Sādhanā (2019)44:178
https://doi.org/10.1007/s12046-019-1159-0

An approach based on classifier combination for online handwritten text and non-text classification in Devanagari script

RAJIB GHOSH*, SAURAV SHANU, SUGANDHA RANJAN and KHUSBOO KUMARI

Department of Computer Science and Engineering, National Institute of Technology Patna, Patna, India
e-mail: rajib.ghosh@nitp.ac.in

MS received 24 January 2019; revised 29 April 2019; accepted 22 May 2019

Abstract. In this article, a method of analysing features of elliptical regions and combining outcomes of classifiers using Dempster–Shafer Theory (DST) is presented to classify online handwritten text and non-text data of any online handwritten document in the most popular Indic script—Devanagari. Although a few works exist in this regard in different non-Indic scripts, to our knowledge, no study is available to classify handwritten text and non-text document in online mode in any Indic script. The present method uses various structural and directional features analysed in elliptical regions to extract feature values from strokes of text and non-text data. The features are then studied separately in classification platforms based on Support Vector Machine (SVM) and Hidden Markov Model (HMM). The probabilistic outcomes of these two classification platforms are then combined using DST to improve the system performance. The efficiency of the present system has been measured on a self-generated dataset and it provides promising result.

Keywords. Online handwriting; text/non-text classification; SVM; HMM; classifier combination; DST.

1. Introduction

It is very natural for human beings to write a document consisting of textual and non-textual information. Dividing one online handwritten document into its constituent textual and non-textual portions is a very crucial problem for document analysis and recognition tasks. The task of text/non-text classification from within a single online handwritten document is required for text recognition, text searching and diagram identification. Hence, research explorations have been started recently to solve this crucial problem. After classifying text and non-text information from within a single document, the textual information can be passed to a text recognizer module to recognize the textual information of the document. Similarly, various graphical entities such as flow chart, transition diagram, 2-D graphical objects, etc. are recognized as non-text portions.

Handwritten texts are not written in a uniform manner as a specific text can be written in dissimilar sizes and styles by different writers or even a single writer. Extracting features to classify handwritten text/non-text is a challenging task as several handwritten textual strokes may have some feature values common to those of non-textual strokes like the one shown in figure 1. Very few research works [1–4] are available in the literature to classify text and non-text portions within a single online handwritten document. However, we could not find any research work on online handwritten text/non-text classification where text is written in any Indic script, including Devanagari. The proposed system classifies online handwritten text and non-text document where text data are written in the most popular Indic script—Devanagari. Figure 2 shows the combination of text and non-text data in Devanagari script present in a single document.

In the proposed system, in order to extract features, each text and non-text data is divided into smaller elliptical regions by constructing several concentric ellipses around the stroke. Each elliptical region is further divided into several sub-regions before extracting various structural and directional features of stroke portions from each sub region. The features are then studied separately in classification platforms based on Support Vector Machine (SVM) and Hidden Markov Model (HMM) to classify text and non-text data. The outcomes of these two classifiers are then combined using Dempster–Shafer Theory (DST) to improve the performance of the present system. Figure 3 shows the overall framework of the proposed system.

The remaining portions of this paper are arranged as follows. Section 2 details the related works. Section 3 discusses the Devanagari script and method of dataset creation. Section 4 depicts the feature extraction technique. The processes of classifying text and non-text data separately using SVM and HMM as well as by combining them together are presented in section 5. Experimental results

*For correspondence
Published online: 12 July 2019

Text		Non-text	
Stroke	Character	Stroke	Graphical object
			Circle
			Rectangle

Figure 1. Same stroke used to write a character (text) in Devanagari script as well as non-textual object.

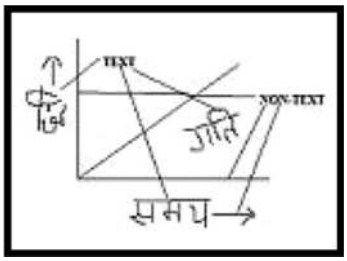

Figure 2. A combination of text and non-text data in Devanagari script.

and analysis are discussed in section 6. Finally, conclusion of the paper is given in section 7.

2. Literature survey

As already mentioned, although limited studies are available on classification of text and non-text portions from within a single online handwritten document in a few non-Indic scripts, we could not find any research work in this problem area in any of the Indic scripts. A few studies, available in non-Indic scripts in this problem area, are discussed here.

Delaye and Lee [1] proposed a method for segmentation of text/non-text portions in online handwritten documents. The strategy relied on single linkage clustering and a pairwise stroke distance measurement. Delaye and Liu [2] also presented a conditional random field (CRF)-based study in this regard in online handwritten documents. VanPhan and Nakagawa [3] proposed a method in this problem area based on advanced version of Recurrent Neural Network (RNN)—Long Short-Term Memory (LSTM). The study integrated local context and global context to enhance the system performance. Zhou and Liu [4] proposed a Markov Random Field (MRF)-based approach to classify text and non-text in Japanese documents. Zhou *et al* [5] presented another study, which focused on extracting text lines from online handwritten documents of both text and non-text matters. It considered temporal and spatial dimensions of online strokes through several partitioning and combining steps. The goal of the work presented by Liwicki *et al* [6] was very similar but did not handle non-textual strokes. Another study [7] presented the text line segmentation problem as stroke partitioning problem. In this work, the model complexity was optimized using gradient descent. Blanchard and Artieres [8] presented a system where probabilistic feature grammars were trained to detect text lines. Although this system has shown reasonable robustness for poorly structured documents, it cannot handle free-form documents properly due to the rule-based nature of the system. In our earlier work [9], an RNN-based recognition method using horizontal zoning of words was proposed to recognize online handwritten isolated words in two different Indic scripts—Devanagari and Bengali. In another study [10], an approach based on deep learning was presented for printed scene text recognition in three different Indic scripts—Devanagari, Telugu and Malayalam.

A few attempts [11, 12] have been reported in the literature to recognize only online hand-drawn circuit or sketched diagrams, i.e. only non-textual document. Feng *et al* [11] proposed a method for online hand-drawn electric circuit diagram recognition. The authors relied upon a two-dimensional dynamic programming technique to generate symbol hypothesis, which leads to correct segmentation and recognition of interspersed symbols. In another study [12], an attempt was made to develop a system to recognize online sketched diagrams, where an approach was proposed to deal with text blocks also present in diagrams.

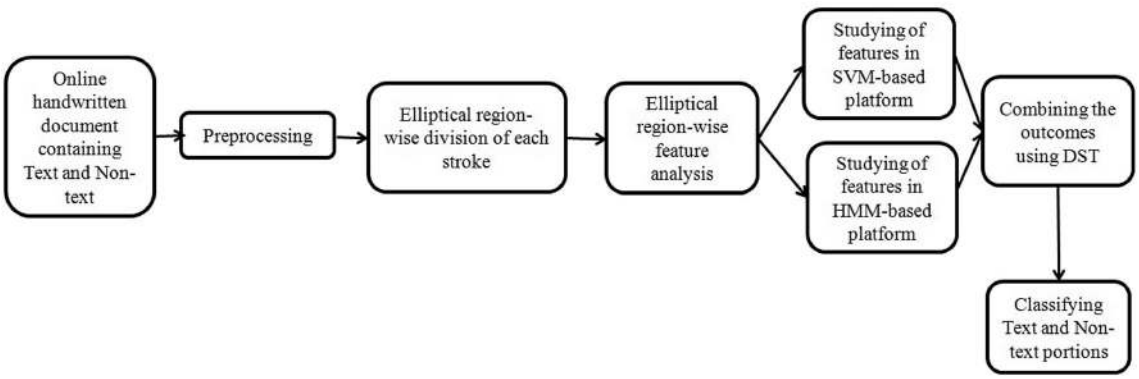

Figure 3. Overall framework of the proposed system.

Hence, a few attempts have been made to date to classify text/non-text portions existing in a single online handwritten document written in various non-Indic scripts. However, we could not find any research work in this problem area for Indic scripts.

3. Devanagari script and dataset details

Among Indic scripts, Devanagari is the most popular Indic script [13]. This script is used to write many Indo-Aryan languages such as Hindi, Nepali, Marathi, etc. The order of writing in this script is from left to right and it does not contain the upper/lower case notation of alphabets. Simple characters in this script consist of vowels and consonants. Current form of this script contains 12 vowels and 36 consonants. In this script, more than one consonants are combined to generate a compound character. After analysing this script, 63 compound characters have been identified. Besides this, Devanagari script contains 10 basic numerals (0–9) and matras. Figure 4 shows a few instances of simple and compound characters, numerals and matras in Devanagari script. This figure demonstrates that almost all characters shown here in Devanagari script have a horizontal line, known as *shirorekha*, at the upper part.

This work has created datasets of both online handwritten text and non-text documents as they are not publicly available. Samples of text and non-text data have been collected separately. Text strokes are written in Devanagari script. The generated datasets will be published for the future researchers.

For generating the datasets of text and non-text samples, we have collected samples of 100 different online handwritten text documents, each containing a character or word or digit and 100 different online hand-drawn diagrams for non-text data. We considered different diagrams like finite automata, flow charts, 2-D and 3-D shapes, various forms of 2-D graphs and object transformation in computer graphics as non-text data; 100 native speakers of Devanagari script of different ages and educational backgrounds provided handwriting samples. Each writer provided two samples of each text and non-text document. The training and testing phases in the present system have been conducted using holdout and 4-fold evaluation methods. In holdout method, the collected samples have been divided into training and testing datasets in 3:1 ratio. The dataset details for holdout method are presented in table 1.

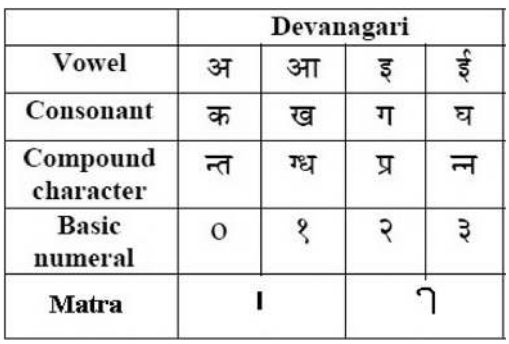

Figure 4. A few instances of different types of characters, basic numerals and matras in Devanagari script.

Table 1. Dataset details.

Document	Total samples	Training dataset size	Testing dataset size
Text	20000	15000	5000
Non-text	20000	15000	5000

4. Feature extraction

In the proposed system, various structural and directional features are extracted from stroke(s) of text and non-text data. The features used in this work are *writing direction*, *slope*, *curvature* and *curliness* [14]. These features can exploit the temporal information inherent to online data very efficiently. These features are extracted in the vicinity of each point x_t, y_t of the stroke. The methods of extracting these features are discussed here briefly.

a. *Writing direction*: The writing direction of a point x_t, y_t is computed using (1) and (2):

$$\cos\alpha = \frac{\Delta x}{\Delta s} \tag{1}$$

$$\sin\alpha = \frac{\Delta y}{\Delta s} \tag{2}$$

where

$$\Delta s = \sqrt{\Delta x^2 + \Delta y^2}, \tag{3}$$

$$\Delta x = x_{t-1} - x_{t+1}, \tag{4}$$

$$\Delta y = y_{t-1} - y_{t+1}. \tag{5}$$

b. *Slope*: The slope of point x_t, y_t is computed as the cosine of angle θ_t of the straight line from starting point of the vicinity to the last vicinity point.

c. *Curvature*: The curvature of point x_t, y_t is computed using sine and cosine of angle β with the help of two non-immediate neighbour points of x_t, y_t, i.e., x_{t-2}, y_{t-2} and x_{t+2}, y_{t+2}. It is computed using (6) and (7):

$$\cos\beta = \cos\alpha_{t-1} \times \cos\alpha_{t+1} + \sin\alpha_{t-1} \times \sin\alpha_{t+1}, \tag{6}$$

$$\sin\beta = \cos\alpha_{t-1} \times \sin\alpha_{t+1} - \sin\alpha_{t-1} \times \cos\alpha_{t+1}. \tag{7}$$

d. *Curliness*: This feature is computed by dividing the length of the vicinity by maximum side of the bounding

box containing all points in the vicinity of point x_t, y_t. Calculation of this feature is shown in (8):

$$C = \frac{L}{\max(\Delta x, \Delta y)} - 2 \tag{8}$$

where L denotes the length of the vicinity. Δx and Δy are the width and height, respectively, of the bounding box.

Before extracting features, entire text and non-text data samples are divided into smaller elliptical regions by constructing several concentric ellipses around the data. Each elliptical region is further divided into octants (tested with halves and quadrant also, but the one using octants has provided the best performance) before extracting feature values of stroke portions from each sub-region. This type of elliptical division is done to reduce inter-class similarity. The idea of segmentation into elliptical regions is illustrated in figure 5. Segmentations into elliptical regions of an online handwritten character (text data) and hand-drawn rectangle (non-text data) are illustrated in figure 6. Here, division is shown for two concentric ellipses.

The feature values obtained from these features from each sub-region are quantized separately into one of the 8 possible values. For example, if the angular values of first three features lie between 0° and 45° then the point x_t, y_t, in the vicinity of which features are extracted, is placed in bin 1, if angular values lie between 46° and 90° then this point is placed in bin 2 and so on up to bin 8. Different bin divisions have been tested, but the one using *pi*/4 has shown the best accuracy. Similarly, 8-level quantization has been performed on the feature values obtained from the fourth feature also. Thus, each bin holds a certain number of points. This number in each bin is divided by the total number of points of a stroke present in a sub-region. Hence, we get 8 normalized feature values ranging from 0 to 1 from 8 bins for each feature in a particular sub-region. Thus, 32 feature values are obtained from all four features in a particular sub-region. Feature vectors of different dimensions are generated depending upon the number of sub-regions created. For 3 concentric ellipses, 24 sub-regions are created; thus, 768 (32×24) feature values are obtained from 24 different sub-regions. For 4 concentric ellipses, 32 sub-regions are created; thus, 1024 (32×32) feature values are obtained from 32 different sub-regions.

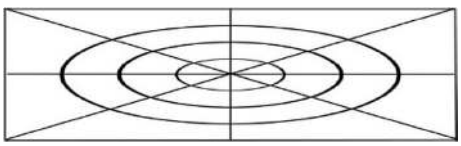

Figure 5. Division into elliptical regions where each elliptical region is divided into octants.

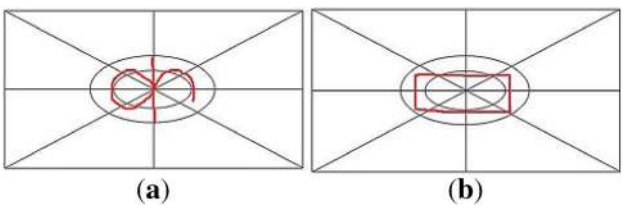

Figure 6. Division into elliptical regions of (**a**) one character (text data) in Devanagari script and (**b**) one rectangle (non-text data).

5. Classification

The present work classifies text and non-text data using SVM and HMM classifiers separately as well as by combining the probabilistic outputs of these two classifiers using DST. The methods of classification using SVM and HMM are discussed here.

5.1 *Classification using SVM*

In the recent past, SVM has been successfully used for various problems on pattern recognition as well as regression [15–17]. The theoretical background of SVM may be found in [15–17]. The present research work is a binary classification problem. To train the present system using SVM, two different class labels are used—one for text data and another for non-text data. During training, initially, feature values are extracted from stroke portions lying in each sub-region of elliptical division. Next, these feature values are quantized and normalized to generate the feature vector of same dimensionality for each training sample of both text and non-text data and are labelled with the appropriate class label in a single training file. During testing, a single test file containing feature vectors of both text and non-text data samples is fed to the SVM to know the label of each test feature vector.

5.2 *Classification using HMM*

HMM is a stochastic sequential classifier and has become popular in modelling temporal sequences [18]. Recognition of the sequences is performed using the Viterbi algorithm [18]. In the proposed system, two different HMMs are used—one for text data and another for non-text data. For each text and non-text data sample, sub-region-wise feature vectors are generated and the resultant sequence of feature vectors is processed using left-to-right continuous density HMMs. For each sequence of feature vectors, the likelihood of belongingness of the sequence to each class is calculated, and the class with which the maximum likelihood is found is considered for the final class label of the sequence. Figure 7 shows a few text and non-text samples that get modelled using HMMs.

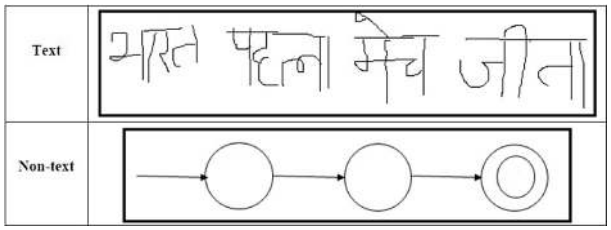

Figure 7. Some examples of text/non-text modelling using HMMs.

5.3 *Combining classifiers using DST*

In this work, probabilistic outcomes of SVM and HMM classifiers are combined using DST. DST, or evidence theory, is a general framework to deal with uncertainty. In this theory, a degree of belief is obtained after combining evidences from multiple sources. This combination considers all the available evidences. It is effectively used for combining multiple information sources with incomplete, imprecise, biased and conflict knowledge [19]. A DST-based combination approach is discussed here briefly.

In DST-based combination, a finite set $F = \{c_1, ..., c_n\}$, also known as a frame, is formed by exclusive class labels of various patterns. Here, n is total number of classes. In the present system, the size of the frame F is 2 as it contains two exclusive class labels—one for text data and another for non-text data. After individual probability calculation, the probabilistic outputs of two different classifiers are converted to a more complex mass function. Initially, the elements of F are sorted in decreasing order of probabilities as defined in (9):

$$p(c_1) > ... > p(c_{|F|}). \tag{9}$$

Here, $p(c_i)$ is the probability value corresponding to a particular class in the frame F.

Next, mass function μ is described using (10) and (11):

$$\mu(\{c_1, c_2, ..., c_{|F|}\}) = \mu(F) = |F| p(c_{|F|}), \tag{10}$$

$$\forall i < |F|, \mu(\{c_1, c_2, ..., c_i\}) = i[p(c_i) - p(c_{i+1})]. \tag{11}$$

As in the present work $|F| = 2$, $\mu_1(c_i)$ and $\mu_1(c_i, c_{i+1})$ have been obtained from the resultant probability set of SVM classifier. Here, each subset of μ_1 is represented by X. Similarly, $\mu_2(c_i)$ and $\mu_2(c_i, c_{i+1})$ have been obtained from the resultant probability set of HMM classifier. Here, each subset of μ_2 is represented by Y. The mass functions μ_1 and μ_2 obtained from two independent sources are combined into a consonant mass function using (12):

$$M(A) = \frac{\sum_{X \cap Y = A} \mu_1(X)\mu_2(Y)}{1 - \sum_{X \cap Y = \phi} \mu_1(X)\mu_2(Y)} \tag{12}$$

where A $\neq \phi$ and A $\subseteq F$. For decision making, *belief*, *plausibility* and *conflict* have been computed for each class C using (13) and (14):

$$belief(C) = \sum_{A \subseteq C} M(A), \tag{13}$$

$$plausibility(C) = \sum_{A \cap C \neq \phi} M(A). \tag{14}$$

The values of *belief* and *plausibility* have been used to calculate the degree of *conflict*. Thus, the *conflict* of a class C is calculated using (15):

$$conflict(C) = plausibility(C) - belief(C). \tag{15}$$

At this point, the class C with the lowest *conflict* value has been selected as the final class of the text/non-text sample.

6. Experimental results and analysis

The performance of the present system has been measured using the test dataset mentioned in section 3. The present work classifies text and non-text data using SVM and HMM classifiers separately as well as by combining the probabilistic outputs of these two classifiers using DST. Hence, this section reports the classification results using both SVM and HMM classifiers separately as well as the combination of probabilistic scores of these two classifiers by DST.

6.1 *Classification results using SVM*

While evaluating the performance of the present system in SVM-based platform, experiments have been carried out using different kernels of SVM–polynomial, Linear and (Gaussian) Radial Basis Function (RBF). It has been noted that RBF kernel provides the best classification performance. Table 2 presents the text/non-text classification accuracies obtained using different kernels of SVM. This table shows that better classification accuracy is obtained using 4-fold evaluation method in comparison with holdout method. The optimal set of values of various hyper-parameters in SVM is shown in table 3. This optimal set has been generated using Bayesian optimization technique.

Table 2. Text/non-text classification results using different kernels of SVM.

	Accuracy (%)		
	Holdout method		4-fold evaluation method
SVM kernel	Train data	Test data	
Linear	91.23	90.54	90.67
Polynomial	91.61	90.89	91.12
(Gaussian) RBF	92.11	91.28	91.43

Table 3. Optimal values of hyper-parameters in SVM.

Soft-margin constant C	Width of (Gaussian) RBF kernel γ	Degree of polynomial kernel
8	2.31	3

6.2 *Classification results using HMM*

Experiments have been carried out with varying number of HMM states (2–6), where the number of Gaussian mixture (GM) components has been varied in each state. Numbers of GM have been considered from 1 to 128 with a step of power of 2. It has been found that with 128 GM and 5 states, HMM provides the best performance. The detailed results of the text/non-text classification are shown in table 4. Like SVM, the results obtained from HMM also show that better classification accuracy is obtained using 4-fold evaluation method in comparison with holdout method. The present system has also been tested by constructing several concentric ellipses around the data. Sub-region-wise classification accuracies using 4-fold evaluation method with 128 GM and 5 states of HMM are shown in table 5. It is seen that construction of 3 concentric ellipses provides the best classification accuracy among all numbers of concentric ellipses. The results presented in tables 2 and 4 also reveal that better classification accuracies are obtained using HMM in comparison with SVM.

Table 4. Text/non-text classification results using various combinations of HMM states and Gaussian mixtures.

		Accuracy (%)	
HMM states	GM	Holdout method	4-fold evaluation method
3	32	86.89	87.07
3	64	87.54	87.68
3	128	88.12	88.33
4	32	89.53	89.71
4	64	90.18	90.33
4	128	90.87	91.06
5	32	91.23	91.34
5	64	91.67	91.81
5	128	92.12	92.33

Table 5. Sub-region-wise text/non-text classification accuracy using 4-fold evaluation method with 128 GM and 5 states of HMM.

Ellipses	Sub-regions	Accuracy (%)
2	16	86.39
3	24	92.33
4	32	91.43
5	40	90.93

6.3 *Classification results using classifier combination*

This subsection reports the text/non-text classification results obtained by combining the probabilistic outcomes of SVM and HMM classifiers using DST. Here, the probabilistic outcomes obtained using 4-fold evaluation method only have been considered for combining as this method has provided better classification accuracies using both SVM and HMM separately. In the 4-fold evaluation method, the probabilistic outcomes obtained using RBF kernel of SVM and 128 GM–5 states combination of HMM have been considered for combining. Feature vectors of dimension 768, obtained by constructing 3 concentric ellipses, have been used here to get the individual probabilistic outcomes from SVM and HMM. The rightmost bar in figure 8 shows the text/non-text classification accuracy after combining the probabilistic outcomes of the aforesaid classifiers using DST. This figure shows that classifier fusion strategy using DST produces better classification accuracy in comparison with use of SVM and HMM separately. The accuracy obtained using DST-based approach is also compared to accuracies of a few existing classifier fusion strategies—Product rule, Borda Count rule and Sum rule. The comparative analysis of these accuracies is shown in figure 8. It can be seen from figure 8 that DST-based results outstrip other fusion strategies.

The performance of the proposed system is also measured using *Precision*, *Recall*, *F1-Score* and *Receiver operating characteristic (ROC)* analysis as performance measurement parameters. *Precision*, *Recall* and *F1-Score* are defined using (16)–(18). *ROC* analysis curve is generated by plotting the True Positive Rate (*TPR*) against the False Positive Rate (*FPR*).

$$Precision = \frac{True\ Positive\ (TP)}{True\ Positive + False\ Positive\ (FP)}, \quad (16)$$

$$Recall = \frac{True\ Positive}{True\ Positive + False\ Negative (FN)}, \quad (17)$$

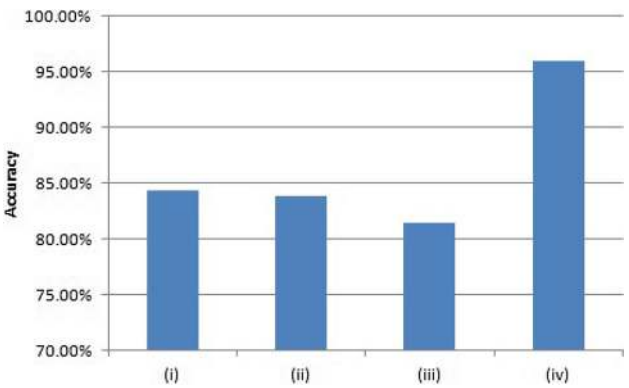

Figure 8. Comparative text/non-text classification performance analysis of various classifier fusion methods: (i) Product rule, (ii) Borda Count rule, (iii) Sum rule and (iv) Proposed method using DST.

Table 6. *Precision*, *Recall* and *F1-Score* of the proposed system.

Parameter	Value (%)	Statistics
Precision	95.99	TP(text) = 4902, FP(text) = 311, TP(non-text) = 4689, FP(non-text) = 98
Recall	95.91	TP(text) = 4902, FN(text) = 98, TP(non-text) = 4689, FN(non-text) = 311
F1-Score	95.94	Precision = 95.99%, Recall = 95.91%

$$F1 - Score = \frac{2 \times Precision \times Recall}{Precision + Recall}. \quad (18)$$

Table 6 shows the *Precision*, *Recall* and *F1-Score* of the proposed system and the related statistics. *ROC* analysis is shown in figure 9.

6.4 *Error analysis*

While analysing errors in the present system performance, it has been noted that most of the classification errors occurred between like-shaped pair of samples of text and non-text data. A few such similar-shaped pairs of samples of text and non-text classes between which misclassification has been reported are shown in figure 10.

6.5 *Comparative performance analysis*

As we could not find any research work on online handwritten text and non-text document classification in any Indic script, including Devanagari, performance of the present work cannot be compared to any existing study in Indic scripts. However, a comparative performance analysis is presented in table 7 with a few studies available in non-Indic scripts. As datasets used in these existing studies are not publicly available, the accuracies of these existing works have been computed on our own dataset used in the present work after implementing the respective algorithms used in these existing works, to have the comparative performance analysis on the same platform.

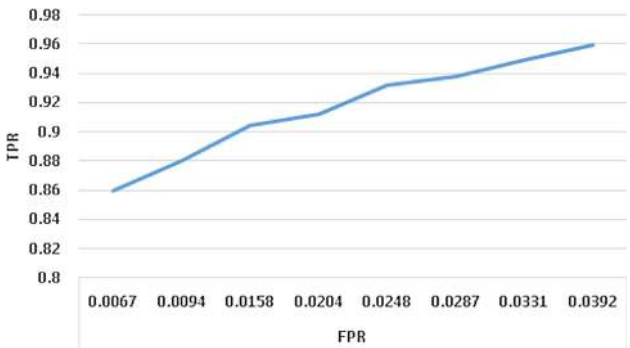

Figure 9. *ROC* analysis of the proposed system.

Text	Character/digit	Non-text	Object
	Character		Circle
	Digit		Circle
	Character		Graph

Figure 10. A few similar-shaped pairs of samples of text and non-text data between which misclassification has been reported.

Table 7. Comparative performance analysis with a few existing studies.

Reference	Dataset	Accuracy (%)
Delaye and Lee [1]	Present work	90.54
VanPhan and Nakagawa [3]	Present work	88.67
Zhou and Liu [4]	Present work	88.72
Proposed method	Present work	95.91

7. Conclusion

Text/non-text classification in any online handwritten document is an interesting field of research from both scientific and commercial points of view. Extracting features to classify handwritten text/non-text is a challenging task as several handwritten textual strokes may have some feature values common to those of non-textual strokes as well as existence of a few like-shaped pairs of samples between text and non-text data. In this work, analysing features of elliptical regions and combining classifiers have produced encouraging result for online handwritten text and non-text classification in the most popular Indic script—Devanagari. However, still there are opportunities to improve the performance of the system; especially the misclassification rate can be reduced between similar-shaped pairs of samples of these two classes and work will be continued for the same. This research work can further be extended for online recognition of different textual and non-textual portions of an online handwritten document in Devanagari script as well as text/non-text classification in other Indic scripts.

References

[1] Delaye A and Lee K 2015 A flexible framework for online document segmentation by pairwise stroke distance learning. *Pattern Recognit.* 48: 1197–1210

[2] Delaye A and Liu C L 2014 Contextual text/non-text stroke classification in online handwritten notes with conditional random fields. *Pattern Recognit.* 47(3): 959–968

[3] Van Phan T and Nakagawa M 2014 Text/non-text classification in online handwritten documents with recurrent neural networks. In: *Proceedings of the 14th International Conference on Frontiers in Handwriting Recognition*, Heraklion, Greece. IEEE Press, pp. 23–28

[4] Zhou X D and Liu C L 2007 Text/non-text ink stroke classification in Japanese handwriting based on Markov random fields. In: *Proceedings of the 9th International Conference on Document Analysis and Recognition*, Parana, Brazil. IEEE Press, pp. 377–381

[5] Zhou X D, Wang D H and Liu C 2009 A robust approach to text line grouping in online handwritten Japanese documents. *Pattern Recognit.* 42(9): 2077–2088

[6] Liwicki M, Indermuhle E and Bunke H 2007 Online hand written text line detection using dynamic programming. In: *Proceedings of the 9th International Conference on Document Analysis and Recognition*, Parana, Brazil. IEEE Press, pp. 447–451

[7] Ye M, Sutanto H, Raghupathy S, Li C and Shilman M 2005 Grouping text lines in free form handwritten notes. In: *Proceedings of the 8th International Conference on Document Analysis and Recognition*, Seoul, South Korea. IEEE Press, pp. 367–371

[8] Blanchard J and Artieres T 2004 On-line handwritten documents segmentation. In: *Proceedings of the 9th International Workshop on Frontiers in Handwriting Recognition*, Tokyo, Japan. IEEE Press, pp. 148–153

[9] Ghosh R, Vamsi C and Kumar P 2019 RNN based online handwritten word recognition in Devanagari and Bengali scripts using horizontal zoning. *Pattern Recognit.* 92: 203–218

[10] Mathew M, Jain M and Jawahar C V 2017 Benchmarking scene text recognition in Devanagari, Telugu and Malayalam. In: *Proceedings of the 14th International Conference on Document Analysis and Recognition*, Kyoto, Japan. IEEE Press, pp. 42–46

[11] Feng G, Viard-Gaudin C and Sun Z 2009 Online hand-drawn electric circuit diagram recognition using 2D dynamic programming. *Pattern Recognit.* 42(12): 3215–3223

[12] Bresler M, VanPhan T, Prusa D, Nakagawa M and Hlavc V 2014 Recognition system for online sketched diagrams. In: *Proceedings of the 14th International Conference on Frontiers in Handwriting Recognition*, Heraklion, Greece. IEEE Press, pp. 563–568

[13] Bharath A and Madhvanath S 2012 HMM-based lexicon-driven and lexicon-free word recognition for online handwritten Indic scripts. *IEEE Trans. Pattern Anal. Mach. Intell.* 34(4): 670–682

[14] Ghosh R, Roy P P and Kumar P 2018 Smart device authentication based on online handwritten script identification and word recognition in Indic scripts using zone-wise features. *Int. J. Inf. Syst. Model. Des.* 9(1): 21–55

[15] Burges C J C 1998 A tutorial on support vector machines for pattern recognition. *Data Min. Knowl. Discov.* 2(2): 121–167

[16] Pal U, Roy P P, Tripathy N and Llados J 2010 Multi-oriented Bangla and Devanagari text recognition. *Pattern Recognit.* 43: 4124–4136

[17] Vapnik V 1995 *The Nature of Statistical Learning Theory*, 2nd ed. New York: Springer, pp. 1–314

[18] Rabiner L R 1989 A tutorial on hidden Markov models and selected applications in speech recognition. *Proc. IEEE* 77(2): 257–286

[19] Ghosh R, Kumar P and Roy P P 2018 A Dempster–Shafer theory based classifier combination for online signature recognition and verification systems. *Int. J. Mach. Learn. Cybern.* https://doi.org/10.1007/s13042-018-0883-9, pp. 1–16