AUTOMATIC LINE-LEVEL SCRIPT IDENTIFICATION FROM HANDWRITTEN DOCUMENT IMAGES - A REGION-WISE CLASSIFICATION FRAMEWORK FOR INDIAN SUBCONTINENT

***Sk Md Obaidullah*[1,*], *Chayan Halder*[2], *K. C. Santosh*[3], *Nibaran Das*[4], *Kaushik Roy*[5]**

[1]Dept. of Computer Science & Engg., Aliah University, Kolkata, India
[2, 5] Dept. of Computer Science, West Bengal State University, Kolkata, India
[3]Dept. of Computer Science, The University of South Dakota, South Dakota, USA
[4]Dept. of Computer Science & Engg., Jadavpur University, Kolkata, India

Email: sk.obaidullah@gmail.com[1], chayan.halderz@gmail.com[2], nibaranju@gmail.com[3], kaushik.mrg@gmail.com[4], santosh.kc@usd.edu[5]

DOI: https://doi.org/10.22452/mjcs.vol31no1.5

ABSTRACT

Script identification is a well-studied problem for automatic processing of document images. Several attempts have been made so far, but it is still far ahead from the complete solution. In this paper, an automatic approach for line-level handwritten script identification (HSI), considering eight official Indic scripts namely: Bangla, Devanagari, Kannada, Malayalam, Oriya, Roman, Telugu, and *Urdu is proposed. We consider a 148-dimensional feature vector using: image component fractal dimension, structural and visual appearance, directional stroke, interpolation and Gabor energy based texture features. For classification, we divide the whole script dataset based on different regions of India, to study a region-wise classification performance. Experimentation was carried out using the state-of-the-art classifiers: multilayer perceptron (MLP), support vector machine (SVM), random forest (RF), and fuzzy unordered rule induction algorithm (FURIA). Among all, we found that MLP as the best performer in terms of average accuracy of 98.2%, 99.5%, 99.1%, 99.5%, 99.9%, 98%, 98.9% for eight-script, bi-script, eastern, north, south Indian script groups, scripts with 'matra' vs without 'matra', and dravidian vs. non-dravidian groups respectively.*

Keywords: Handwritten script identification, image component fractal dimension, structural feature, directional stroke, interpolation based feature, gabor energy features, classification

1.0 INTRODUCTION

Optical Character Recognizer (OCR) [1] is an electromechanical tool for conversion of document images to its textual version. Firstly, collected documents are digitized using a scanner/camera, then by applying different intelligent OCR algorithm, textual information is extracted. Historical evidence shows, in earlier days, OCR development were initiated mainly for the blind and visually impaired users. But the need of text conversion from document images for automatic archiving and better indexing of huge volume of data has compelled the OCR research community to develop sophisticated and advanced technique for various real life applications. Few of these applications are: automatic form processing, sorting of postal documents, automatic processing of government documents irrespective of language barrier, etc. to name a few. Nowadays, OCR for different scripts is available throughout the world and the work is progressing very rapidly. But there is a problem for OCR development in a country like India, where 23 different languages (including English) are present and 11 different scripts (including Roman) are used to write them ([2]; [3]), making India as a multi-lingual/multi-script country. In our day to day life we come across various documents which are multi-script in nature such as postal documents and pre-printed application form Fig. 1 shows a sample line-level, multi-script document image of Bangla, Devanagari, Oriya, and Roman script. Processing these multi-script documents is a real challenge because in general OCR is script specific. A standard solution is to develop a General Purpose OCR (GPOCR) for all official Indic scripts. But this is not feasible because of the presence of many official scripts in India, which will generate a large character set. Another solution is to design a pre-processor which will identify the nature of the script first, then supply those scripts to the script specific OCR. This pre-processor is known as the Script

Identification System (SIS). So, there is a pressing need for the development of a SIS for official scripts of India. Fig. 2 shows a block diagram of the present work, where at the first line-level, multi-script documents are provided as input, then preprocessing module performs the binarization and line segmentation. After that we extract suitable features followed by classification. Finally, output is produced as a specific script type.

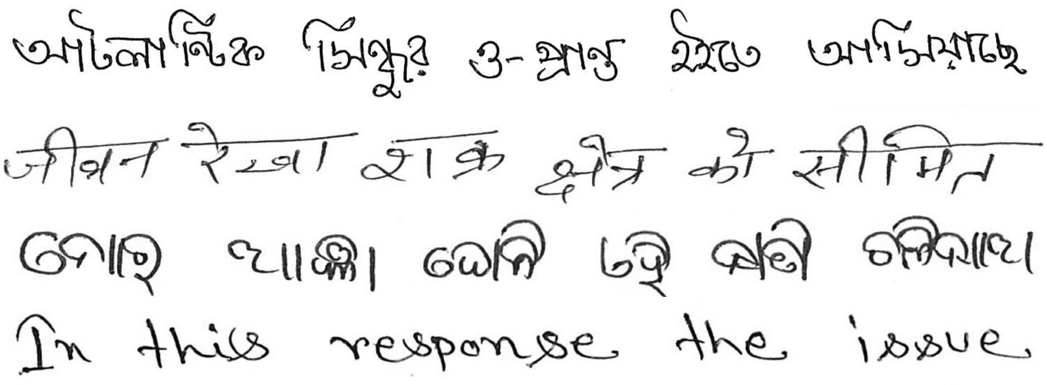

Fig. 1: Sample line-level multi-script document images

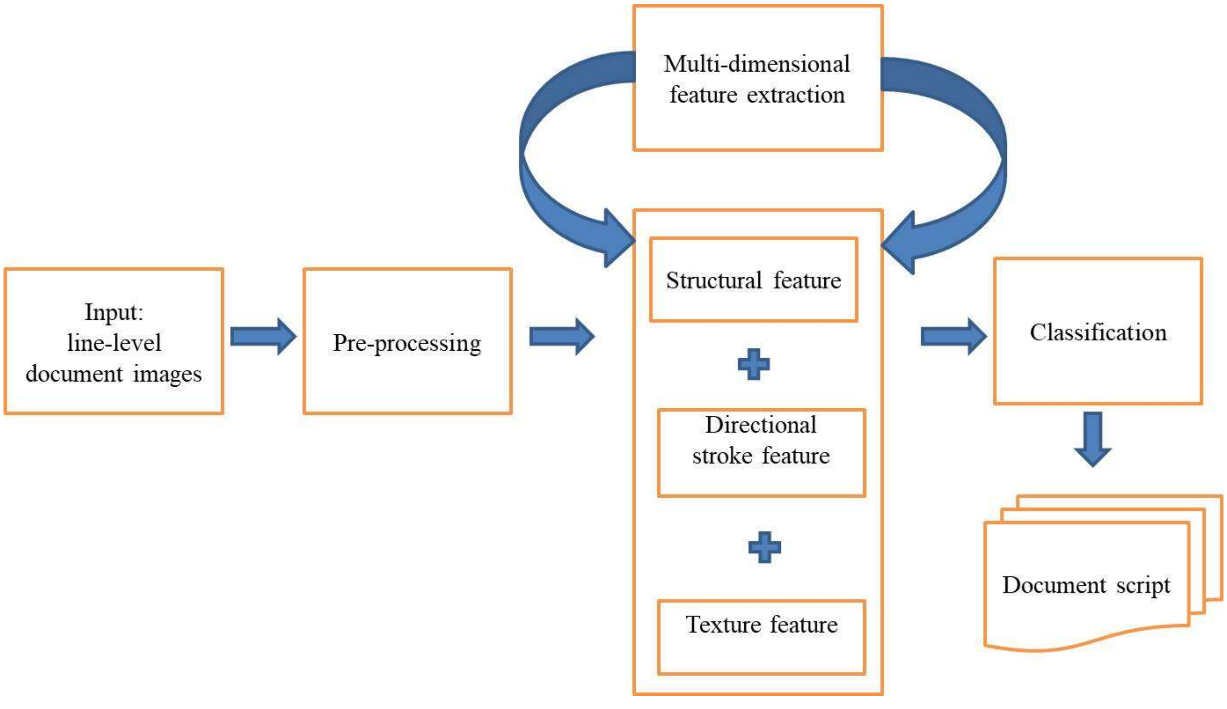
Multi-dimensional feature extraction
Input: line-level document images
Pre-processing
Structural feature
Directional stroke feature
Texture feature
Classification
Document script

Fig. 2: General block diagram of the proposed system

1.1 PROPERTIES OF INDIC SCRIPTS

As mentioned earlier, in India, there are 23 different languages (including English) and 11 official scripts (including Roman) are used to write them.

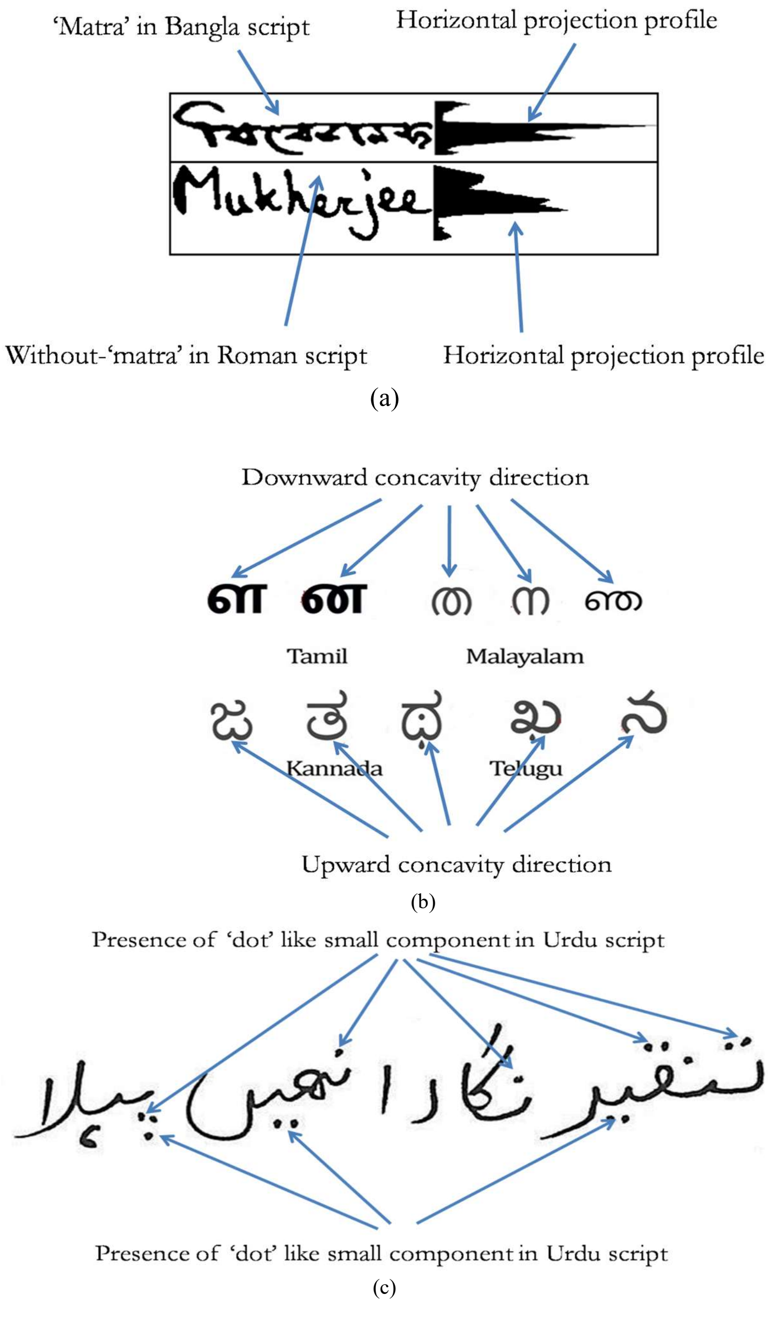
'Matra' in Bangla script
Horizontal projection profile
Mukherjee
Without-'matra' in Roman script
Horizontal projection profile
(a)
Downward concavity direction
ள ன
ത ന ഞ
Tamil
Malayalam
ಜ ತ ಥ
ఖ న
Kannada
Telugu
Upward concavity direction
(b)
Presence of 'dot' like small component in Urdu script
Presence of 'dot' like small component in Urdu script
(c)

Fig. 3: (a) Scripts with 'matra' and without 'matra' [15] (b) Concavity direction in south Indian scripts (c) Presence of dot ('.') like characters in upper and lower part of most of the Urdu characters

These scripts vary from one another in visual and structural appearance. Some of the key observations about these scripts are as follows:

- Presence of 'matra' or 'shirorekha': a horizontal line on the upper part of the words or sentence connecting more than one character resulting a larger connected component. Examples of 'matra'-based scripts are Bangla and Devanagari.
- Oriya and Malayalam scripts have components of more circular shape than others.
- Most of the characters in Tamil scripts contain a 'T' like shape in their structure.
- Urdu script is having maximum dot ('.') like small components as shown in figure 3(c). This script looks quite unlike than other Indic scripts. Many characters of Urdu contain directional strokes with orientation of 75^0.
- Roman scripts contain many vertical, horizontal, and slanting (45^0) strokes.
- Kannada and Telugu scripts are quite similar, except a 'tick' like symbol present in Telugu script which is not there in Kannada. Similarly Tamil and Malayalam characters are very much similar. Tamil and Malayalam characters have downward concavities and Kannada and Telugu characters have upward concavities as shown in figure 3(b).

1.2 Handwritten script identification – a quick survey

Script identification can be categorized into two broad types depending upon the document types, these are: printed script identification (PSI) and handwritten script identification (HSI). Handling handwritten documents is more complex due to several reasons like: irregular line, word and character spacing, character style, font and size vary from writer to writer, variations in stroke direction, stroke width. Many works are reported in literature on PSI since last decade ([4]; [5]; [6]; [7]; [8]; [9]; [10]; [11]; [41]). Among the few available works on HSI, Hochberg et al. [12] proposed a page-level script identification from few Indian and non-Indian scripts. They employed features like: aspect ratio, sphericity, white holes. Connected component based features were studied by Zhou et al. [13] to identify Bangla and Roman script. Line-level Roman, Devanagari, Bangla and Telugu scripts were identified by Singhal et al. [14]. They employed two texture features: Gabor filter with rotation invariant property and gray level co-occurrence matrix (GLCM). During the work of Indian postal automation, Roy et al. [15] studied water reservoir and busy zone based local features to identify Bangla and Roman scripts. Block-level script identification was studied by Hangarge et al. [16] for Roman, Devanagari and Urdu scripts. The authors employed some visual discriminating features to classify them. In another work, Hangarge et al. [17] discussed about directional discrete cosine transform (DCT) based features to identify six different Indic scripts. Discrete wavelet transform (DWT), radon transform (RT), DCT and statistical filters were collectively employed by Pardeshi et al. [18] to perform word-level script identification from eleven official Indic scripts. Though, the number of scripts considered in this work is fairly large, but the computational cost is a concerning issue in this work. Recently, the GLCM texture feature was employed by Singh et al. [19] for page-level script identification.

From the survey, it is indeed clear that most of the work focused on printed documents [2]. Very few works dealt with handwritten documents on Indic scripts, especially at page-level. In this paper, we propose a technique to identify script from handwritten document images written by eight official Indic scripts: Bangla, Devanagari, Kannada, Malayalam, Oriya, Roman, Telugu and Urdu to handle line-level multi-script documents. Our preliminary task is to find suitable features from input document images and then classifying those documents according to the script by which they have written. Features are in general application dependent. This means, a particular technique/system is designed targeting a particular application/dataset. In general, texture based features are found to be commonly used even though the performances did not go well as expected [2]. Therefore, we combine features to develop a generic concept to be applied for all possible scripts. To do so, we employ three different features: structural features, directional stroke features and texture features. The performances of these features have been analyzed individually and collectively using the state-of-the-art classifiers. Finally, our method is compared with some existing techniques at the same level, i.e. on the same dataset and setup. The present work is an extension of one of our earlier work [20]. We have analyzed a region-wise classification framework for Indic scripts and the experimentation has been carried out with extended dataset.

The remainder of the paper is organized as follows. We describe the proposed methodology in Section 2, which consists of pre-processing, feature extraction and classification. In Section 3, we discuss about experimental results. Finally, we conclude our paper in Section 4.

2.0 PROPOSED METHODOLOGY

2.1 Pre-processing

Pre-processing includes: storing the collected data into gray scale format, and converting those data into binary level. A two-stage based binarization technique is applied [15] to convert the gray scale image into two tone image. Then we extract multi-dimensional features from those binary images. In our work, a total of 148-dimensional feature vector is considered which are discussed in the following section.

2.2 Feature extraction

In a recent review [2], the authors have reported that no universal feature exists which can effectively classify all Indic scripts. So, we combine different features to propose a generic solution to the current problem. The features considered in our work are mainly three types: (i) structural features (fractal and shape), (ii) directional stroke features, and (iii) texture features (interpolation and gabor energy). Table 1 shows a brief description of different features and their dimensionality.

Table 1: Different features and their dimensionality in our work

Feature type		**Feature description**	**Feature Dimension**
Structural features (*f_1*)	Image component fractal dimension [45]	Average fractal dimension of top and bottom profile applying box counting algorithm	2
	Shape and visual appearance [10]	Convexity of a component as a global measure	8
		Circular or roundness of an image component	10
		Chain code based feature on outer and inner contour	16
		Bounding box fitting as a global measure	8
Directional stroke features (*f_2*)	Directional morphological reconstruction (Proposed)	Morphological operation using four directional kernels namely H-kernel, V-kernel, RD-kernel and LD-kernel	72
Texture feature (*f_3*)	Interpolation based [46]	Ratio of interpolated image and morphological image obtained after applying directional kernel	24
	Gabor energy ([9]; [40])	Gabor filter based feature with varying orientations and frequencies	8
Total			148

2.2.1 Structural features

2.2.1.1 Image component fractal dimension

The fractal dimension is an important topological property which can be very effective to distinguish scripts with ‘matra’ from their counterpart. The concept was employed from the idea of Mandelbrot’s fractal geometry theory [37]. We employed fractal dimension to distinguish scripts with ‘matra’, i.e. Bangla, Devnagari etc. from scripts without ‘matra, i.e. Roman, Urdu etc. [45]. It has several advantages: low dimension feature vector and computationally fast. Figure 4 presents a block diagram to compute fractal based features. Figure 5 shows the computation of fractal dimension of top and bottom profile of Bangla and Urdu scripts.

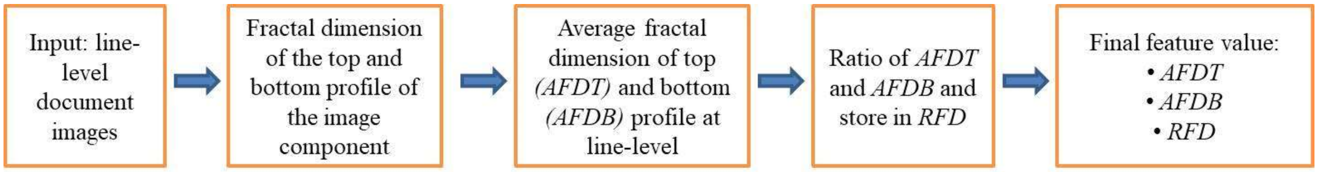
Input: line-level document images
Fractal dimension of the top and bottom profile of the image component
Average fractal dimension of top (AFDT) and bottom (AFDB) profile at line-level
Ratio of AFDT and AFDB and store in RFD
Final feature value:
• AFDT
• AFDB
• RFD

Fig. 4: Block diagram showing computation fractal features

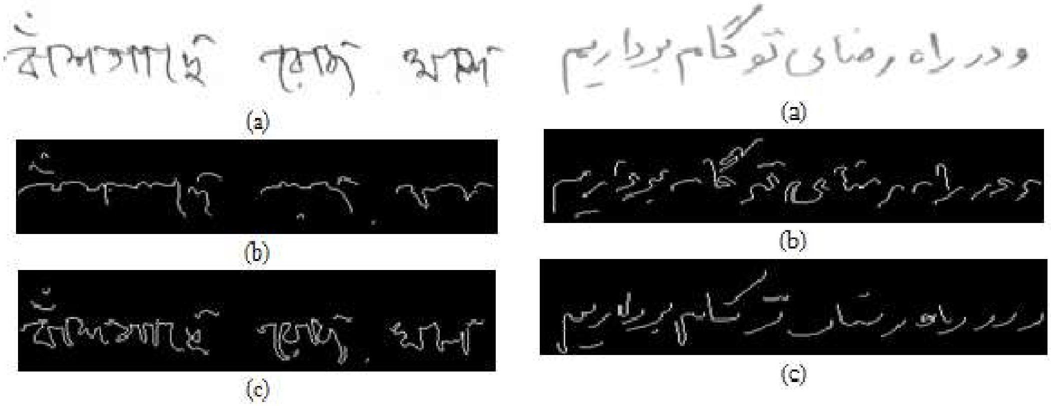

Fig. 5: Fractal dimension based feature computed on script with 'matra' (Bangla) and script without 'matra' (Urdu) (a) Original component, (b) Upper part of the contour, (c) Lower part of the contour

2.2.1.2 *Shape and visual appearance of the graphemes*

Shape or structure of the graphemes of different scripts carries very useful information about the overall visual appearance of the particular script. Few of our observation regarding visual appearance of different Indic scripts are as follows: Malayalam and Oriya script components are more circular in nature compared to others. So, circularity feature will fit well to them. Many characters and components in Roman have a rectangular shape. Directional chain code can be useful for separating scripts with 'matra' from their counterpart. We have computed different structural features like convexity defects, convex hull, circularity, rectangularity, chain code on the input images at component level [10]. The block diagram of computation of structural and visual appearance based features is shown in figure 6.

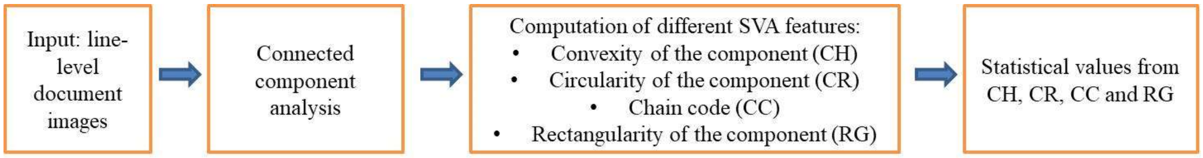
Input: line-level document images
Connected component analysis
Computation of different SVA features:
• Convexity of the component (CH)
• Circularity of the component (CR)
• Chain code (CC)
• Rectangularity of the component (RG)
Statistical values from CH, CR, CC and RG

Fig. 6: Block diagram of computation of SVA features

- *Chain code*

The presence of different directional strokes like horizontal, vertical, left, and right diagonal or any stroke with arbitrary orientation can be captured using chain code. First, contours of image component are drawn, and then 8-directional chain code is drawn on the contours (both inner and outer contour). So, the code of the components of different scripts will differ from each other. Then we compute chain-code direction histogram values as feature. A total of 16 features from both outer and inner contours are calculated. Figure 7 shows an example of popular 8-directional freeman chain code computed for a character "A" showing different direction.

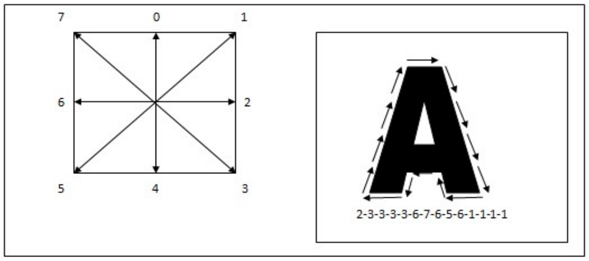

Fig. 7: An example of 8-directional freeman chain code [39]

- *Circularity*

Circularity is one of the important shape descriptor for Indic script components. Scripts like Oriya, Malayalam etc. have more circular components compared to others. The circularity feature in our work is computed as follows:

(i) A minimum enclosing circle: the circumscribed circle of the minimum area of the contour, where the radius of the circle is recorded.

(ii) Next we apply a technique to fit an ellipse around the given points of the contour. By calculating the averages of both the axes of the ellipse, a fitting circle is drawn. The radius of the fitting circle is recorded.

(iii) The difference of the two radiuses obtained in step (i) and (ii) is calculated which gives the circularity measure of those components.

Figure 8(a) shows the computation of circularity feature on Gujarati script. Here the 'blue' line is a minimum enclosing circle and 'green' line is drawn to fit an image component. If a component is perfectly circular then this difference will be perfect zero.

- *Bounding box*

Bounding box is used to measure the shape of the component whether it is perfectly square or not. Three measures are taken here: (i) perfect square (height/width = 1), (ii) 'horizontal rectangle' (height/width < 1) (iii) 'vertical rectangle' (height/width > 1). The script with 'matra' will have larger bounding box size compared to the scripts without-'matra' due to presence of larger component size (as 'matra' joins different characters). So, it is a distinguishable shape descriptor. To compute the feature, we measured the height, width, aspect ratio. Figure 8(b) shows sample output of bounding box computation.

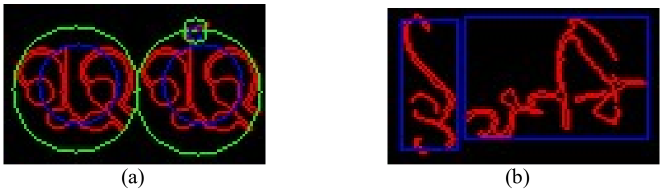

Fig. 8: (a) Computation of circularity of component on Oriya script using fitted circles (blue: minimum enclosing circle & green: best fitted circle), (b) Computation of Rectangularity of component on Bangla script (blue: rectangular box)

- *Convex hull*

The hull is computed for every selected component's inner and outer contours in our experiment. Minimum and maximum of the surroundings of inner and outer contour of the components are computed. Their average values and variance are also calculated. The convexity defects can also be computed from the hull images which have shown in figure 9(a). Figure 9(b) shows the computation of convex hull and convexity defects in Roman script.

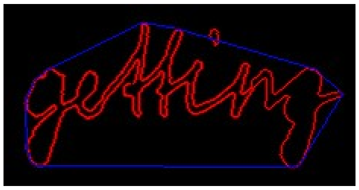

Fig. 9: Computation of convex hull on a from outer contour of Roman script component

2.2.2 Directional stroke feature

Different directional strokes are present in different Indic scripts. Urdu script has many characters with 75^0 directional strokes. Roman script is characterized with a 45^0 strokes. Bangla and Devanagari scripts contain 'matra' which is an 180^0 stroke. Besides these, other scripts also have different directional strokes with arbitrary orientations. To capture the stroke features, we have used directional morphological reconstruction with directional kernels. The morphological operations considered in this work are: image dilation, erosion, opening, closing, top-hat, and black-hat transforms. Based on our visual observation of the different directional strokes presence in different Indic scripts, we define four directional morphological kernels: *H-kernel* (horinzontal direction), *V-kernel* (vertical direction), *RD-kernel* (right diagonal direction) and *LD-kernel* (left diagonal direction). These kernels are 3x11, 11x3, 11x11 and 11x11 matrices correspondingly, where horizontal, vertical, right diagonal, and left diagonal pixels are 1 and rests are 0. A sample *H-kernel* is as follows:

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

To compute the feature values, at first the original image is dilated using a default kernel. Then each of the dilated images is eroded four times using four directional kernels (i.e. *H-kernel*, *V-kernel*, *RD-kernel* and *LD-kernel*). The ratio of those eroded images with the dilated one provides four features and computation of the average and standard deviation of the eroded images give other eight features, resulting a total of 12 features. In a similar way, other morphological operations namely opening, closing, gradient, top-hat and bottom-hat were performed, where each of them generates 12 features. Finally, under this category, 72 dimensional feature set is generated.

2.2.3 Texture feature

2.2.3.1 Interpolated morphological transform

Interpolation operations are used to perform image upsizing and downsizing. Popular interpolation techniques for texts [39] are: nearest neighbor, bilinear, pixel area re-sampling method, and bicubic interpolation. Pixel distribution based texture analysis can be done using an image interpolation technique to study the spatial variations of different Indic scripts. Figure 14 shows a flow diagram to carry out the interpolation based spatial texture feature. At first, line-level images are resized using nearest neighbor interpolation technique. Two types of morphological processing are done using four directional kernels. Then the ratio of the interpolated image and morphologically processed image are obtained to compute the final feature values [46].

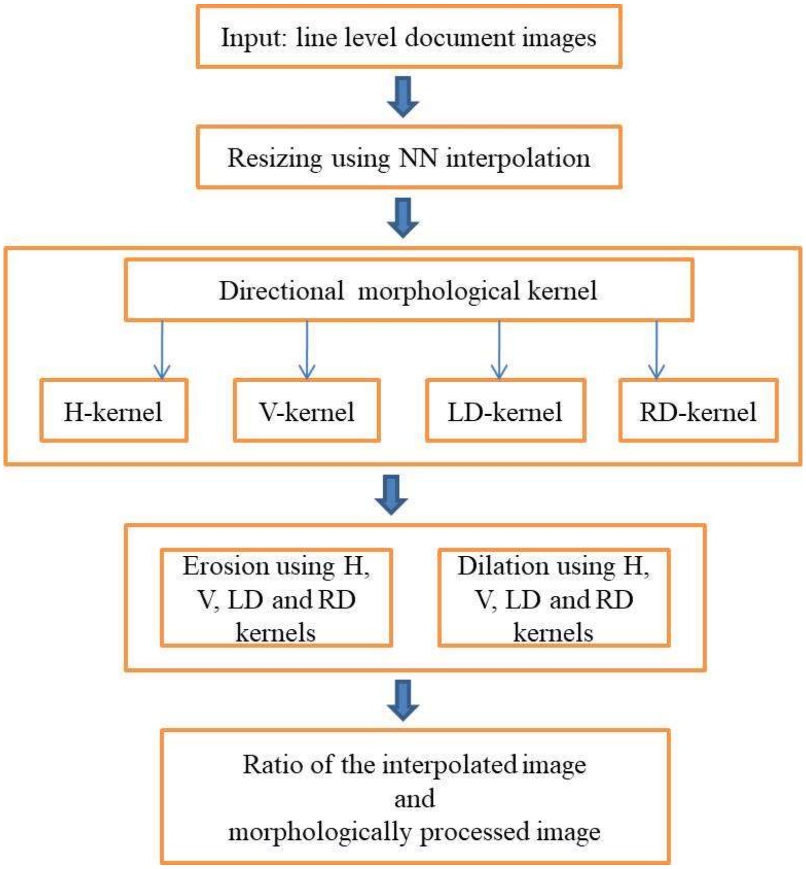
Input: line level document images
Resizing using NN interpolation
Directional morphological kernel
H-kernel
V-kernel
LD-kernel
RD-kernel
Erosion using H, V, LD and RD kernels
Dilation using H, V, LD and RD kernels
Ratio of the interpolated image and morphologically processed image

Fig. 10: General flow diagram of the computation of interpolation based feature

2.2.3.2 *Gabor energy*

Gabor filter has been widely used as texture feature for image analysis ([9]; [40]). Gabor filter response to an input image is computed by a two dimensional convolution operation by a sinusoidal wave (a plane wave for two dimensional Gabor filters). Let, an input image is *I(x,y)* and the Gabor filter response is *G(x, y, f, ϕ)*, where, *f* is frequency and *ϕ* is orientation. Then, G can be represented by the following equation:

$$G(x, y, f, \phi) = \iint I(p, q) g(x - p, y - q, f, \phi) dp\, dq \qquad (1)$$

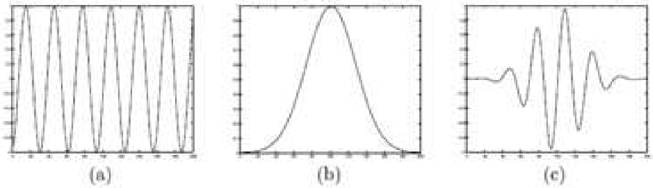

Fig. 11: Illustration of 1-D gabor filter [40]

Fig. 11 and 12 illustrate the graphical representation of one and two dimensional Gabor filter. In our experiment, to compute the feature vector, the value of f was chosen as 0.25. The ϕ values are taken as 60º, 90º, 120º and 150º. These values of f and ϕ were selected experimentally observing their suitability for our problem. So, a total of eight dimensional Gabor energy feature was generated for the present experiment.

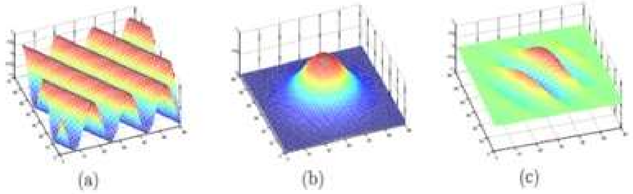

Fig. 12: Illustration of 2-D Gabor filter [40]

3.0 EXPERIMENTAL DETAILS

3.1 Dataset development

Different single script and multi-script dataset on Bangla, Roman, Devanagari, and Telugu are available at ([21]; [22]; [23]; [24]; [25]; [26]). An Urdu dataset can be found in [27], which includes isolated digits, numeral strings with/without decimal points, five special symbols, 44 isolated characters, and 57 Urdu words. IAM dataset is very popular Roman script dataset which consists of 1539 pages, 5685 sentences, 13353 lines, 115320 words distributed at the document, line, sentence and word level [28]. A Kannada script dataset consisting 204 documents, 4298 lines, 26115 words distributed over the document, line and word level is developed in the year 2011 [29]. In the same year a Devanagari script dataset at character level, both for alphabets, and numerals was developed [30]. An offline sentence dataset of Urdu handwritten documents along with preprocessing and segmentation techniques was reported in [31]. QUWI or Qatar University Writer Identification dataset [32] is an Arabic and Roman sentence level handwritten dataset build in the year 2013. The CVL Dataset is a public Roman script dataset for writer retrieval, writer identification and word spotting [33]. Tamil-DB is a popular and very useful handwritten city name dataset [34], was primarily developed for postal automation. A Bangla numeral dataset was reported at [35]. It contains about 8348 online numeral strings and 23392 offline isolated numerals. Many postal documents are considered here during the generation of this dataset. In [36], a dataset is reported which was built as a part of Tamil and Kannada handwriting recognition work.

So, it is clear that although different dataset were reported in literature, they were not intended for script identification. Almost all the dataset contains only one script or in very limited cases multiple scripts. That is why we have to collect and prepare our own dataset for experimentation. Table 2 shows the dataset distribution among different scripts considered for the present work. In total 2034 line level document images are collected. For Kannada script, KHTD [29] dataset was used. A standard data collection form was prepared where some of the forms contain pre-specified texts that were asked to be copied and some forms were left blank where users were asked to write anything as per their choice. The writing methodology was totally unconstrained, i.e. users were free to write in any ink color, writing direction, writing hand etc. More than 90% data was collected from the native speaker of the particular language. During data collection, we have considered different persons with varying sex (M/F), age (18-65 years), educational qualification (12th pass/ UG/ PG) to incorporate maximum realness among the data. There were few page-level images among the collected data. So, for line extraction our existing technique was used [15]. Document digitization was carried out using a flatbed scanner at 300 dpi and stored them as .jpg files. Our existing binarization technique was used to convert those images into binary form. Fig. 13 shows few sample line-level images from our dataset (top to bottom: Bangla, Devanagari, Kannada, Malayalam, Oriya, Roman, Telugu, and Urdu).

Table 2: Dataset distribution of eight scripts#

Script	Number of line-level images
Bangla	325
Devanagari	200
Kannada	332
Malayalam	100
Oriya	308
Roman	370
Telugu	90
Urdu	309
Total	**2034**

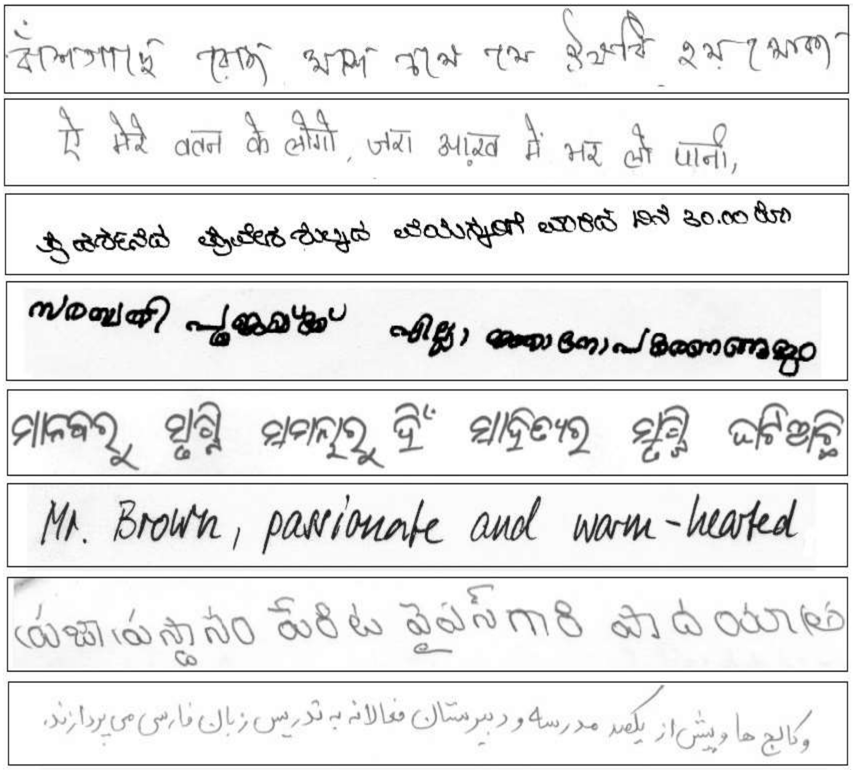

Fig. 13: Sample line-level document images from our prepared dataset.

# the dataset will be freely available upon request.

3.2 Setup and classification framework

The learning process of different properties of an individual script is known as training and the evaluation of the dissimilarity measure of the script classes are done at test phases. Different schemes can be found in literature for training and test dataset breakup. We have followed a breakup of 2:1 ratio of whole data as training and test samples. So, it means 66.6% training data and rests are for testing. To evaluate the success rate, the following equation is followed:

$$Success\_rate = \frac{\#correctly\_classified\_words}{\#total\_words} x100\% \qquad (2)$$

For the present work, we have followed a region-wise classification framework. In general, the multi-script documents are expected to be region wise rather than whole country wise. For example, though it is not impossible, it is unlikely that Oriya (eastern Indian script), and Kannada (southern Indian script) scripts will appear together in a multi-script document. But the proximity of being togetherness of Bangla-Oriya or Kannada-Malayalam is significantly higher in real life multi-script document, because their presence is region wise close. So, we observe that, a region-wise classification strategy is needed for better processing of real life multi-script Indian documents. The following section describes about the outcome of the test phase.

3.3 Classifiers

The classifiers applied in this work are multilayer perceptron (MLP), support vector machine (SVM), random forest (RF), and fuzzy unordered rule induction algorithm (FURIA). The motivation behind choosing these classifiers is their promising performance in our earlier work [10]. In the following section, we discuss briefly about these classifiers.

3.3.1 MLP

MLP consists of three layers with the number of neurons in each layer represented as a directed graph [3]. In our experiment, we choose the configuration of the MLP as 148-hl-8 (i.e. 148 is the number of attributes, hl is the number of hidden layers and output classes are 8). We empirically computed the number of neurons in the hidden layer hl, as a function of the dimensionality of the input size of the feature vector. Each neuron has a sigmoid transfer function and the network has been trained using the back propagation algorithm and the maximum epoch size was set at 500.

3.3.2 SVM

Support vector machine is one of the most widely used classifier in many areas of pattern recognition [47]. The classification performed by SVM is done by constructing a hyper plane in a high dimensional feature space. SVM can use different linear or non-linear kernels. Performance of SVM directly depends on the selection of particular kernel, the kernel's parameters, and the soft margin parameter C. We have used Linear SVM because of its promising and faster performance. We have done experiment using other non-linear kernels like: RBF kernel, and sigmoid function. But they did not improve performance of our task and also time consuming.

3.3.3 RF

RF is an ensemble classifier with a combination of many decision trees generating a random forest. The output of the class is the mode of the classes output by individual trees. For more detail, refer [43].

3.3.4 FURIA

FURIA is a fuzzy-rule-based classifier, used to obtain fuzzy rules. It is an extension of the well-known RIPPER algorithm. Instead of conventional rules and rule lists, FURIA learns fuzzy rules and unordered rule sets. Furthermore, it uses an efficient rule stretching scheme to deal with uncovered examples [44].

3.4 Result and analysis

To evaluate the performance of our method, we have used the state-of-the-art classifiers: MLP, SVM, RF and FURIA. The performance metrics considered in our work: average accuracy (AAR), model building time (MBT), true positive rate (TP rate), false positive rate (FP rate), false negative rate (FN), precision, recall, f-measure, and ROC area. A brief description of these metrics is discussed in following section.

AAR: Average accuracy (%) is the actual identification rate. It is measured by the equation (2).
MBT: Measured in second It is the total time to train the system.
TP Rate: True positive rate is the proportion of the test samples among all which were classified correctly to a target class at which they should belong.
FP Rate: It is a kind of "false alarm". False positive rate is the proportion of the test samples which belongs to a particular class but misclassified to a different class.
FN Rate: False negative rate provides the total misclassification rate, i.e. the proportion of samples among all which were misclassified to other classes.
Precision: It is defined as the proportion of the test samples which truly have classified to a particular class among all those which were classified to that class. So, *Precision = TP Number / (TP Number + FP Number)*. As an example, in table 3, weighted average precision is ((117/117) + (64/66) + (110/113) + (28/28) + (97/103) + (124/125) + (29/30) + (110/110))/8 = 0.981 (data obtained from table 5).
Recall: Recall is defined as follows: *Recall = TP Number / (TP Number + FN Number).* Here *FN Number* is the false negative number.
F-Measure: It is a combined measure of precision and recall. It is defined as: *F-Measure = 2 * Precision * Recall / (Precision + Recall).*

According to AAR (%) we found that, MLP > SVM > RF > FURIA. The identification result produced by MLP is higher (> 3.4% than SVM, > 5.7% than RF, and > 8% than FURIA) compared to other two. Being motivated by these results, for further experiments we have continued with MLP. The configuration of MLP for present experiment is followed as discussed in section 3.3.1.

Feature selection is discussed in Table 4. We computed the performance of three different features: f_1, f_2 and f_3 individually and collectively. The reported results are in the order: $f_2 > f_1 > f_3$. So, it implies that in the individual category, directional stroke feature perform relatively well compared to structural and texture features. Then we combined these features and the results show some improvement over individual categories. Comparing all the selection strategies we found: $f_1 + f_2 + f_3 > f_1 + f_2 > f_1 + f_3 > f_2 + f_3 > f_2 > f_1 > f_3$. So, we have continued the remaining experiments using combined features $f_1 + f_2 + f_3$.

Table 3: Comparison of the performance of different classifiers on the test dataset (all-script accuracy)

Classifier	AAR (%)	MBT (s)	TP rate	FP rate	FN rate	Precision	Recall	F-measure	ROC area
MLP	**98.2**	972.62	0.981	0.003	0.0195	0.981	0.981	0.981	0.999
SVM	94.8	0.88	0.948	0.008	0.0706	0.950	0.948	0.948	0.970
RF	92.5	0.87	0.925	0.012	0.0955	0.927	0.925	0.925	0.991
FURIA	90.2	23.9	0.902	0.017	0.1067	0.903	0.902	0.902	0.966

Table 4: Performance of different features individually and collectively using MLP classifier (all-script accuracy)

Feature combination	**AAR (%)**
f_1	89.4
f_2	95.5
f_3	86.8
$f_1 + f_2$	97.3
$f_1 + f_3$	96.2
$f_2 + f_3$	96.0
$\boldsymbol{f_1 + f_2 + f_3}$	**98.2**

Table 5: Confusion matrix using MLP classifier on the test dataset for eight-script combination

Script considered	Bangla	Devanagari	Kannada	Malayalam	Oriya	Roman	Telugu	Urdu
Bangla	117	0	0	0	0	0	0	0
Devanagari	0	64	0	0	1	0	0	1
Kannada	1	1	110	0	1	0	0	0
Malayalam	0	0	0	28	0	0	0	0
Oriya	3	0	1	0	97	1	0	1
Roman	0	0	0	0	1	124	0	0
Telugu	0	1	0	0	0	0	29	0
Urdu	0	0	0	0	0	0	0	110
Average eight-script accuracy: 98.2%								

The confusion matrix for all-script combination is shown in Table 5. All-script accuracy is 98.2% as shown in Table 6.. From Table 6, 100% accuracy is found for twelve combinations. 26 combinations show higher accuracy compared to the eight-script average accuracy. However, in two instances, namely Devanagari-Telugu and Bangla-Devanagari show 0.2% and 0.4% lower than the eight-script average accuracy. The Bangla-Devanagari combination result is due to some similar features in their writing style (presence of topological feature like 'matra' in both cases). The Devanagari-Telugu combination produces discouraging results due to the presence of few structural similarities like the concavity direction of the graphemes of these two scripts. We need to investigate this issue and some script dependent features need to be developed to resolve the issue.

Table 6: Bi-script accuracy using MLP classifier. ($^8C_2$ bi-script combinations are considered)

Scripts considered	**Accuracy (%)**	**Scripts considered**	**Accuracy (%)**
{Bangla, Roman}	100	{Devanagari, Roman}	99.5
{Bangla, Urdu}	100	{Devanagari, Urdu}	99.5
{Devanagari, Kannada}	100	{Bangla, Malayalam}	99.4
{Kannada, Telugu}	100	{Roman, Telugu}	99.4
{Kannada, Urdu}	100	{Bangla, Telugu}	99.3
{Malayalam, Oriya}	100	{Kannada, Roman}	99.2
{Malayalam, Roman}	100	{Oriya, Roman}	99.2
{Malayalam, Telugu}	100	{Bangla, Oriya}	99.1
{Malayalam, Urdu}	100	{Devanagari, Malayalam}	99.1
{Oriya, Telugu}	100	{Devanagari, Oriya}	98.9
{Oriya, Urdu}	100	{Kannada, Malayalam}	98.7
{Telugu, Urdu}	100	{Kannada, Oriya}	98.2
{Bangla, Kannada}	99.6	{Devanagari, Telugu}	98
{Roman, Urdu}	99.6	{Bangla, Devanagari}	97.8

3.5 Eastern Indic scripts

The major eastern Indian states are West Bengal, Bihar, Jharkhand, Orissa, and North-East India. Four scripts namely Bangla, Devanagari, Oriya and Roman are mainly used by the Eastern Indian. Table 7 shows the confusion matrix where the maximum misclassification occurs between Bangla and Devanagari.

Table 7: Confusion matrix using MLP classifier on the test dataset for eastern Indian scripts

Script considered	Bangla	Devanagari	Oriya	Roman
Bangla	106	1	1	0
Devanagari	3	76	0	0
Oriya	0	2	96	1
Roman	1	1	0	121
Average eastern Indian accuracy: 97.6%				

One of the reasons for this misclassification is due to the presence of common feature like 'matra' between them. The average accuracy is 97.6% when considering all the four scripts. The tri-script performance is shown in Table 8, where the average bi-script (as obtained from Table 6) and tri-script accuracy are 99.1% and 98.4% respectively.

Table 8: Tri-script accuracy for eastern Indian scripts

Tri-script combinations	**Accuracy (%)**
{Bangla, Devanagari, Roman}	99.1
{Devanagari, Oriya, Roman}	98.7
{Bangla, Oriya, Roman}	98.6
{Bangla, Devanagari, Oriya}	97.2
Average tri-script	**98.4**

3.6 North Indic scripts

Out of the eight scripts being considered , Devanagari, Roman, and Urdu belong to the North Indian states. Table 9 shows the confusion matrix using the MLP classifier on the test dataset and it reports 98.4% average accuracy for all scripts. Here, misclassification mostly occurs between the Devanagari and Urdu. The bi-script combinations produced an average bi-script accuracy of 99.5% (as obtained from Table 6).

Table 9: Confusion matrix using MLP classifier on the test dataset for north Indic scripts

Script considered	Devanagari	Roman	Urdu
Devanagari	65	0	1
Roman	0	125	0
Urdu	4	0	104
Average all-script accuracy: 98.4%			

3.7 South Indic scripts

We have four south Indian scripts: Kannada, Malayalam, Roman and Telugu. Table 10 shows the confusion matrix on the test dataset using MLP classifier and an average accuracy of 99.1% was obtained. Misclassification occurs between Malayalam and Roman as shown in Table 11. We have total $^6C_2$ bi-script and $^6C_3$ tri-script combinations for experiment. In three bi-script cases, namely ({Kannada, Telugu}, {Malayalam, Roman} and {Malayalam, Telugu}) we obtained 100% accuracy. Similarly, for two tri-script cases namely ({Kannada, Roman, Telugu} and {Malayalam, Roman, Telugu}) we also obtained 100% accuracy. Finally, the average bi-script (as obtained from Table 6) and tri-script accuracy was 99.6% and 99.8% respectively.

Table 10: Confusion matrix using MLP classifier on the test dataset for south Indian scripts

Script considered	Kannada	Malayalam	Roman	Telugu
Kannada	109	0	0	0
Malayalam	0	25	0	0
Roman	0	3	130	0
Telugu	0	0	0	36
Average all-script accuracy: 99.1%				

Table 11: Tri-script accuracy for south Indian scripts

Script combinations	**Accuracy (%)**
{Kannada, Roman, Telugu}	100
{Malayalam, Roman, Telugu}	100
{Kannada, Malayalam, Telugu}	99.5
{Kannada, Malayalam, Roman}	98.6
Average tri-script	**99.8**

3.8 Scripts with ‘matra’ vs. without-‘matra’

Separation of scripts with ‘matra’ from their counterpart can be used as a precursor to the script identification work. A ‘matra’ is a horizontal line joining characters of a word or words of a sentence. It is a distinguishable topological property of scripts like Bangla and Devanagari. In our dataset, there are two scripts with ‘matra’ and six scripts without ‘matra’. Table 12 shows the confusion matrix of scripts with ‘matra’ and without ‘matra’.

Table 12: Confusion matrix using MLP classifier on the test dataset for scripts with ‘matra’ and without ‘matra’

Script Considered	Scripts with ‘matra’ *(Bangla, Devanagari)*	Scripts without ‘matra’ *(Kannada, Malayalam, Oriya, Roman, Telugu, Urdu)*
Scripts with ‘matra’ *(Bangla, Devanagari)*	175	8
Scripts without ‘matra’ *(Kannada, Malayalam, Oriya, Roman, Telugu, Urdu)*	6	503
Average accuracy of scripts with ‘matra’ vs. without ‘matra’ : 98%		

3.9 Dravidian vs. non-Dravidian scripts

In this experiment we have differentiated all the scripts into two groups based on their originality that is Dravidian and non-Dravidian. In our dataset, Kannada, Malayalam and Telugu belong to Dravidian group, and the rest are non-Dravidian. In our test, we found an average accuracy of 98.9% for these two groups as shown in Table 13.

Table 13: Confusion matrix using MLP classifier on the test dataset for scripts with ‘matra’ and without ‘matra’

Script considered	Dravidian group *(Kannada, Malayalam, Telugu)*	Non-Dravidian group *(Bangla, Devanagari, Oriya, Roman, Urdu)*
Dravidian group *(Kannada, Malayalam, Telugu)*	175	7
Non-Dravidian group *(Bangla, Devanagari, Oriya, Roman, Urdu)*	1	509
Average accuracy of Dravidian and non-Dravidian groups: 98.9%		

3.10 Comparative study

Different methods described in [19], [41] and [42] have been evaluated on the present dataset at 8-script scenario. All these techniques are experimented in the same setup, i.e. using Matlab 7.6.0 software, a machine with Intel core i3 2.13GHz processor and 4 GB memory. From the experiment (results are shown in Table 14), we found that, as per recognition accuracy of the features: ***Proposed method*** > DWT+RT > GLCM > WE. So, the proposed features perform significantly better compared to state-of-the-art methods.

Table 14: Comparison of script identification results (all-scripts accuracy) with state-of-the-art methods in same setup

Methods	**Line-level scripts considered**	**Accuracy (%) (all-script)**
Busch et al. [41]	Bangla, Devnagari, Kannada, Malayalam, Oriya, Roman, Telugu and Urdu	94.2
Singh et al. [19]		95.3
Obaidullah et al. [42]		97.6
Proposed method		**98.2**

4.0 CONCLUSION & FUTURE PLANS

Scripts identification is a well-studied document image analysis problem in the literature. But the complete solution for HSI is still lacking. In this paper a line-level HSI approach for multi-script document of eight official Indic scripts is proposed. Combining different script dependent and independent features, we propose a generic solution to the current problem. The proposed method is invariant to moderate skew and noise that normally appears in real life documents. The eight-script and bi-script average accuracy has been found to be 98.2% and 99.5%. Our method is compared with the state-of-the-art and it outperforms over them on the same setup (i.e. on same dataset and system). In a nutshell, the key contribution of this work can be summarized as follows:

(i) We addressed the real life HSI problem for eight different Indic scripts and combining features to propose a generic solution.

(ii) We analyzed the performance of the script classification based on different regions and regional groups in India.

(iii) Evaluated different state-of-the-art classifiers: MLP, SVM, FURIA and RF.

(iv) The script dataset used in this work will be available upon request for handwritten script identification.

Our immediate step is to evaluate our approach using more number of scripts and more number of samples per script. The research scope can be further extended towards the real life script identification problems: video based script identification, script identification from scene images, and character-level script identification from multi-script artistic words. Improving the accuracy of our system by integrating multiple classifiers is also in the pipeline.

REFERENCES

[1] S. Bag and G. Harit, "A Survey on Optical Character Recognition for Bengali and Hindi Documents", *Sadhana-Academy Proceedings in Engineering Science, IAS & Springer,* Vol. 38, No. 1, 2013, pp. 133–68.

[2] D. Ghosh, T. Dube and S. P. Shivprasad, "Script Recognition – A Review", *IEEE Transactions on Pattern Analysis & Machine Intelligence*, Vol. 32, No. 12, 2010, pp. 2142–61.

[3] S. M. Obaidullah, S. K. Das and K. Roy, "System for Handwritten Script Identification from Indian Document", *Journal of Pattern Recognition Research*, Vol. 8, No. 1, 2013, pp. 1-12.

[4] Patil, B. and N. V. Subareddy, "Neural Network Based System for Script Identification in Indian Scripts" *Sadhana-Academy Proceedings in Engineering Science, IAS & Springer*, Vol. 27, No. 1, 2002, pp. 83–97.

[5] U. Pal and B. B. Chaudhuri, "Identification of Different Script Lines from Multi-Script Documents", *Image & Vision Computing*, Vol. 20, No.13-14, 2002, pp. 945–54.

[6] J. Hochberg, P. Kelly, T. Thomas and L. Kerns, "Automatic Script Identification from Document Images Using Cluster-Based Templates", *IEEE Transaction on Pattern Analysis and Machine Intelligence,* Vol. 19, 1997, pp. 176–81.

[7] S. Chaudhury, G. Harit, S. Madnani, and R. B. Shet, "Identification of Scripts of Indian Languages by Combining Trainable Classifiers", in *Indian Conference on Computer Vision, Graphics and Image Processing*, Bangalore, India, 2000.

[8] U. Pal and B. B. Chaudhuri, "Script Line Separation from Indian Multi-Script Documents", *IETE Journal of Research*, Vol. 49, No. 3–11, 2003.

[9] P. B. Pati and A. G. Ramakrishnan, "Word-Level Multi-Script Identification", *Pattern Recognition Letters*, Vol. 29, No. 9, 2008, pp. 1218–29.

[10] S. M. Obaidullah, A. Mondal, N. Das, and K. Roy, "Script Identification from Printed Indian Document Images and Performance Evaluation Using Different Classifiers", *Applied Computational Intelligence and Soft Computing*, Vol. 2014, Article ID. 896128, 2014, pp. 12 pages.

[11] S. Kavitha, P. Shivakumara, G. H. Kumar and C. L. Tan, "A Robust Script Identification System for Historical Indian Document Images", in *Malaysian Journal of Computer Science,* Vol. 28, No. 4, 2015, pp. 283–300.

[12] J. Hochberg, K. Bowers, M. Cannon, and P. Kelly, "Script and Language Identification for Handwritten Document Images", *International Journal of Document Analysis & Recognition*, Vol. 2, No. 2/3, 1999, pp. 45–52.

[13] L. Zhou, Y. Lu and C. L. Tan, "Bangla/English Script Identification Based on Analysis of Connected Component Profiles", in *International Workshop on Document Analysis Systems*, 2006, Nelson, New Zealand, pp. 243–54.

[14] V. Singhal, N. Navin and D. Ghosh, "Script-Based Classification of Hand-Written Text Documents in a Multi-Lingual Environment", in *International Workshop on Research Issues in Data Engineering: Multi-lingual Information Management*, 2003, pp. 47–54.

[15] K. Roy, A. Banerjee and U. Pal, "A System for Word Wise Handwritten Script Identification for Indian Postal Automation", in *IEEE India Annual Conference*, India, 2004, pp. 266–71.

[16] M. Hangarge and B. V. Dhandra, "Offline Handwritten Script Identification in Document Images", *International journal of Computer Application*, Vol. 4, No. 6, 2010, pp. 6–10.

[17] M. Hangarge, K. C. Santosh and R. Pardeshi, "Directional Discrete Cosine Transform for Handwritten Script Identification", in *International Conference on Document Analysis and Recognition*, Washington, USA, 2013, pp. 344–48.

[18] R. Pardeshi, B. B. Chaudhuri, M. Hangarge and K. C. Santosh, "Automatic Handwritten Indian Scripts Identification", in *International Conference on Frontiers in Handwriting Recognition*, Greece, 2014, pp. 375–80.

[19] P. K. Singh, S. K. Dalal, R. Sarkar and M. Nasipuri, "Page-level script identification from multi-script handwritten documents", in *International Conference Computer, Communication, Control and Information Technology*, Kolkata, India, 2015, pp. 1-6.

[20] S. M. Obaidullah, C. Halder, N. Das and K. Roy, "An Approach for Automatic Indic Script Identification from Handwritten Document Images", in *Doctoral Symposium on Applied Computation and Security Systems,* Kolkata, India, 2015, pp. 37-51.

[21] R. Sarkar, N. Das, S. Basu, M. Kundu, M. Nasipuri and D. K. Basu, "CMATERdb1: A Dataset of Unconstrained Handwritten Bangla and Bangla–English Mixed Script Document Image", *International Journal on Document Analysis and Recognition*, Vol. 15, No. 1, 2012, pp. 71–83.

[22] S. Basu, C. Chaudhury, M. Kundu, M. Nasipuri and D. K. Basu, "Text Line Extraction from Multi Skewed Handwritten Documents", *Pattern Recognition*, Vol. 40, No. 6, 2007, pp. 1825–39.

[23] N. Das, S. Basu, R. Sarkar, M. Kundu and M. Nasipuri, "Handwritten Bangla Compound Character Recognition: Potential Challenges and Probable Solution", in *Indian International Conference on Artificial Intelligence, Bangalore*, 2009, pp. 1901–13.

[24] N. Das, R. Sarkar, S. Basu , M. Kundu, M. Nasipuri and D. K. Basu, "A Genetic Algorithm Based Region Sampling for Selection of Local Features in Handwritten Digit Recognition Application", *Applied Soft Computing*, Vol. 12, No. 5, 2012, pp. 1592–1606.

[25] N. Das, J. M. Reddy, R. Sarkar, S. Basu, M. Kundu, M. Nasipuri and D. K. Basu, "A Statistical–topological Feature Combination for Recognition of Handwritten Numerals", *Applied Soft Computing*, Vol. 12, No. 8. 2012, pp. 2486–95.

[26] CAMTERDB, "The JU Image Dataset", at *https://code.google.com/p/cmaterdb/* last accessed 20.10.2016

[27] M. W. Saqheer, C. L. He, N. Nobile and C. Y. Suen, "A New Large Urdu Dataset for Off-Line Handwriting Recognition", in *International Conference on Image Analysis and Processing*, Vietri sul Mare, Italy, 2009, pp. 538–46.

[28] U. Marti and H. Bunke, "An English Sentence Dataset for Off-Line Handwriting Recognition", *International Journal on Document Analysis and Recognition*, Vol. 5, No. 1, 2002, pp. 39 – 46.

[29] A. Aleai, P. Nagabhushan and U. Pal, "A Benchmark Kannada Handwritten Document Dataset and Its Segmentation", in *International Conference on Document Analysis and Recognition,* Beijing, China, 2011, pp. 140–45.

[30] V. J. Dongre and V. H. Mankar, "Development of Comprehensive Devnagari Numeral and Character Dataset for Offline Handwritten Character Recognition", *Applied Computational Intelligence and Soft Computing*, Vol. 2012, Article ID. 871834, 2012, pp. 5 pages.

[31] A. Raza, I. Siddiqi, A. Abidi and F. Arif, "An Unconstrained Benchmark Urdu Sentence Dataset with Automatic Line Segmentation", in *International Conference on Frontiers in Handwriting Recognition*, Bari, Italy, 2012, pp. 491–96.

[32] A. Raza, I. Siddiqi, A. Abidi and F. Arif, "QUWI: An Arabic and English Handwriting Dataset for Offline Writer Identification", in *International Conference on Frontiers in Handwriting Recognition*, 2012, Bari, Italy, pp. 746–51.

[33] M. Diem, S. Fiel, F. Kleber and R. Sablatnig, "CVL-Dataset: An Off-Line Dataset for Writer Retrieval, Writer Identification and Word Spotting", in *International Conference on Document Analysis and Recognition*, Washington, USA, 2013, pp. 560–64.

[34] S. Thadchanamoorthy, N. D. Kodikara, H. L. Premaretne, U. Pal and F. Kimura, "Tamil Handwritten City Name Dataset Development and Recognition for Postal Automation", in *International Conference on Document Analysis and Recognition*, Washington, USA, 2013, pp. 793–97.

[35] B. B. Chaudhuri, "A Complete Handwritten Numeral Dataset of Bangla-A Major Indic Script", in *International Workshop on Frontiers of Handwriting Recognition,* La Baule, France, 2006, pp. 379–84.

[36] B. Nethravathi, C. P. Archana, K. Shashikiran, A. G. Ramakrishnan and V. Kumar, "Creation of a Huge Annotated Dataset for Tamil and Kannada OHR", in *International Workshop on Frontiers of Handwriting Recognition*, Istambul, Turkey, 2010, pp. 415–20.

[37] B. B. Mandelbrot, *The Fractal Geometry of Nature (New York: Freeman)*, 1982.

[38] K. Roy, S. K. Das and S. M. Obaidullah, "Script Identification from Handwritten Document", in *National Conference on Computer Vision, Pattern Recognition, Image Processing and Graphics,* Hubli, India, 2011, pp. 66–69.

[39] A. Kaehler and G. R. Bradski, *Learning OpenCV,* 2008.

[40] S. M. Obaidullah, N. Das and K. Roy, "Convolution Based Technique for Indic Script Identification from Handwritten Document Images", in *International Journal of Image, Graphics and Signal Processing,* Vol. 7, No. 5, 2015, pp. 49-57.

[41] A. Busch, W. W. Boles and S. Sridharan, "Texture for script identification", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 27, No. 11, 2005, pp. 1720-1732.

[42] S. M. Obaidullah, C. Halder, N. Das and K. Roy, "A New Dataset of Word-level Offline Handwritten Numeral Images from Four Official Indic Scripts and Its Benchmarking using Image Transform Fusion", *International Journal of Intelligent Engineering Informatics*, Vol. 4, No. 1, 2016, pp. 1-20.

[43] M. Moohebat, R.G. Raj, D. Thorleuchter and S. Abdul-Kareem. "Linguistic Feature Classifying and Tracing". Malaysian Journal of Computer Science, Vol. 30, No.2, 2017, pp 77-90.

[44] J. C. Huehn and E. Huellermeier, "FURIA: An Algorithm for Unordered Fuzzy Rule Induction", *Data Mining and Knowledge Discovery*, Vol. 19, 2009, pp. 293-319.

[45] S. M. Obaidullah, C. Goswami, K. C. Santosh, C. Halder, N. Das and K. Roy, "Separating Indic scripts with `shirorekha' -- a precursor to script identification in multi-script documents", in *IAPR International Conference on Computer Vision & Image Processing* (CVIP), Roorkee, India, 2016.

[46] S. M. Obaidullah, C. Halder, N. Das and K. Roy, "Bangla and Oriya Script Lines Identification from Handwritten Document Images in Tri-script Scenario", *International Journal of Service Science, Management, Engineering, and Technology*, Vol. 7, No. 1, 2016, pp. 43-60.

[47] C. C. Chang and C.J. Lin "LIBSVM : a library for support vector machines", *ACM Transactions on Intelligent Systems and Technology*, Vol. 2, No. 3, Article 27, 2011, pp. 27 pages.