0.973
Average	**89.28**	**0.893**	**0.047**	**0.897**	**0.893**	**0.894**	**0.976**

Table 9 Recognition accuracies attained by three different classifiers on 10 randomly selected datasets for performing statistical significance tests (the numbers in bracket indicates the ranks)

Classifier	Dataset										Mean rank
	#1	#2	#3	#4	#5	#6	#7	#8	#9	#10	
MLP	95.6 (1)	96.3 (1)	96.6 (1.5)	98.07 (1)	98.27 (1)	96.2 (1.5)	97.52 (1)	96.93 (1)	92.87 (1)	98.45 (1)	$\mathbf{R_1 = 1.10}$
SVM	94.2 (2)	95.8 (2)	96.6 (1.5)	96.4 (2.5)	96.55 (2)	96.2 (1.5)	93.5 (3)	96.8 (2)	93.3 (2)	96.93 (2.5)	$\mathbf{R_2 = 2.10}$
Random forest	91.2 (3)	93.7 3 (3)	95.2 (3)	96.4 (2.5)	94.33 (3)	95 (3)	96.2 (2)	92.27 (3)	91.07 (3)	96.93 (2.5)	$\mathbf{R_3 = 2.80}$

Table 10 Summarization of results of two statistical significance tests computed from Table 9

Test	Degrees of freedom	Level of significance	Calculated value	Critical value	Null hypothesis (accepted/rejected)
Friedman (1937)	$(k-1)$	0.05	14.6	5.99	**Rejected**
Iman and Davenport (1980)	$[(k-1),(k-1)(N-1)]$	0.05	12.97	2.147	**Rejected**

ROC curve is a way to reduce ROC performance to a single value representing expected performance.

Tables 6, 7 and 8 illustrate the above mentioned statistical measures for script-wise classification using three classifiers (*viz.*, MLP, SVM and random forest). The values of two parameters *namely*, η and α for MLP classifier are experimentally set as 0.6 and 0.5 respectively and the classifier is made to run for 1000 epochs. Tables 6, 7 and 8 suggest that MLP classifier performs the best among three classifiers in terms of overall recognition accuracy which is found to be 94.23%. All the scripts except *Urdu* have been recognized with large margin by the MLP classifier. For *Urdu* script, the best result is obtained by SVM classifier.

5.4 Statistical significance tests

The statistical significance tests have also been performed for validating the performance of the multiple classifiers using multiple datasets. The details of these tests can also be found in Singh et al. (2014a, b). In the present work, we have performed non-parametric Friedman and Iman et al. test along with two post-hoc tests *namely*, Nemenyi test (Nemenyi 1963) and Bonferroni–Dunn test (Dunn 1961). For the experimentation, the number of datasets (N) and the number of classifiers (k) are set as 10 and 3 respectively. The recognition accuracies attained by three different classifiers on 10 randomly selected datasets are shown in Table 9. On the basis of these performances, the classifiers are then ranked for each dataset separately, the best performer gets

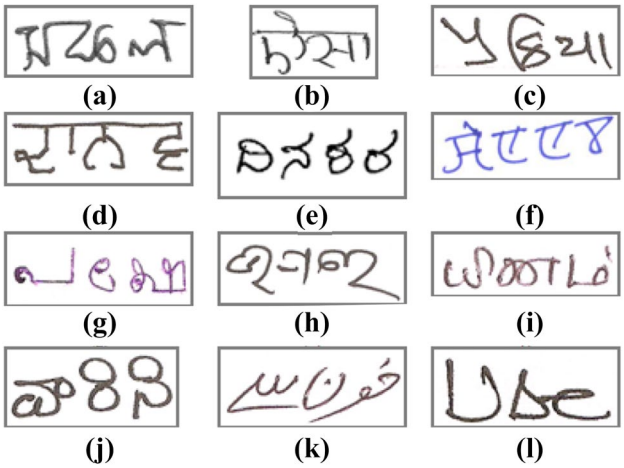

Fig. 8 Samples of successfully classified handwritten word images written in: **a** *Bangla*, **b** *Devanagari*, **c** *Gujarati*, **d** *Gurumukhi*, **e** *Kannada*, **f** *Manipuri*, **g** *Malayalam*, **h** *Oriya*, **i** *Tamil*, **j** *Telugu*, **g** *Urdu* and **h** *Roman* scripts respectively

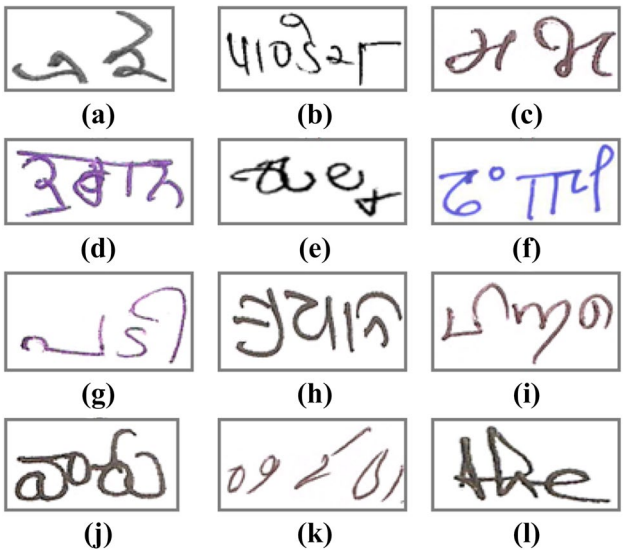

Fig. 9 Sample handwritten word images misclassified by the present technique due to presence of: **a, b** significantly non-Matra like structure constituting the text words in *Bangla* and *Devanagari*, **c** relatively few characters in *Gujarati*, structural similarity found in **d** *Gurumukhi*(misclassified as *Devanagari*), **e** *Kannada* (misclassified as *Telugu*), **f** *Manipuri*(misclassified as *Roman*), **g** *Malayalam* (misclassified as *Tamil*), **h** *Oriya* (misclassified as *Gujarati*), **i** *Tamil* (misclassified as *Malayalam*), **j** *Telugu* (misclassified as *Kannada*), **k** *Urdu* (misclassified as *Roman*) and **l** Matra like structure in *Roman* script respectively

rank 1, second best one gets rank 2, and so on (see Table 9). In case of ties, average ranks are assigned.

Table 10 shows the overall results of the above mentioned two tests. It can be noted from Table 10 that both Friedman and Iman et al. statistics reject the null hypothesis.

Since the null hypothesis is rejected, a post-hoc test known as Nemenyi test (Nemenyi 1963) is carried out for pair-wise comparison of the best and worst performing classifiers. For Nemenyi's test, the value of $q_{0.05}$ for 3 classifiers is 2.343 (see Table 5(a) of Demsar 2006). So, the *CD* is calculated as $2.343\sqrt{\frac{3\times4}{6\times10}}$ i.e. 1.047. Since, the difference between mean ranks between the best (MLP classifier) and worst performing classifiers (Random Forest classifier) is much greater than the *CD*, we can conclude that there is a significant difference between the performing ability of MLP classifier with the remaining two classifiers. For comparing all the classifiers with a *control classifier* (here MLP), we have also applied Bonferroni–Dunn test (Dunn 1961). But here, the value of $q_{0.05}$ for 3 classifiers is 2.241 (see Table 5(b) of Demsar 2006). In a similar manner, the *CD* for Bonferroni–Dunn test is calculated as $2.241\sqrt{\frac{3\times4}{6\times10}}$ i.e. 1.002. As the difference between the mean ranks of any classifier and MLP is always greater than *CD*, so it can be said that the chosen *control classifier* performs significantly better than other two classifiers. Hence, after the application of statistical significance tests, it can be concluded that MLP classifier outperforms the other two classifiers.

5.5 Error analysis

Some sample handwritten word images which are successfully classified by the present technique are shown in Fig. 8. It is observed from the confusion matrix that almost all the word images written in *Urdu* script are classified successfully. Regarding the errors observed here, it can be said that in most cases, small words containing less than 3 characters are misclassified by the present script recognition technique (Fig. 9c). The possible reason for this might be significant number of discriminatory feature values is not found to distinguish them from the others. Sometimes, the existence of non-uniform spaces in between characters of a single word image also causes the misclassification between scripts. Even, discontinuities in Matra in certain scripts like *Bangla* and *Devanagari* (see Fig. 9a, b) and existence of Matra like structure in *Roman* script (see Fig. 9l) misclassify them among each other. Also, the presence of structural similarity among the characters of different scripts such as *Gurumukhi* and *Devanagari* (Fig. 9d), *Kannada* and *Telugu* (Fig. 9e), *Manipuri* and *Roman* (Fig. 9f), *Malayalam* and *Tamil* (Fig. 9g), *Oriya* and *Gujarati* (Fig. 9h), *Tamil* and *Malayalam* (Fig. 9i), *Telugu* and *Kannada* (Fig. 9j), *Urdu* and *Roman*(Fig. 9k) lead to wrong identification of the script word images.

Table 11 Result evaluation of the developed script classification technique with few precedent techniques

Researchers	Feature description	Feature dimension	Type of *Indic* scripts used	Execution time (s)	AAR (%)
Padma and Vijaya (2010a, b)	Wavelet packet based features	44	*Bangla, Devanagari, Gurumukhi, Gujarati, Oriya, Malayalam, Manipuri, Kannada, Tamil, Telugu, Urdu* and *Roman*	798.957	81.54
Chaudhuri and Gulati (2016)	Gabor filters	30		795.28	79.36
Sarkar et al. (2010)	Holistic features	8		855.803	65.02
Singh et al. (2013)	Topological and convex hull based features	39		1076.575	76.47
Singh et al. (2014a, b)	Shape based and HOG features	87		961.892	85.43
Hangarge et al. (2013)	D-DCT	12		651.046	72.81
Obaidullah et al. (2015)	RT, DCT, FFT and DT	20		749.224	76.23
Obaidullah et al. (2017a)	Fractal geometry analysis	12		756.831	85.50
Obaidullah et al. (2017b)	Script dependent and independent features	56		827.650	89.42
Hiremath and Shivashankar (2008)	Wavelet based co-occurrence histogram	32		801.244	79.55
Proposed method	**DHT and Modified log-Gabor filter**	**Variable**		**624.756**	**94.23**

5.6 Performance assessment with well-known preceding techniques

The proposed tree-based script classification system is also compared with some previous techniques. The implementation of the feature sets, mentioned in Padma and Vijaya (2010a, b), Chaudhuri and Gulati (2016), Sarkar et al. (2010), Singh et al. (2013, 2014a, b), Hangarge et al. (2013), Obaidullah et al. (2015, 2017a, b) and Hiremath and Shivashankar (2008) have been performed and tested on the word-level script database prepared here. The comparison in terms of the classification accuracy, number of features utilized and the time requirement by different methods is detailed in Table 11. This analysis leads to the conclusion that the developed tree-based script classification approach performs better than the previous classification schemes considered here for comparison.

6 Conclusion

In this research work, the issue of handwritten script classification has been addressed for *Indic* script word images where we have introduced a tree-based approach for recognizing them. At the first level of tree-based classification, Matra and non-Matra based scripts are pooled together in two different groups. At the second level, all the Matra based scripts along with two non-Matra based scripts *namely, Urdu* and *Roman* are recognized and tested together. Finally, the third level identifies the remaining six non-Matra based scripts. Two feature sets are mainly employed for the identification purpose. DHT algorithm is applied at the first level whereas MLG based features at multiple scales and orientations are utilized for identifying the scripts in the remaining two levels. The proposed approach has attained the best overall recognition accuracy of 94.23% on 12 official *Indic* scripts using MLP classifier. The classification results are quite persuasive bearing in mind the number of scripts, their shape variations and complexities involved. The performance of the proposed tree-based approach is also compared with some of the previously script recognition methodologies and it can be noticed from the comparison results that the present work performs significantly better than those considered here for comparison. This work may further be extended involving other popular non-*Indic* scripts. Any newly designed technique may be integrated with the proposed system to develop effective multi-script OCR software.

Acknowledgements The authors are thankful to the Center for Microprocessor Application for Training Education and Research (*CMATER*) and Project on Storage Retrieval and Understanding of Video for Multimedia (SRUVM) of Computer Science and Engineering Department, Jadavpur University, for providing infrastructure facilities during progress of the work. The authors of this paper are thankful to all those individuals who had given appropriate consents and contributed wholeheartedly in developing the script database used in the current research.

References

Ancient, Scripts (2017) http://www.ancientscripts.com. Accessed 05 Dec 2017

Basu S, Das N, Sarkar R, Kundu M, Nasipuri M, Basu DK (2005) Handwritten 'Bangla' alphabet recognition using an MLP based

classifier. In: 2nd national conference on computer processing of Bangla, pp 285–291
Breiman L (2001) Random forests. Mach Learn 45(1):5–32
Can EF, Duygulu P (2011) A line-based representation for matching words in historical manuscripts. Pattern Recognit Lett 32(8):1126–1138
Chanda S, Pal U (2005) English, Devnagari and Urdu text identification. In: International conference on cognition and recognition, pp 538–545
Chanda S, Pal S, Pal U (2008) Word-wise Sinhala, Tamil and English script identification using Gaussian kernel SVM. In: 19th IEEE international conference on pattern recognition, pp 1–4
Chanda S, Pal S, Franke K, Pal U (2009) Two-stage approach for word-wise script identification. In: 10th IEEE international conference on document analysis and recognition (ICDAR), pp 926–930
Chaudhuri S, Gulati RM (2016) Script identification using Gabor feature and SVM classifier. In: 7th international conference on communication, computing and virtualization, Procedia computer science, vol 79, pp 85–92
Cristianini N, Shawe-Taylor J (2000) An introduction to support vector machines and other kernel-based learning methods. Cambridge University Press, Cambridge
Das MS, Sandhya Rani D, Reddy CRK, Govadhan A (2011) Script identification from multilingual Telugu, Hindi and English text documents. Int J Wisdom Based Comput 1(3):79–85
Demsar J (2006) Statistical comparisons of classifiers over multiple data sets. J Mach Learn Res 7:1–30
Dhandra BV, Mallikarjun H, Hegadi R, Malemath VS (2006) Word-wise script identification from bilingual documents based on morphological reconstruction. In: 1st IEEE international conference on digital information management, pp 389–394
Dhanya D, Ramakrishnan AG, Pati PB (2002) Script identification in printed bilingual documents. Sadhana 27(1):73–82
Duda RO, Hart PE (1972) Use of the Hough transformation to detect lines and curves in pictures. Commun ACM 15:11–15
Dunn OJ (1961) Multiple comparisons among means. J Am Stat Assoc 56:52–64
Friedman M (1937) The use of ranks to avoid the assumption of normality implicit in the analysis of variance. J Am Stat Assoc 32:675–701
Gonzalez RC, Woods RE (1992) Digital image processing. Prentice-Hall, New Delhi
Hangarge M, Santosh KC, Pardeshi R (2013) Directional discrete cosine transform for handwritten script identification. In: 12th IEEE international conference on document analysis and recognition (ICDAR), pp 344–348
Hiremath PS, Shivashankar S (2008) Wavelet based co-occurrence histogram features for texture classification with an application to script identification in a document image. Pattern Recognit Lett 29:1182–1189
Iman RL, Davenport JM (1980) Approximations of the critical region of the Friedman statistic. Commun Stat 9: 571–595
Languages of India (2017) Languages spoken by more than 10 million people. http://web.archive.org/web/20071203134724/http://encarta.msn.com/media_701500404/Languages_Spoken_by_More_Than_10_Million_People.html. Accessed 03 Dec 2017
Nemenyi PB (1963) Distribution-free multiple comparisons. PhD Dissertation, Princeton University
Nomura S, Yamanaka K, Shiose T, Kawakami H, Katai O (2009) Morphological preprocessing method to thresholding degraded word images. Pattern Recognit Lett 30(8):729–744
Obaidullah SM, Halder C, Das N, Roy K (2015) Indic script identification from handwritten document images-an unconstrained block-level approach. In: 2nd IEEE international conference on recent trends in information systems (ReTIS), pp 213–218
Obaidullah SM, Santosh KC, Halder C, Das N, Roy K (2017a) Automatic Indic script identification from handwritten documents: page, block, line and word-level approach. Int J Mach Learn Cybern. https://doi.org/10.1007/s13042-017-0702-8
Obaidullah SM, Goswami C, Santosh KC, Halder C, Das N, Roy K (2017b) Separating Indic scripts with 'matra' for effective handwritten script identification in multi-script documents. Int J Pattern Recognit Artif Intell 31(5):17
Padma MC, Vijaya PA (2010a) Script identification of text words from a tri lingual document using voting technique. Int J Image Process 4(1):35–52
Padma MC, Vijaya PA (2010b) Wavelet Packet Based Texture Features for Automatic Script Identification. International Journal of Image Processing 4(1):53–65
Pal U, Chaudhuri BB (1997) Automatic separation of words in multi lingual multi script indian documents. In: 4th IEEE international conference on document analysis and recognition (ICDAR), pp 576–579
Pal U, Chaudhuri BB (2001) Machine-printed and hand-written text lines identification. Pattern Recognit Lett 22(3–4):431–441
Pardeshi R, Chaudhuri BB, Hangarge M, Santosh KC (2014) Automatic Handwritten indian scripts identification. In: 14th IEEE international conference on frontiers in handwriting recognition (ICFHR), pp 375–380
Pati PB, Ramakrishnan AG (2008) Word level multi-script identification. Pattern Recognit Lett 29(9):1218–1229
Patil SB, Subbareddy NV (2002) Neural network based system for script identification in Indian documents. Sadhana 27(1):83–97
Rosenfeld A, Pfaltz J (1966) Sequential operations in digital picture processing. J ACM 13(4):471–494
Sahare P, Chaudhari RE, Dhok SB (2018) Word level multi-script identification using curvelet transform in log-polar domain. IETE J Res. https://doi.org/10.1080/03772063.2018.1430516
Sarkar R, Das N, Basu S, Kundu M, Nasipuri M, Basu DK (2010) Word level script Identification from Bangla and Devnagari Handwritten texts mixed with Roman scripts. J Comput 2(2):103–108
Singh PK (2013) Script identification from multi-script handwritten documents. M. Tech Dissertation, CSE Department, Jadavpur University
Singh PK, Sarkar R, Das N, Basu S, Nasipuri M (2013) Identification of Devnagari and Romanscript from multiscript handwritten documents. In: 5th international conference on PReMI. Lecture Notes in Computer Science, vol 8251, pp 509–514
Singh PK, Mondal A, Bhowmik S, Sarkar R, Nasipuri M (2014a) Word-level script identification from multi-script handwritten documents. In: 3rd International conference on frontiers in intelligent computing theory and applications (FICTA), AISC 327, vol 1, pp 551–558
Singh PK, Sarkar R, Das N, Basu S, Nasipuri M (2014b) Statistical comparison of classifiers for script identification from multi-script handwritten documents. Int J Appl Pattern Recognit (IJAPR) 1(2):152–172
Singh PK, Chatterjee I, Sarkar R (2015a) Page-level handwritten script identification using modified log-Gabor filter based features. In: 2nd IEEE international conference on recent trends in information systems (ReTIS), pp 225–230
Singh PK, Sarkar R, Nasipuri M (2015b) Offline Script Identification from multilingual Indic-script documents: a state-of-the-art. Comput Sci Rev 15–16:1–28
Singh PK, Sarkar R, Nasipuri M, Doermann D (2015c) Word-level script identification for handwritten Indic scripts. In: 13th IEEE international conference on document analysis and recognition (ICDAR), pp 1106–1110
Singh PK, Das S, Sarkar R, Nasipuri M (2017) A two-stage approach to handwritten indic script identification. In: Bhattacharyya S, Pan I,

Mukherjee A, Dutta P (eds) Hybrid intelligence for image analysis and understanding. Wiley, New York, pp 47–77

Singh PK, Sarkar R, Das N, Basu S, Kundu M, Nasipuri M (2018) Benchmark databases of handwritten Bangla-Roman and Devanagari-Roman mixed-script document images. Multimed Tools Appl 77(7):8441–8473

Sinha S, Pal U, Chaudhuri BB (2004) Word-wise script identification from Indian documents. Lecture Notes in computer science, DAS, vol 3163. Springer, Berlin, pp 310–321