Turkish Journal of Electrical Engineering & Computer Sciences

http://journals.tubitak.gov.tr/elektrik/

Research Article

Turk J Elec Eng & Comp Sci
(2019) 27: 3195 – 3207
© TÜBİTAK
doi:10.3906/elk-1901-48

A comparative study on handwritten Bangla character recognition

Md. Atiqul Islam RIZVI, Kaushik DEB*, Md. Ibrahim KHAN, Mir Md. Saki KOWSAR, Tahmina KHANAM
Department of Computer Science & Engineering, Chittagong University of Engineering & Technology, Chattogram, Bangladesh

Received: 06.01.2019 • **Accepted/Published Online:** 27.04.2019 • **Final Version:** 26.07.2019

Abstract: Recognition of handwritten Bangla characters has drawn considerable attention recently. The Bangla language is rich with characters of various styles such as numerals, basic characters, and compound and modifier characters. The inherent variation in individual writing styles, along with the complex, cursive nature of characters, makes the recognition task more challenging. To compare the outcomes of handwritten Bangla character recognition, this study considers two different approaches. The first one is classifier-based, where a hybrid model of the feature extraction technique extracts the features and a multiclass support vector machine (SVM) performs the recognition. The second one is based on a convolution neural network (CNN). For recognition, we considered 10 Bangla numerals, 50 basic characters, and a subset of compound characters that are frequently used in the Bangla language. Experimental results demonstrate that the CNN model outperforms the traditional classifier-based approach, obtaining 98.04%, 99.68%, and 98.18% recognition accuracy for Bangla basic characters, numerals, and the subset of compound characters, respectively.

Key words: Bangla handwritten character recognition, zone-based density feature, ICZ+ZCZ feature, histogram of oriented gradient feature, multiclass support vector machine, convolutional neural network, ResNet

1. Introduction

In the field of pattern recognition, optical character recognition (OCR) is a very popular and active research field. OCR is a process to convert printed text or images to digital form. There are two ways of character recognition. Online OCR deals with the recognition of the incoming stream of characters. Offline OCR is the process of extracting text from scanned images of printed or handwritten documents.

Bangla is the official and the most widely spoken language of Bangladesh. Approximately 250–300 million people utilize Bangla in various aspects. Rank-wise, it is the seventh most spoken native language in the world. There are two forms to represent the characters of the Bangla language, such as printed characters and handwritten characters. Despite the wide use and popularity of Bangla, there exist very few research works for complete Bangla handwritten character recognition compared to other popular languages such as English.

An official Bangla OCR for printed documents, named 'Puthi' was launched in 2014. Sadly, in the case of handwritten characters, we cannot utilize the methods used for recognizing printed characters. The primary reason is the variation and unpredictability of the handwriting style. Bangla characters recognized from our framework may be used for a wide range of applications, such as assistive technology for blind and visually impaired users, automated system for banks and postal office, or extraction of information from different types of boxed application forms.

*Correspondence: debkaushik99@cuet.ac.bd

Characters of the Bangla language use a Sanskrit-based style. It is completely different from English or similar language styles. The Bangla language consists of 10 numerals, 50 basic characters, and more than 260 compound characters. Properties such as shape, size, and orientation remain uniform for a given font size in printed form. These properties do not apply to the handwritten character recognition problem. Hence, achieving higher recognition accuracy is more difficult than ever.

To compare the recognition results of handwritten Bangla characters, this study proposes two different approaches. First, a feature-classifier framework is considered, which uses a hybrid feature extraction model and a multiclass support vector machine (SVM). Second, ResNet, a convolution neural network (CNN) model, is considered. The performances of the proposed models are compared and several previous works are also considered.

2. Related research works

One of the most challenging factors in recognition of handwritten Bangla characters is the large number of complexly shaped character classes. To understand the handwritten Bangla character recognition framework, we should have a clear understanding of the previous research works. There exist very few research works on overall recognition of the handwritten form of Bangla characters. Here, we will try to cover various techniques considered by different researchers. Some studies [1–3] represent the earlier stages of handwritten Bangla character recognition. They used several feature extraction techniques. Those works widely used multilayer perceptron (MLP) networks and achieved notable accuracy in the field.

Later, Bhowmik et al. proposed a hierarchical classification architecture based on the SVM classifier in [4]. In this work, the authors emphasized the classification step rather than feature extraction. They used wavelet transform to extract the features. For classification, the authors used three different classifiers in combination and compared the results with each other. The hierarchical architecture used three different grouping schemes. They are the disjoint grouping scheme, overlapping grouping scheme, and grouping scheme with neural gas. Fusion of three different classifiers, the MLP network, the radial basis function (RBF) network, and SVM based on SVM, performs better in this work. In [5], Das et al. proposed a feature extraction technique based on two different methods. In this research, the author not only worked on basic characters of the Bangla language but also on the compound characters. For extraction of features, the used techniques are the longest run feature and quad-tree feature based on shadow technique. The authors compared the classification results by considering both the MLP and SVM classifiers, using 3-fold cross-validation. After the evaluation of results, the authors concluded that the SVM classifier performed better than the MLP classifier.

There are 10 characters in Bangla numerals/digits. For the recognition of handwritten Bangla numerals, Bhattacharya et al. proposed a recognition architecture using neural network models in [6]. Here, the authors used a topology-adaptive self-organizing neural network (TASONN) to extract the geometrical shapes of numerals. They used features extracted from those shapes to classify samples into groups. Another MLP classifier performs further classification whenever a group consists of more than one numeral. In [7], Basu et al. proposed a combination of MLP classifiers for the recognition of Bangla numerals. The first feature set includes shadow features and centroid features. The second one uses longest run features, computed from the image. The authors used both of these features to train two different MLP classifiers. For classification, they combined decisions of these two classifiers using the Dempster–Shafer (DS) technique.

Among the characters of the Bangla language, the compound characters exhibit the most complex nature of shape and strokes. The combination of more than one character makes the recognition of compound characters more difficult. Pal et al. proposed a method using the modified quadratic discriminant feature (MQDF) classifier in [8]. Using filtering techniques on the image, they obtained the gradient of the image. Then they derived the feature sets by computing the arc tangent of those gradients. The authors used the five-fold cross-validation technique of MQDF for classification. In [9], Bag et al. proposed a method based on topological features of the characters. Various writing strokes of the handwritten characters represent geometrical convex shapes. The authors extracted features from the previously segmented convex shapes. They used the template matching scheme to assign each of the characters.

In recent times, deep learning structures have started to attract many researchers. They eliminate the need for static feature extraction and perform far better than traditional machine learning methods. Sazal et al. [10] considered a deep belief network (DBN) in their study, which includes three layers of restricted Boltzmann machine. For numeral recognition, Sharif et al. proposed a hybrid model of deep CNN in [11]. An artificial neural network with histogram of oriented gradient (HOG) is the first component of their network. They used a traditional CNN with two convolutional layers as their second component. These two component are connected to a fully connected layer. Much simpler use of CNN is observed in the work of Purkaystha et al. in [12]. The CNN model contains two convolutional layers, which are connected to three dense layers. They applied their method on a dataset, including numerals and basic and compound characters. Rabby et al. proposed a model of multilayered CNN, comprising a total of 22 layers in [13]. Their network architecture includes several convolutional layers, max pooling layers, and fully connected layers.

3. Methodology

In this study, we focus on two different architectures for the recognition of handwritten Bangla characters. First, a hybrid model of a feature descriptor is considered. A multiclass SVM classifier uses these features to train the model. Using this trained classifier, we try to recognize unknown testing samples. Second, we consider ResNet, a multilayer CNN that uses residual learning. As the input image propagates through the network, it automatically learns the necessary features to perform recognition.

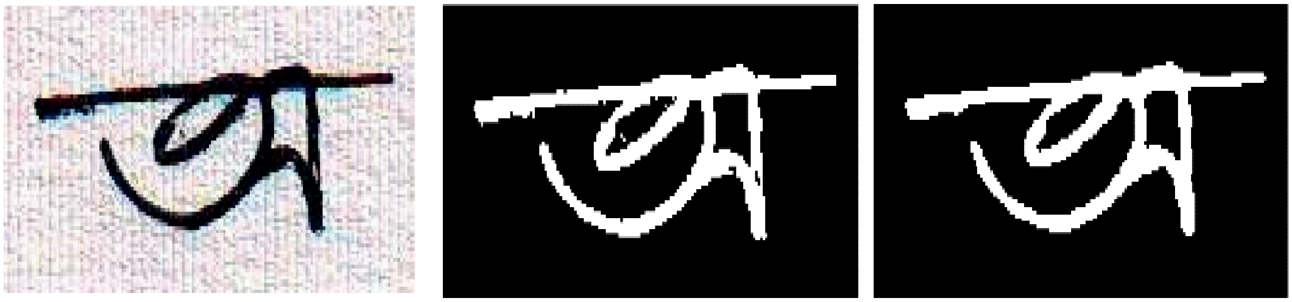
3.1. SVM-based architecture

3.1.1. Preprocessing

In the preprocessing part of this hybrid feature-classifier model, binarization is the first step to be executed. Since the computational cost of working with RGB and gray-level images is higher, it is preferred to convert the input image into a binary image. Different binarization algorithms [14] exist for this process, such as Sauvola's method, Bernsen's method, etc. Here, we applied Otsu's method [15] for the binarization process. To remove existing noises from the input samples, a median filter was used. While removing noise, it is also important to preserve the edges of the characters. Median filters work better than other filters in this regard. The size of the extracted feature vector is proportional to the size of the image. For achieving the trade-off between computational cost and image information, we scaled all the scanned images to a 32×32 matrix for the handwritten characters. Figure 1a shows the scanned image of a sample character as input to the preprocessing part. Figure 1b shows the resultant image after binarization. Finally, Figure 1c shows the filtered image after implementing the median filter.

3.1.2. Feature extraction

A feature is information that is relevant to a type of specific property of an object. The major goal of any feature extraction technique is to extract a minimum number of features that will maximize the classification



(a) (b) (c)

Figure 1. Input and output of preprocessing steps: (a) scanned image of a character, (b) binarized image, (c) filtered image.

of the object while keeping the cost minimal. In [16], Trier et al. gave an overview of various methods of extracting features for offline character recognition. Among many others, we considered zone-based density (ZBD), combination of an image and zone centroid zone (ICZ+ZCZ), and histogram of oriented gradient (HOG) feature in this hybrid model.

Zoning is one of the most popular feature extraction methods. In this method, we partitioned the input image into zones of predefined height and width. Then we sequentially calculated the density characteristics for each of these zones. Here, we considered the main concept of the zoning method, combined with the density features [17]. The size of the feature vector is 26. The following pseudocode depicts the steps of this method.

Zone-based density (ZBD) feature

Step 1: Divide the input image into 16 zones. The size of each zone is 8×8 pixels.
Step 2: Compute the density features from each of the 16 zones, Di, where i = 1, 2, ...
Step 3: Accumulate the density features of each zone into four basic directions such as Up, Down, Left, Right. Then calculate the difference between these four directions. Here, Up = Zones 1 to 8, Down = Zones 9 to 16
Left = Zones 1, 2, 5, 6, 9, 10, 13, 14
Right = Zones 3, 4, 7, 8, 11, 12, 15, 16
f1 = Up − Down & f2 = Left − Right
Step 4: Also calculate the average density of consecutive zones.
Step 5: Concatenate the features obtained in Steps 2, 3, and 4 to form the ZBD feature vector.

The image centroid zone (ICZ) and zone centroid zone (ZCZ) are two interrelated feature extraction methods. We obtained 2 features from each zone, resulting in a feature vector of size 128. We have chosen a hybrid approach [18] as a feature extraction technique, which is the combination of these two methods. The following pseudocode depicts the steps of this method.

ICZ+ZCZ feature

Step 1: Divide the input image into 64 zones. The size of each zone is 4×4 pixels.
Step 2: Calculate the centroid of the input image. Compute the centroid for each zone, i = 1, 2, ...
Step 3: Calculate the Euclidean distance from the image centroid and zone centroid to foreground pixels present in each of the zones.

Step 4: Averages of these distances are the features for each zone.

HOG is a widely used method in object detection. Dalal and Triggs first introduced the HOG features in [19]. In this method, we partition the image into overlapping cells and blocks. Then we calculate histograms of gradient direction for each cell [20].

The first step of this method is the gradient calculation. For gradient calculation, we consider the Sobel mask, which is a 2-D mask. We convolve this 2-D mask with the whole image to determine $f_x(x, y)$ and $f_y(x, y)$, where f(x, y) is the intensity value of the pixel (x, y). Magnitude and orientation of these gradients can be calculated by the following equations:

$$Magnitude = \sqrt{f_x(x,y)^2 + f_y(x,y)^2}, \tag{1}$$

$$Orientation = \arctan(\frac{f_y(x,y)}{f_x(x,y)}). \tag{2}$$

The second step is the calculation of cell histograms. Each pixel within a cell casts a weighted vote for an orientation-based histogram. The total orientation is evenly spread into 9 bins, from $0°$ to $180°$, where the size of each bin is $20°$. In order to reduce the aliasing effect, we interpolate the weighted votes between two neighboring bins. For each pixel, the bin contributions are split between the bin to the left and the bin to the right, based on how far the angle is from the bin centers. Figure 2a shows the representation of the calculated histogram for a sample cell.

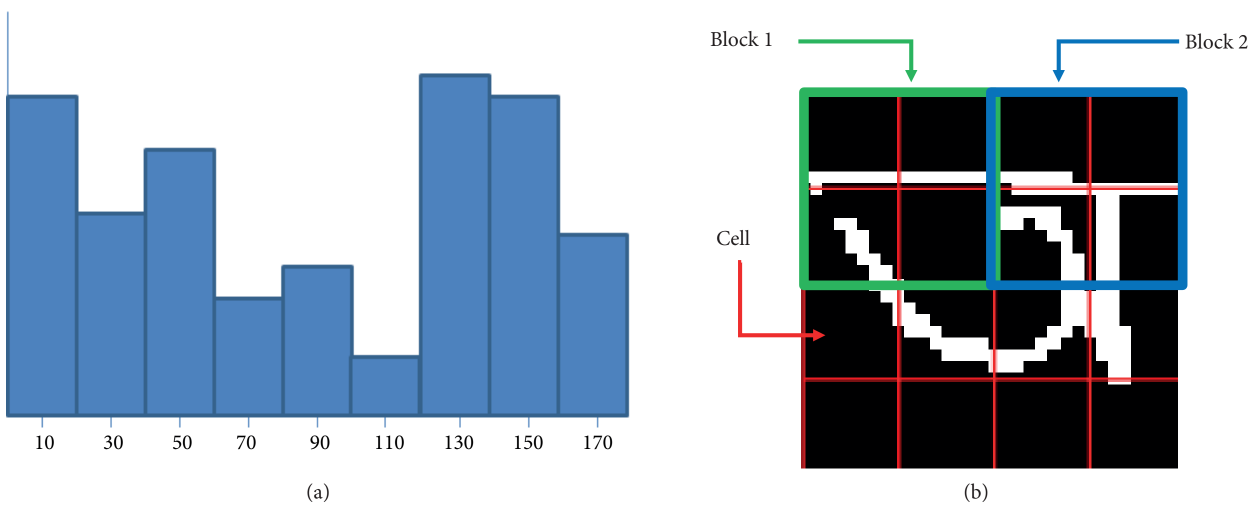


Figure 2. Visualization of cells, blocks, and histograms of HOG: (a) histogram for a cell, (b) cells and overlapping blocks.

To normalize the gradient magnitudes, the grouping of cells is necessary. This grouping, called blocks, holds the histograms of all normalized cells. Typically block overlapping occurs, which means each cell contributes to more than one block. Figure 2b shows the cells and blocks, along with the overlapping of blocks. Here we define cell size at 8×8 pixels and block size at 2×2 cells. We consider 50% of overlapping of cells to ensure contrast normalization between adjacent blocks.

The third step is normalization of histogram blocks. The range of calculated gradients can be very large, resulting in unevenness in the feature descriptor. Hence, normalization of the feature vector is necessary. Normalization is a process that alters the range of intensity values. There are many types of normalization methods. In this HOG feature extraction method, we apply the L2-norm, as stated in Eq. (3):

$$\nu = \frac{\nu}{\sqrt{||\nu||^2 + \varepsilon^2}}, \tag{3}$$

where ε is a constant, generally taking value 1. ν is the normalized feature vector.

Thus, the total number of blocks generated for any sample image is $3 \times 3 = 9$. The total size of the generated HOG feature vector is $9 \times 4 \times 9 = 324$. Among the three feature extraction methods, only HOG is scale-invariant. The scale-invariant property of a feature descriptor denotes that the size of the feature remains unchanged, even if the dimension of the image changes.

3.1.3. Classification

Classification is the process of assigning an unknown sample to a predefined class or type based on the training data. Among many other classifiers, the support vector machine (SVM) classifier has gained popularity in recognition tasks recently. SVM is a supervised machine learning algorithm. This method plots each data item as a point in n-dimensional space. ($\mathrm{x_i}$, $\mathrm{y_i}$) represents the data item, where $\mathrm{x_i}$ is the feature vector of sample i and $\mathrm{y_i}$ is the associated label (usually given as +1 or -1). Then SVM performs classification by finding the hyperplane that separates the classes. SVM tries to maximize the cost function $\frac{1}{2}(W^T W)$ to find the hyperplane. Constraints for this problem are stated in Eq. (4):

$$y_i = \begin{cases} +1, & \text{if } w \times x_i + b \geq 1, \\ -1, & \text{otherwise,} \end{cases} \tag{4}$$

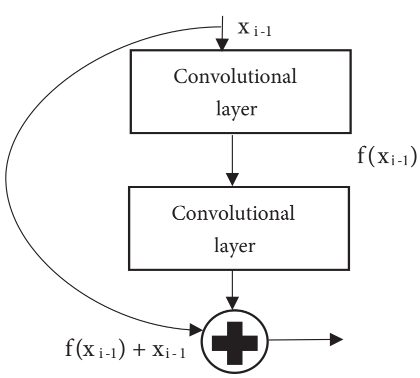
where w is the weight factor and b is the bias. As we are dealing with about 75 classes of handwritten characters, it is obviously a multiclass problem. The main concept of the binary class problem can be extended to solve the multiclass problem. In [4], the authors constructed the multiclass SVM by combining several binary class SVMs. There are two different coding design approaches while constructing this: the one vs. one and one vs. all methods.

In the one vs. all method, c binary classifiers are generated for the c-class problem. The ith binary classifier creates a boundary between class i and all others. The winner-takes-all (WTA) method is applied, where an unknown sample is assigned to a class having the largest decision value, even if that value is negative. In the one vs. one method, $\frac{c(c-1)}{2}$ binary classifiers are generated. Binary $\mathrm{classifier_{i,\,j}}$ is trained with the positive sample of class i and the negative sample of class j. The max-wins voting (MWV) method is applied, where voting is increased iteratively among $\frac{c(c-1)}{2}$ classifiers whenever a positive decision value is encountered. Then unknown sample x is assigned to the class having the largest vote. Here, we used a multiclass SVM, which utilizes the one vs. one encoding method because it provides better accuracy than the one vs. all method. However, the initial generation of the $\frac{c(c-1)}{2}$ classifier increases the overhead more than one vs. all.

3.2. CNN-based architecture

Deep CNN (DCNN) is generally a class of deep neural networks (DNN). DNNs have created massive hype in recent times, outperforming several traditional machine learning algorithms. However, as the network goes deeper, a problem arises. As cited by many researchers, it was found that accuracy becomes saturated. Even test error rises when training a deeper network composed of many layers. Also, DNNs are susceptible to the vanishing gradient problem, in which a very small gradient prevents the updating of weights of the layers. This problem exponentially slows down the training of the deeper layers of the network. By using residual learning [21], we can avoid this problem.

In residual learning, instead of features, the residuals of the features train the network. ResNet is a state-of-the-art DCNN model, built upon the concept of residual learning. ResNet uses a short-cut connection in its architecture, connecting the input of the nth layer to some $n+i$ layer. The ResNet architecture consists of several residual building blocks. Let the input to the residual block be x_{i-1} and output from the block be x_i. Also, we assume, after applying different operations such as convolution, batch normalization, and ReLU activation on input, that the output is $f(x_{i-1})$. Considering residual learning, we can define $f(x_{i-1}) = x_i - x_{i-1}$. Thus, we obtain $x_i = f(x_{i-1}) + x_{i-1}$. Using this method enables the information of the previous layer to be propagated in the following layer. A basic building block of ResNet is shown in Figure 3a. There are several variations of ResNet. In this work, we evaluate the recognition accuracy using ResNet-18, which contains 17 convolutional layers and one fully connected (FC) layer. The layers are stacked upon one another, using the concept of residual learning. Figure 3b shows the structure and size of different filters used in the ResNet-18 architecture. Each convolutional layer in the residual block is followed by its associated batch normalization layer and ReLU layer. At the end of the first residual block, a max pooling layer is used. After the fifth residual block, an average pooling layer is used, which is connected to the FC layer. The output size of the FC layer is equal to the number of the class.



(a)

Residual block	Filter size	Number of filters	Total convolutional layer
1	7×7	64	1
2	3×3	64	4
3	3×3	128	4
4	3×3	256	4
5	3×3	512	4

(b)

Figure 3. ResNet-18 architecture: (a) basic building block of residual learning, (b) filter information of different convolutional layer.

4. Experimental results

4.1. Dataset description

In this work, we use two different datasets, BanglaLekha-Isolated [22] and Ekush [13]. Both datasets are a collection of isolated handwritten characters of the Bangla language. We formed our dataset by taking 700 samples from BanglaLekha-Isolated and 3000 samples from Ekush per unique character. For the convenience of computation, we divided the dataset into three portions such as basic characters, which include the vowels and consonants, and numerals and compound characters. Some samples of basic characters, numerals, and compound characters from the dataset of handwritten Bangla characters are shown in Figure 4a, Figure 4b, and Figure 4c, respectively.

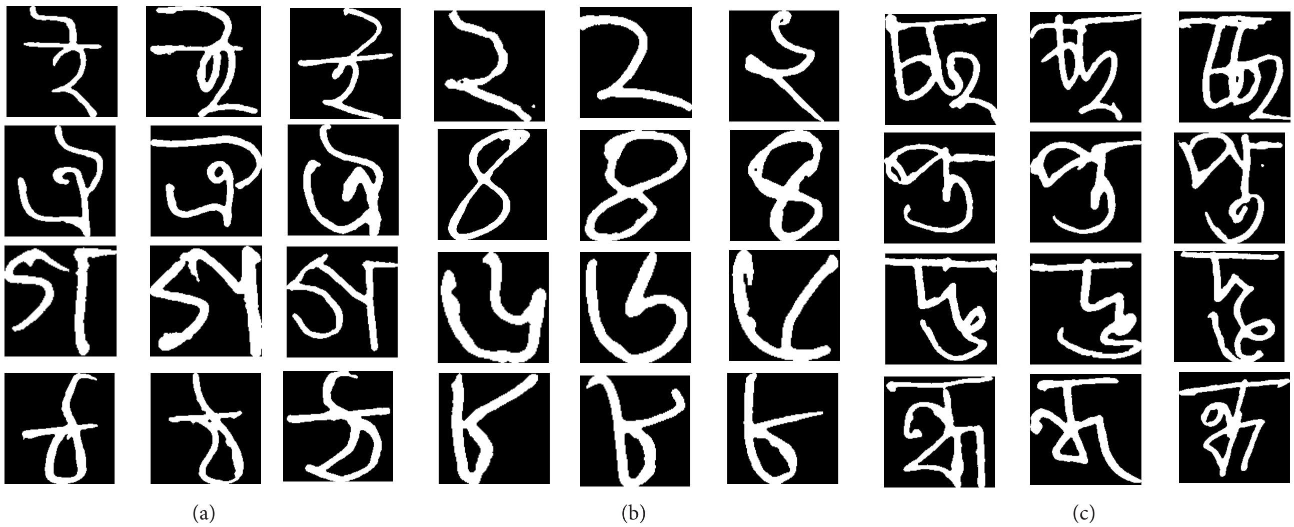


Figure 4. Samples of handwritten Bangla (a) basic characters, (b) numerals, and (c) compound characters.

To evaluate the performance of the SVM-based architecture, we have used the k-fold cross-validation technique. We have considered 5 as the value of parameter k. Here, we randomly divide the dataset into 5 subsets of equal size. Among the 5 subsets, we have used k-1 subsets as training data and the remaining subset as the validation data to test the model. The classification process is then repeated 5 times, with each of the 5 subsets as the testing data, one at a time. Then we average the results obtained from 5 iterations to produce a single estimation. The advantage of using this method is that we can use all of the samples for both training and testing, and each sample comes into testing eventually. In the CNN-based architecture, we randomly shuffle the entire dataset after every epoch. We use 70% of the dataset for training, 15% for validation, and the remaining 15% for testing.

4.2. Evaluation metrics

Performance evaluation of binary class problems is quite easy, but for the multiclass problem, it is quite the opposite. For the performance measure of a multiclass classification task, Sokolova proposed some formulas in [23]. From the confusion matrix, we derived a set of value of true positives (t_p), true negatives (t_n), false positives (f_p), and false negatives (f_n), and c is the number of classes in this multiclass problem. The F1-score is the harmonic mean of precision and recall. The F1-score depicts the overall system performance. In a multiclass problem, there are two types of measures, namely micro and macro. We use the formulas below to determine the performance of both approaches.

$$Precision_\mu = \frac{\sum_{i=1}^{c} t_p}{\sum_{i=1}^{c} t_p + f_p}. \tag{5}$$

$$Precision_M = \frac{\sum_{i=1}^{c} \frac{t_p}{t_p+f_p}}{c}. \tag{6}$$

$$Recall_{\mu} = \frac{\sum_{i=1}^{c} t_p}{\sum_{i=1}^{c} t_p + f_n}. \tag{7}$$

$$Recall_{M} = \frac{\sum_{i=1}^{c} \frac{t_p}{t_p + f_n}}{c}. \tag{8}$$

$$F1 - score = 2 \times \frac{Precision \times Recall}{Precision + Recall}. \tag{9}$$

4.3. Results

There are two types of characters in the Bangla basic character set: vowels and consonants. Bangla vowels include 11 characters and consonants include 39 characters. Bangla numerals include 10 digits from 0 to 9. There are more than 260 compound character classes in the Bangla language, although frequently we do not use all of them. Due to the insufficiency of a standard dataset on compound characters, we consider a subset of 15 compound character classes in this work. From the confusion matrix, we observe that sometimes misclassification occurs among similar structure characters.

Table 1 depicts the comparison of performances for three different character types. For further comparison, we also applied a decision tree classifier to our dataset, but its recognition accuracy is even worse than that of the two models suggested in our work. If we consider the SVM-based model with hybrid feature extraction, we can see that it performs rather better than some previous classifier-based works, but the recognition accuracy

Table 1. Comparison of performance.

Character type	Related work	Evaluation model used	Result
Basic characters	Bhowmik et al. [4]	SVM-based classifier (no cross-validation)	86.07%
	Das et al. [5]	SVM classifier (3-fold cross-validation)	80.51%
	-	**SVM classifier (5-fold cross-validation)**	**87%**
	Sazal et al. [10]	DBN	90.27%
	Purkaystha et al. [12]	DCNN (2 convolutional layers)	91.23%
	Rabby et al. [13]	DCNN (8 convolutional layers)	97.73%
	-	**ResNet-18**	**98.04%**
	-	Decision tree-based classifier	83.64%
Numerals	Bhattacharya et al. [7]	MLP classifier (no cross-validation)	90.56%
	Basu et al. [8]	MLP classifier (3-fold cross-validation)	95.1%
	-	**SVM classifier (5-fold cross-validation)**	**96.6%**
	Sazal et al. [10]	DBN	90.27%
	Sharif et al. [11]	Hybrid DCNN	99.17%
	Purkaystha et al. [12]	DCNN (2 convolutional layers)	98.66%
	Rabby et al. [13]	DCNN (8 convolutional layers)	97.73%
	-	**ResNet-18**	**99.68%**
	-	Decision tree-based classifier	95.60%
Compound characters	Pal et al. [9]	MQDF classifier (5-fold cross-validation)	85.90%
	Das et al. [5]	SVM classifier (3-fold cross-validation)	80.51%
	-	**SVM classifier (5-fold cross-validation)**	**86.14%**
	Purkaystha et al. [12]	DCNN (2 convolutional layers)	91.60%
	Rabby et al. [13]	DCNN (8 convolutional layers)	97.73%
	-	**ResNet-18**	**98.18%**
	-	Decision tree-based classifier	79.53%

of this approach fails to reach the level of DCNN-based models. From the comparison, we see that recent works depending on deep learning achieve far better accuracy than traditional classifier-based models. In this work, we considered ResNet for the CNN-based approach. ResNet has been used successfully in many object detection problems. Here, we implemented an 18-layer deep ResNet for the recognition of handwritten Bangla characters. From Table 1, it is evident that the ResNet architecture outperforms the SVM-based model, as well as several recent works based on deep learning models.

We also analyzed the computational complexity of both architectures in Table 2, along with the decision tree classifier. The complexity of the multiclass SVM-based method depends on parameters n_t (number of training samples), n_f (number of features), and n_s (number of support vectors). The complexity of the decision tree is less than that of SVM, which uses only n_t and n_f. Calculation of complexity for the DCNN method is quite difficult. The complexity of the DCNN is proportional to the complexity of the convolutional layers. It depends primarily on four parameters: nf (number of filters), is (input size), fs (filter size), and l (number of layers).

Table 2. Comparison of computational complexity.

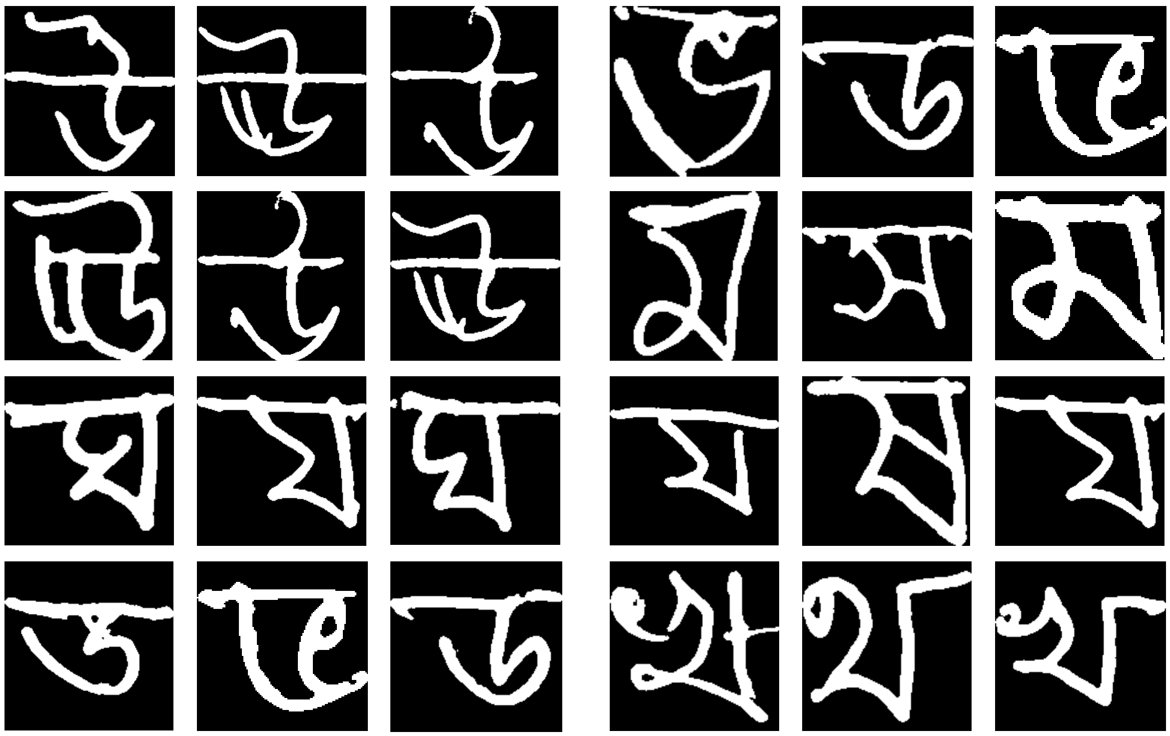
Method	DCNN	Decision tree	Proposed method
Complexity (C)	$C \propto \sum_{i=1}^{l} nf_{i\text{-}1} \times fs^2 \times nf_i \times is^2$	Training - $O({n_t}^2 n_f)$	Training - $O({n_t}^2 n_f + {n_t}^3)$
		Testing - $O(n_f)$	Testing - $O(n_s n_f)$

In Figures 5 and 6, we provide some samples of misclassified characters along with their original class. From the figures, we observe that some characters are almost identical to others in shape except for some small portion of the character. Sometimes misclassification occurs interchangeably among these character classes. In the case of compound characters, this similarity becomes more troublesome. As compound characters are the combination of more than one character, the shape of the primary character remains the same for different character classes. Hence, the possibility of misclassification arises.

5. Conclusion

In this study, we proposed two different approaches for the recognition of handwritten Bangla characters. Some obstacles are faced in recognizing handwritten Bangla characters. These are the variation in writing style and the complex and cursive nature of the Bangla characters varying from person to person and even from time to time. Recognition of handwritten Bangla characters is still under development for various factors. In the feature-classifier approach, we applied a hybrid model of three feature extraction methods. For classification, a multiclass SVM classifier is used that executes a one vs. one encoding method. For the CNN-based approach, we considered ResNet-18, which is a DCNN model. ResNet-18 uses residual learning as its core concept and the network includes 17 convolutional layers and one FC layer.

In this work, we formed a dataset of 277,500 samples in total. This dataset was formed by taking 700 samples from BanglaLekha-Isolated and 3000 samples from Ekush per unique character. Our proposed models are applied to 50 basic characters, 10 numerals, and a subset of 15 compound characters. For better understandability, we computed the performance in three distinct portions of the whole dataset. Comparing the results between the two approaches, it is evident that ResNet-18, the CNN-based model, outperforms the traditional classifier-based approach. Also, the comparison shows that this CNN-based model achieves better recognition accuracy than several recent works, all of which worked on DNN architectures. Among the three character types, the best accuracy has been achieved in numeral recognition.



(a) (b) (c) (a) (b) (c)

Figure 5. Misclassification of basic characters with (a) unknown sample, (b) predicted class, and (c) original class.

(a) (b) (c) (a) (b) (c)

(i) (ii)

Figure 6. Misclassification of (i) numerals and (ii) compound characters with (a) unknown sample, (b) predicted class, and (c) original class.

For future work, the recognition of modifiers may be considered in addition, as mostly modifier characters are transformed shapes of vowels and consonants. Moreover, the number of compound character classes may be increased to observe the performance of different character classes. Also, some other DCNN models may be considered for implementation. The layers of ResNet may be increased to achieve better accuracy.

References

[1] Rahman AFR, Rahman R, Fairhurst MC. Recognition of handwritten Bengali characters: a novel multistage approach. Pattern Recogn 2002; 35: 997-1006.

[2] Bhowmik TK, Bhattacharya U, Parui SK. Recognition of Bangla handwritten characters using an MLP classifier based on stroke features. In: 11th International Conference on Neural Information Processing ICONIP; 22–25 November 2004; Kolkata, India. pp. 814-819.

[3] Bhattacharya U, Shridhar M, Parui SK. On recognition of handwritten Bangla characters. In: 5th Indian Conference on Computer Vision, Graphics and Image Processing ICVGIP; 13–16 December 2006; Madurai, India. pp. 817-828.

[4] Bhowmik TK, Ghanty P, Roy A, Parui SK. SVM-based hierarchical architectures for handwritten Bangla character recognition. Int J Doc Anal Recog 2009; 12: 97-108.

[5] Das N, Das B, Sarkar R, Basu S, Kundu M, Nasipuri M. Handwritten Bangla basic and compound character recognition using MLP and SVM classifier. Journal of Computing 2010; 2: 109-115.

[6] Bhattacharya U, Das TK, Datta A, Parui SK, Chaudhuri BB. Recognition of handprinted Bangla numerals using neural network models. In: AFSS International Conference on Fuzzy Systems; 3–6 February 2002; Kolkata, India. pp. 228-235.

[7] Basu S, Sarkar R, Das N, Kundu M, Nasipuri M, Basu DK. Handwritten Bangla digit recognition using classifier combination through DS technique. In: 1st International Conference on Pattern Recognition and Machine Intelligence PReMI; 20–22 December 2005; Kolkata, India. pp. 236-241.

[8] Pal U, Wakabayashi T, Kimura F. Handwritten Bangla compound character recognition using gradient feature. In: 10th International Conference on Information Technology; 17–20 December 2007; Orissa, India. pp. 208-213.

[9] Bag S, Harit G, Bhowmick P. Topological features for recognizing printed and handwritten Bangla characters. In: Joint Workshop on Multilingual OCR and Analytics for Noisy Unstructured Text Data; 17 September 2011; Beijing, China.

[10] Sazal MMR, Biswas SK, Amin MF, Murase K. Bangla handwritten character recognition using deep belief network. In: International Conference on Electrical Information and Communication Technology EICT; 13–15 February 2014; Khulna, Bangladesh. pp. 1-5.

[11] Sharif SMA, Mohammed N, Mansoor N, Momen S. A hybrid deep model with HOG features for Bangla handwritten numeral classification. In: 9th International Conference on Electrical and Computer Engineering; 20–22 December 2016; Dhaka, Bangladesh. pp. 463-466.

[12] Purkaystha B, Datta T, Islam MS. Bengali handwritten character recognition using deep convolutional neural network. In: 20th International Conference of Computer and Information Technology; 22–24 December 2017; Dhaka, Bangladesh. pp. 1-5.

[13] Rabby AKMSA, Haque S, Abujar S, Hossain SA. EkushNet: Using convolutional neural network for Bangla handwritten recognition. Procedia Comput Sci 2018; 143: 603-610.

[14] Lokhande SS, Dawande NA. A survey on document image binarization techniques. In: Proceedings of International Conference on Computing Communication Control and Automation; 2015; Pune, India. pp. 742-746.

[15] Otsu N. A threshold selection method from gray-level histograms. IEEE T Syst Man Cyb 1979; 9: 62-66.

[16] Trier ØD, Jain AK, Taxt T. Feature extraction methods for character recognition-A survey. Pattern Recogn 1996; 29: 641-662.

[17] Bharambe M. Recognition of offline handwritten mathematical expressions. International Journal of Computer Applications 2015; 2015: 35-39

[18] Sinha G, Rani R, Dhir R. Handwritten Gurmukhi numeral recognition using zone-based hybrid feature extraction techniques. International Journal of Computer Applications 2012; 47: 24-29

[19] Dalal N, Triggs B. Histograms of oriented gradients for human detection. In: IEEE Computer Society Conference on Computer Vision and Pattern Recognition CVPR; 20–25 June 2005; San Diego, CA, USA. pp. 886-893.

[20] Li Y, Su G. Simplified histograms of oriented gradient features extraction algorithm for the hardware implementation. In: International Conference on Computers, Communications, and Systems; 2–3 November 2015; Kanyakumari, India. pp. 192-195.

[21] He K, Zhang X, Ren S, Sun J. Deep residual learning for image recognition. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition CVPR; 2016; Las Vegas, NV, USA. pp. 770-778.

[22] Biswas M, Islam R, Shom GK, Shopon M, Mohammed N, Momen S, Abedin A. BanglaLekha-Isolated: A multipurpose comprehensive dataset of handwritten Bangla isolated characters. Data in Brief 2017; 12: 103-107.

[23] Sokolova M, Lapalme G. A systematic analysis of performance measures for classification tasks. Inform Process Manag 2009; 45: 427-437.