Hindawi
Computational Intelligence and Neuroscience
Volume 2018, Article ID 6747098, 13 pages
https://doi.org/10.1155/2018/6747098

Research Article

Handwritten Bangla Character Recognition Using the State-of-the-Art Deep Convolutional Neural Networks

Md Zahangir Alom,[1] **Paheding Sidike**,[2] **Mahmudul Hasan**,[3] **Tarek M. Taha**,[1] **and Vijayan K. Asari**[1]

[1]*Department of Electrical and Computer Engineering, University of Dayton, Dayton, OH, USA*
[2]*Department of Earth and Atmospheric Sciences, Saint Louis University, St. Louis, MO, USA*
[3]*Comcast Labs, Washington, DC, USA*

Correspondence should be addressed to Md Zahangir Alom; alomm1@udayton.edu

Received 1 March 2018; Revised 10 May 2018; Accepted 10 July 2018; Published 27 August 2018

Academic Editor: Friedhelm Schwenker

In spite of advances in object recognition technology, handwritten Bangla character recognition (HBCR) remains largely unsolved due to the presence of many ambiguous handwritten characters and excessively cursive Bangla handwritings. Even many advanced existing methods do not lead to satisfactory performance in practice that related to HBCR. In this paper, a set of the state-of-the-art deep convolutional neural networks (DCNNs) is discussed and their performance on the application of HBCR is systematically evaluated. The main advantage of DCNN approaches is that they can extract discriminative features from raw data and represent them with a high degree of invariance to object distortions. The experimental results show the superior performance of DCNN models compared with the other popular object recognition approaches, which implies DCNN can be a good candidate for building an automatic HBCR system for practical applications.

1. Introduction

Automatic handwriting character recognition has many academic and commercial interests. The main challenge in handwritten character recognition is to deal with the enormous variety of handwriting styles by different writers. Furthermore, some complex handwriting scripts comprise different styles for writing words. Depending on the language, characters are written isolated from each other in some cases (e.g., Thai, Laos, and Japanese). In some other cases, they are cursive and sometimes characters are related to each other (e.g., English, Bangladeshi, and Arabic). This challenge has been already recognized by many researchers in the field of natural language processing (NLP) [1–3].

Handwritten character recognition is more challenging compared with the printed forms of character due to the following reasons: (1) Handwritten characters written by different writers are not only nonidentical but also vary in different aspects such as size and shape; (2) numerous variations in writing styles of individual character make the recognition task difficult; (3) the similarities of different character in shapes, the overlaps, and the interconnections of the neighbouring characters further complicate the character recognition problem. In summary, a large variety of writing styles and the complex features of the handwritten characters make it a challenge to accurately classifying handwritten characters.

Bangla is one of the most spoken languages and ranked fifth in the world and spoken by more than 200 million people [4, 5]. It is the national and official language of Bangladesh and the second most popular language in India. In addition, Bangla has a rich heritage. February 21st is announced as the International Mother Language day by UNESCO to respect the language martyrs for the language in Bangladesh in the year of 1952. In terms of Bangla character, it involves a Sanskrit-based script that is inherently different from English- or Latin-based scripts, and it is relatively difficult to achieve desired accuracy on the recognition tasks.

Therefore, developing a recognition system for Bangla characters is of a great interest [4, 6, 7].

In Bangla language, there are 10 digits and 50 characters including vowel and consonant, where some contain additional sign up and/or below. Moreover, Bangla consists of many similar shaped characters. In some cases, a character differs from its similar one with a single dot or mark. Furthermore, Bangla also contains some special characters with equivalent representation of vowels. This makes it difficult to achieve a better performance with simple classification technique as well as hinders to the development of a reliable handwritten Bangla character recognition (HBCR) system.

There are many applications of HBCR, such as Bangla optical character recognition, national ID number recognition system, automatic license plate recognition system for vehicle and parking lot management system, post office automation, and online banking. Some example images of these applications are shown in Figure 1. In this work, we investigate the HBCR on Bangla numerals, alphabets, and special characters using the state-of-the-art deep convolutional neural network (DCNN) [8] models. The contributions of this paper can be summarized as follows:

(i) First time to comprehensive evaluation of the state-of-the-art DCNN models including VGG Net [9], All Convolutional Neural Network (All-Conv) [10], Network in Network (NiN) [11], Residual Network (ResNet) [12], Fractal Network (FractalNet) [13], and Densely connected convolutional Network (DenseNet) [14] on the application of HBCR.

(ii) Extensive experiments on HBCR including handwritten digits, alphabets, and special character recognition.

(iii) The better recognition accuracy is achieved, to the best of knowledge, compared with other existing approaches that reported in the literature.

2. Related Work

Although some studies on Bangla character recognition have been reported in the past years [15–17], there is a few remarkable works available for HBCR. Pal and Chaudhuri [5] proposed a new feature extraction-based method for handwritten Bangla character recognition where the concept of water overflow from the reservoir is utilized. Liu and Suen [18] introduced directional gradient features for handwritten Bangla digit classification using ISI Bangla numeral dataset [19], which consists of 19,392 training samples, 4000 test samples, and 10 classes (i.e., 0 to 9). Surinta et al. [20] proposed a system using a set of features such as the contour of the handwritten image computed using 8-directional codes, distance calculated between hotspots and black pixels, and the intensity of pixel space of small blocks. Each of these features is separately fed into support vector machine (SVM) [21] classifier, and the final decision is made by the majority voting strategy. Das et al. [22] exploited genetic algorithms-based region sampling method for local feature selection and achieved 97% accuracy on HBCR. Xu et al. [23] used a hierarchical Bayesian network which directly takes raw images as the network inputs and classifies them using a bottom-up approach. Sparse representation classifier has also been applied for Bangla digit recognition [4], where 94% accuracy was reported for handwritten digit recognition. In [6], handwritten Bangla basic and compound character recognition using multilayer perceptron (MLP) [24] and SVM classifier was suggested, while handwritten Bangla numeral recognition using MLP was presented in [7] where the average recognition rate reached 96.67%.

Recently, deep learning-based methods have drawn increasing attention in handwritten character recognition [25, 26]. Ciregan and Meier [27] applied multicolumn CNNs to Chinese character classification. Kim and Xie [25] applied DCNN to Hangul handwritten character recognition and superior performance has been achieved against classical methods. A deep learning framework such as a CNN-based HBCR scheme was introduced in [26] where the best recognition accuracy reached at 85.36% on their own dataset. In this paper, we, for the first time, introduce the very latest DCNN models, including VGG network, All-Conv, NiN, ResNet, FractalNet, and DenseNet, for handwritten Bangla character (e.g., digits, alphabets, and special characters) recognition.

3. Deep Neural Networks

Deep neural network (DNN) is an active area in the field of machine learning and computer vision [28] and it generally contains three popular architectures: Deep Belief Net (DBN) [29], Stacked Autoencoder (SAE) [30], and CNN. Due to the composition of many layers, DNN methods are more capable for representing the highly varying nonlinear function compared with shallow learning approaches [31]. The low and middle level of DNN abstract the feature from the input image, whereas the high level performs classification operation on the extracted features. As a result, an end-to-end framework is formed by integrating with all necessary layers within a single network. Therefore, DNN models often lead to better accuracy compared with the other type of machine learning methods. Recent successful practice of DNN covers a variety of topics such as electricity consumption monitoring [32], radar signal examination [33], medical image analysis [34–36], food security [37–39], and remote sensing [40–42].

Among all deep learning approaches, CNN is one of the most popular models and has been providing the state-of-the-art performance on segmentation [43, 44], human action recognition [45], image superresolution [46], scene labelling [47], and visual tracking [48].

3.1. Convolutional Neural Network (CNN). CNN was initially applied to digit recognition task by LeCun et al. [8]. CNN and its variants are gradually adopted to various applications [46, 49]. CNN is designed to imitate human visual processing, and it has highly optimized structures to process 2D images. Furthermore, CNN can effectively learn the extraction and abstraction of 2D features. In detail, the

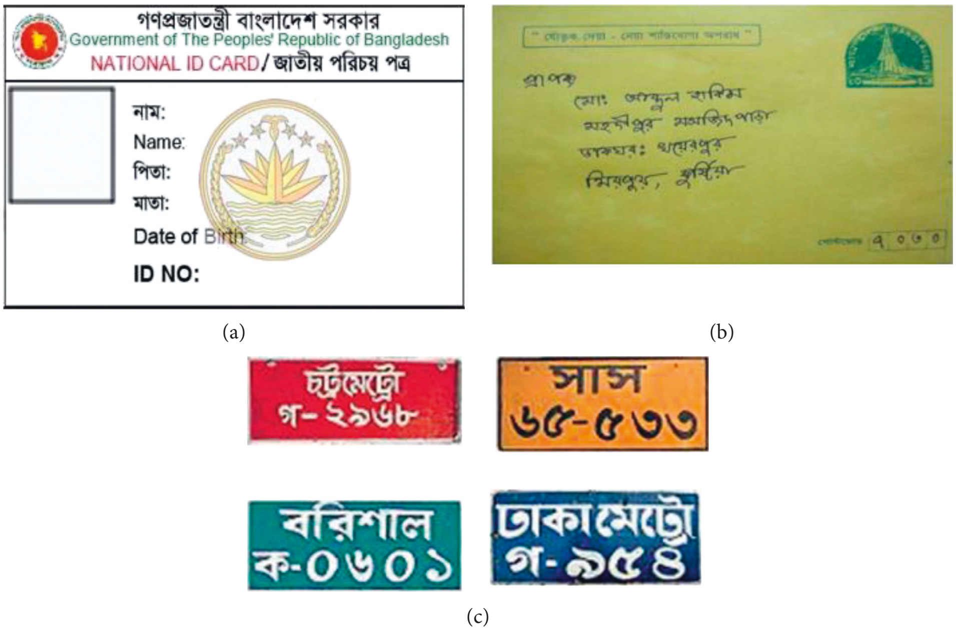

গণপ্রজাতন্ত্রী বাংলাদেশ সরকার
Government of The Peoples' Republic of Bangladesh
NATIONAL ID CARD/ জাতীয় পরিচয় পত্র
নাম:
Name:
পিতা:
মাতা:
Date of Birth
ID NO:
(a)
(b)
চট্টমেট্রো
গ-২৯৬৮
সাস
৬৫-৫৩৩
বরিশাল
ক-০৬০১
ঢাকামেট্রো
গ-৯৫৪
(c)

Figure 1: Application of handwritten character recognition: (a) national ID number recognition system, (b) post office automation with code number recognition on envelope, and (c) automatic license plate recognition.

max-pooling layer of CNN is very effective in absorbing shape variations. Moreover, sparse connection with tied weights makes CNN involve with fewer parameters than a fully connected network with similar size. Most importantly, CNN is trainable with the gradient-based learning algorithm and suffers less from the diminishing gradient problem. Given that the gradient-based algorithm trains the whole network to minimize an error criterion directly, CNN can produce highly optimized weights and good generalization performance [50].

The overall architecture of a CNN, as shown in Figure 2, consists of two main parts: feature extractor and classifier. In the feature extraction unit, each layer of the network receives the output from its immediate previous layer as inputs and passes current output as inputs to the immediate next layer, whereas classification part generates the predicted outputs associated with the input data. The two basic layers in CNN architecture are convolution and pooling [8] layers. In convolution layer, each node extracts the features from the input images by convolution operation on the input nodes. The max-pooling layer abstracts the feature through average or maximum operation on input nodes. The outputs of $l-1$th layer are used as input for the lth layer, where the inputs go through a set of kernels followed by nonlinear function ReLU. Here, f refers to activation function of ReLU. For example, if x_i^{l-1} inputs from $l-1$th layer, $k_{i,j}^l$ are kernels of lth layer. The biases of lth layer are represented with b_j^l. Then, the convolution operation can be expressed as

$$x_j^l = f\left(x_i^{l-1} * k_{i,j}^l\right) + b_j^l. \quad (1)$$

The subsampling or pooling layer abstracts the feature through average or maximum operation on input nodes. For example, if a 2 × 2 down sampling kernel is applied, then each output dimension will be the half of the corresponding input dimension for all the inputs. The pooling operation can be stated as follows:

$$x_j^l = \operatorname{down}\left(x_i^{l-1}\right). \quad (2)$$

In contrast to traditional neural networks, CNN extracts low- to high-level features. The higher-level features can be derived from the propagated feature of the lower-level layers. As the features propagate to the highest layer, the dimension of the feature is reduced depending on the size of the convolution and pooling masks. However, the number of feature mapping usually increased for selecting or mapping the extreme suitable features of the input images for better classification accuracy. The outputs of the last layer of CNN are used as inputs to the fully connected network and it typically uses a Softmax operation to produce the classification outputs. For an input sample $\mathbf{x}$, weight vector $\mathbf{w}$, and K distinct linear functions, the Softmax operation can be defined for the ith class as follows:

$$P(y = i|\mathbf{x}) = \frac{\exp\left(\mathbf{x}^T \mathbf{w}_i\right)}{\sum_{k=1}^{K} \exp\left(\mathbf{x}^T \mathbf{w}_k\right)}. \quad (3)$$

However, there are different variants of DCNN architecture that have been proposed over the last few years. The following section discusses six popular DCNN models.

3.2. CNN Variants. As far as CNN architecture is concerned, it can be observed that there are some important and fundamental components that are used to construct an efficient DCNN architecture. These components are convolution layer, pooling layer, fully connected layer, and

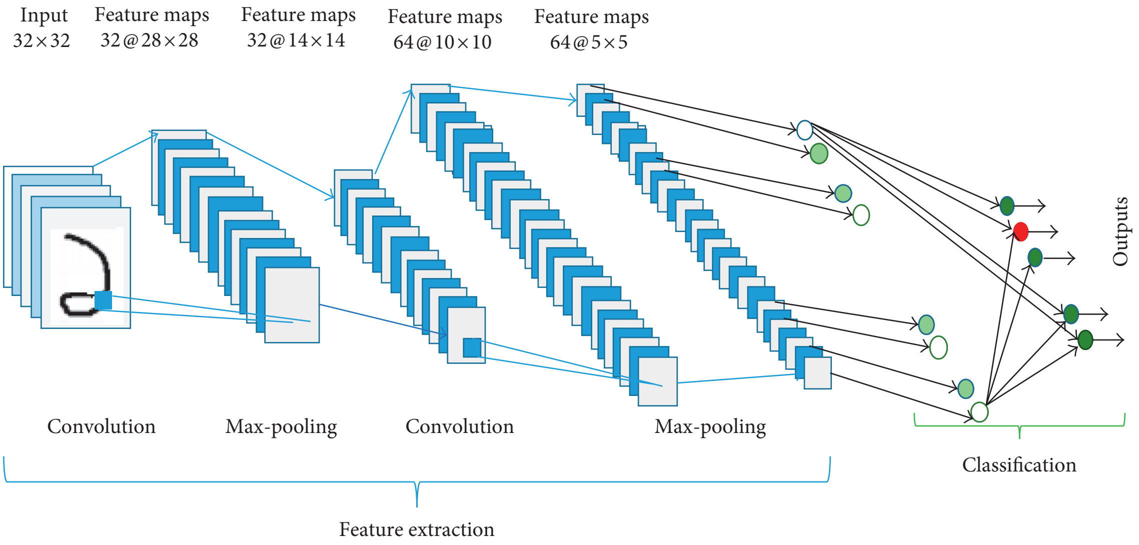


Figure 2: Basic CNN architecture for digit recognition.

Softmax layer. The advanced architecture of this network consists of a stack of convolutional layers and max-pooling layers followed by fully connected and Softmax layer at the end. Noticeable examples of such networks include LeNet [8], AlexNet [49], VGG Net, All-Conv, and NiN. There are some other alternative and advanced architecture that have been proposed, including GoogleNet with inception layers [51], ResNet, FractalNet, and DenseNet. However, there are some topological differences observed in the modern architectures. Out of many DCNN architectures, AlexNet, VGG Net, GoogleNet, ResNet, DenseNet, and FractalNet can be viewed as most popular architectures with respect to their enormous performance on different benchmarks for object classification. Among these models, some of the models are designed especially for large-scale implementation such as ResNet and GoogleNet, whereas the VGG Net consists of a general architecture. On the other hand, FractalNet is an alternative of ResNet. In contrast, DenseNet's architecture is unique in terms of unit connectivity where every layer to all subsequent layers is directly connected. In this paper, we provide a review and comparative study of All-Conv, NiN, VGG-16, ResNet, FractalNet, and DenseNet for Bangla character recognition. The basic overview of these architectures is given in the following section.

3.2.1. VGG-16. The visual geometry group (VGG) was the runner up of the ImageNet Large Scale Visual Recognition Competition (ILSVRC) in 2014 [52]. In this architecture, two convolutional layers are used consecutively with a rectified linear unit (ReLU) [53] activation function followed by single max-pooling layer, several fully connected layers with ReLU and Softmax as the final layer. There are three types of VGG Net based on the architecture. These three networks contain 11, 16, and 19 layers and named as VGG-11, VGG-16, and VGG-19, respectively. The basic structure for VGG-11 architecture contains eight convolution layers, one max-pooling layer, and three fully connected (FC) layers followed by single Softmax layer. The configuration of VGG-16 is as follows: the number of convolutions and max-pooling layers: 13, max-pooling layer: 1, FC layers: 3, and Softmax layer: 1. Total weights is 138 million. The VGG-19 consisted of 16 convolutional layers, one max-pooling layer, 3 FC layers followed by a Softmax layer. The basic building blocks of VGG architecture is shown in Figure 3. In this implementation, we have used VGG-16 network with less number of feature maps in convolutional layers compared with the standard VGG-16 network.

3.2.2. All Convolutional Network (All-Conv). The layer specification of All-Conv is given in Figure 4. The basic architecture is composed with two convolutional layers followed by a max-pooling layer. Instead of using fully connected layer, global average pooling (GAP) [11] with the dimension of 6 × 6 is used. Finally, the Softmax layer is used for classification. The output dimension is assigned based on the number of classes.

3.2.3. Network in Network (NiN). This model is quite different compared with the aforementioned DCNN models due to the following properties [11]:

(i) It uses multilayer convolution where convolution is performed with 1 × 1 filters.

(ii) It uses GAP instead of fully connected layer.

The concept of using 1 × 1 convolution helps to increase the depth of the network. The GAP significantly changes the network structure, which is used nowadays very often as a replacement of fully connected layers. The GAP on a large feature map is used to generate a final low-dimensional feature vector instead of reducing the feature map to a small size and then flattening the feature vector.

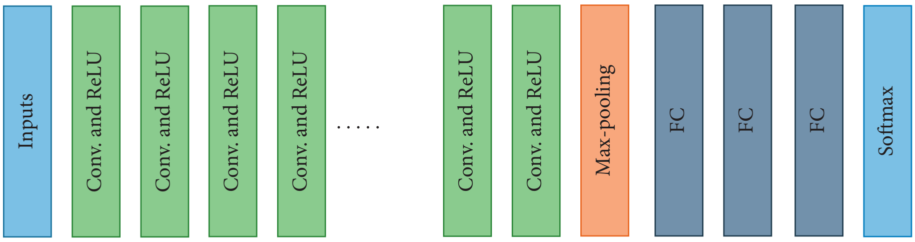


Figure 3: Basic architecture of VGG Net: convolution (Conv) and FC for fully connected layers and Softmax layer at the end.

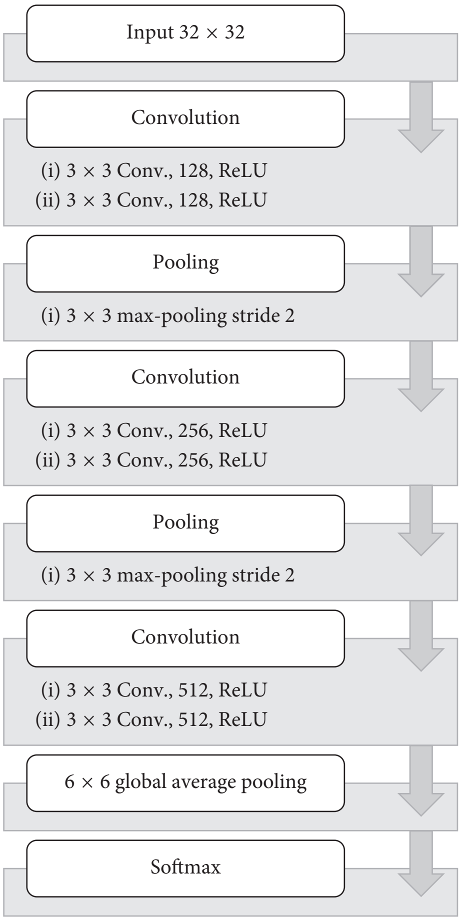

Input 32 × 32
Convolution
(i) 3 × 3 Conv., 128, ReLU
(ii) 3 × 3 Conv., 128, ReLU
Pooling
(i) 3 × 3 max-pooling stride 2
Convolution
(i) 3 × 3 Conv., 256, ReLU
(ii) 3 × 3 Conv., 256, ReLU
Pooling
(i) 3 × 3 max-pooling stride 2
Convolution
(i) 3 × 3 Conv., 512, ReLU
(ii) 3 × 3 Conv., 512, ReLU
6 × 6 global average pooling
Softmax

Figure 4: All convolutional network framework.

3.2.4. Residual Network (ResNet). ResNet architecture becomes very popular in computer vision community. The ResNet variants have been experimented with different number of layers as follows: number of convolution layers: 49 (34, 152, and 1202 layers for other versions of ResNet), number of fully connected layers: 1, weights: 25.5 M. The basic block diagram of ResNet architecture is shown in Figure 5. If the input of the residual block is x_{l-1}, the output of this block is x_l. After performing operations (e.g., convolution with different size of filters, batch normalization (BN) [54] followed by a activation function such as ReLU) on x_{l-1}, the output $F(x_{l-1})$ is produced. The final output of the residual unit is defined as

$$x_l = F(x_{l-1}) + x_{l-1}. \tag{4}$$

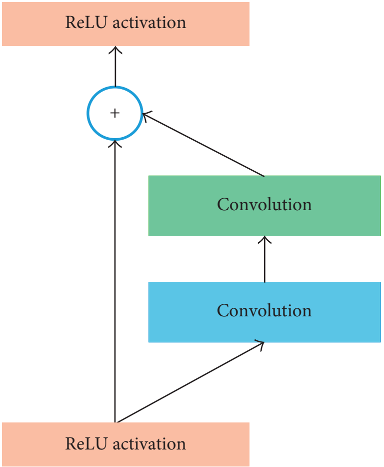


Figure 5: Basic diagram of residual block.

The Residual Network consists of several basic residual units. The different residual units are proposed with different types of layers. However, the operations between the residual units vary depending on the architectures that are explained in [12].

3.2.5. FractalNet. The FractalNet architecture is an advanced and alternative one of ResNet, which is very efficient for designing very large network with shallow subnetworks, but shorter paths for the propagation of gradient during training [13]. This concept is based on drop path which is another regularization for large network. As a result, this concept helps to enforce speed versus accuracy tradeoff. The basic block diagram of FractalNet is shown in Figure 6. Here x is the actual inputs of FractalNet, and z and $f(z)$ are the inputs and outputs of Fractal block, respectively.

3.2.6. Densely Connected Network (DenseNet). DenseNet is densely connected CNN where each layer is connected to all previous layers [14]. Therefore, it forms very dense connectivity between the layers and so it is called DenseNet. The DenseNet consists of several dense blocks, and the layer between two adjacent blocks is called transition layers. The conceptual diagram of the dense block is shown in Figure 7. According to the figure, the lth layer receives all the feature maps $x_0, x_1, x_2, \ldots, x_{l-1}$ from the previous layers as input, which is expressed by

$$x_l = H_l([x_0, x_1, x_2, \ldots, x_{l-1}]), \tag{5}$$

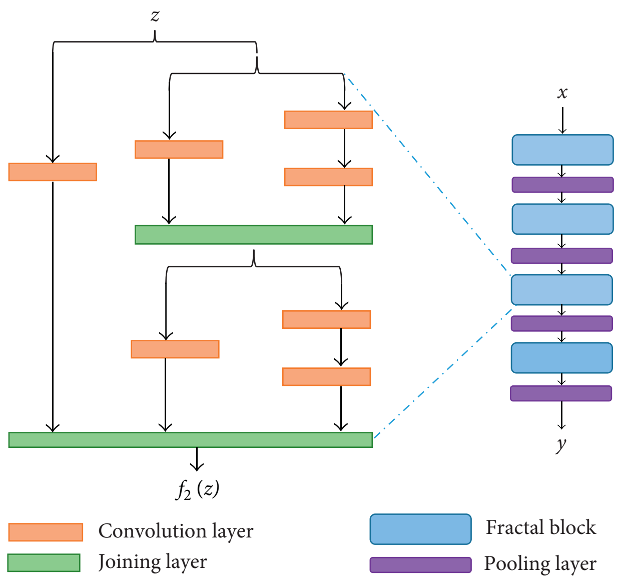


Figure 6: An example of FractalNet architecture.

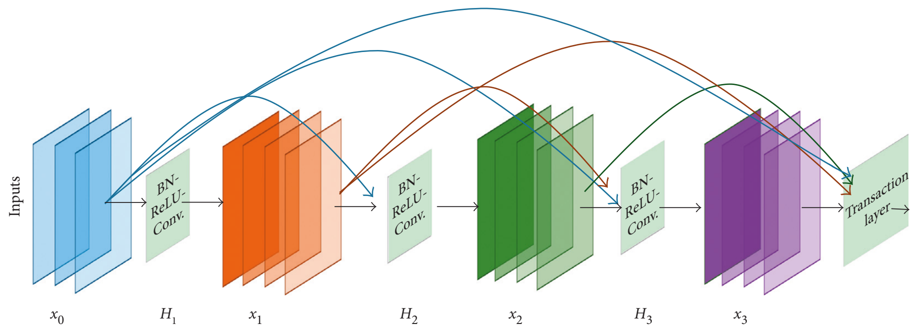


Figure 7: A 4-layer dense block with growth rate of $k = 3$. Each of the layers takes all of the preceding feature maps as input.

where $[x_0, x_1, x_2, \ldots, x_{l-1}]$ is the concatenated features from $0, \ldots, l-1$ layers and $H_l(\cdot)$ is a single tensor. DenseNet performs three consecutive operations, BN, followed by ReLU and a 3 × 3 convolution. In the transition block, 1 × 1 convolutional operations are performed with BN followed by 2 × 2 average pooling layer. This new architecture has achieved state-of-the-art accuracy for object recognition on the five different competitive benchmarks.

3.2.7. Network Parameters. The number of network parameters is a very important criterion to assess the complexity of the architecture. The number of parameters can be used to make comparison between different architectures. At first, the dimension of the output feature map can be computed as

$$M = \frac{N - F}{S} + 1, \tag{6}$$

where N denotes the dimension of input feature maps, F refers to the dimension of filters or receptive field, S represents stride in the convolution, and M is the dimension of output feature maps. The number of parameters (without bias) for a single layer is obtained by

$$P_l = \left(F \times F \times FM_{l-1}\right) \times FM_l, \tag{7}$$

where P_l represents the total number of parameters in the lth layer, FM_l is the total number of output feature maps of lth layer, and FM_{l-1} is the total number of feature maps in the $(l-1)$th layer. For example, let a 32 × 32 dimensional (N) image be an input. The size of the filter (F) is 5 × 5 and stride (S) is 1 for convolutional layer. The output dimension (M) of the convolutional layer is 28 × 28 which is calculated according to (6). For better illustration, a summary of parameters used in All-Conv architecture is shown in Table 1. Note that the number of trainable parameters is zero in the pooling layer.

4. Results and Discussion

The entire experiment is performed on desktop computer with Intel® Core-I7 CPU @ 3.33 GHz, 56.00 GB memory, and Keras with Theano on the backend on Linux environment. We evaluate the state-of-the-art DCNN models on three datasets from CMATERdb (available at: https://code.google.com/archive/p/cmaterdb/) containing Bangla handwritten digits, alphabets, and special character recognition.

Table 1: Parameter specification in All-Conv model.

Layers	Operations	Feature maps	Size of feature maps	Size of kernels	Number of parameters
Inputs		32 × 32 × 3			
C_1	Convolution	128	30 × 30	3 × 3	3,456
C_2	Convolution	128	28 × 28	3 × 3	147,456
S_1	Max-pooling	128	14 × 14	2 × 2	N/A
C_3	Convolution	256	12 × 12	3 × 3	294,912
C_4	Convolution	256	10 × 10	3 × 3	589,824
S_2	Max-pooling	256	5 × 5	2 × 2	N/A
C_5	Convolution	512	3 × 3	3 × 3	1,179,648
C_6	Convolution	512	3 × 3	1 × 1	262,144
GAP_1	GAP	512	3 × 3	N/A	N/A
Outputs	Softmax	10	1x1	N/A	5,120

The statistics of three datasets used in this paper are summarized in Table 2. For convenience, we named the datasets as Digit-10, Alphabet-50, and SpecialChar-13, respectively. All images are rescaled to 32 × 32 pixels in our experiment.

Table 2: Statistics of the database used in our experiment.

Dataset	# training samples	# testing samples	Total samples	Number of classes
Digit-10	4000	2000	6000	10
Alphabet-50	12,000	3,000	15,000	50
SpecialChar-13	2196	935	2231	13

4.1. Bangla Handwritten Digit Dataset. The standard samples of the numeral with respective Arabic numerals are shown in Figure 8. The performance of both DBN and CNN is evaluated on a Bangla handwritten benchmark dataset called CMATERdb 3.1.1 [22]. This dataset contains 6,000 images of unconstrained handwritten isolated Bangla numerals. Each digit has 600 images that are rescaled to 32 × 32 pixels. Some sample images in the database are shown in Figure 9. Visual inspection depicts that there is no visible noise. However, variability in writing style is quite high due to user dependency. In our experiments, the dataset is split into a training set and a test set for the evaluation of different DCNN models. The training set consists of 4,000 images (400 randomly selected images of each digit). The rest of the 2,000 images are used for testing.

Figure 10 shows the training loss of all DCNN models during 250 epochs. It can be observed that FractalNet and DenseNet converge faster compared with other networks, and worst convergence is obtained to be for the All-Conv Network. The validation accuracy is shown in Figure 11, where DenseNet and FractalNet show better recognition accuracy among all DCNN models. Finally, the testing accuracy of all the DCNN models is shown in Figure 12. From the result, it can be clearly seen that DenseNet provides the best recognition accuracy compared with other networks.

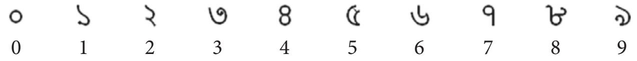


Figure 8: First row shows the Bangla actual digits and second row shows the corresponding Arabic numerals.

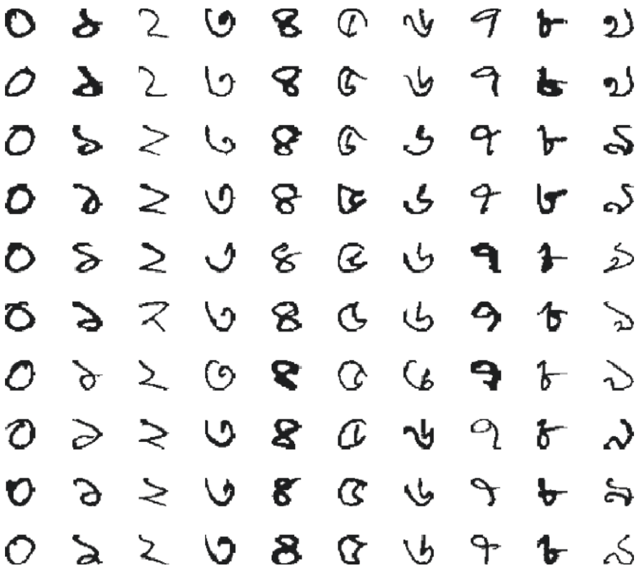


Figure 9: Sample handwritten Bangla numeral images from CMATERdb 3.1.1 database, including digits from 1 to 10.

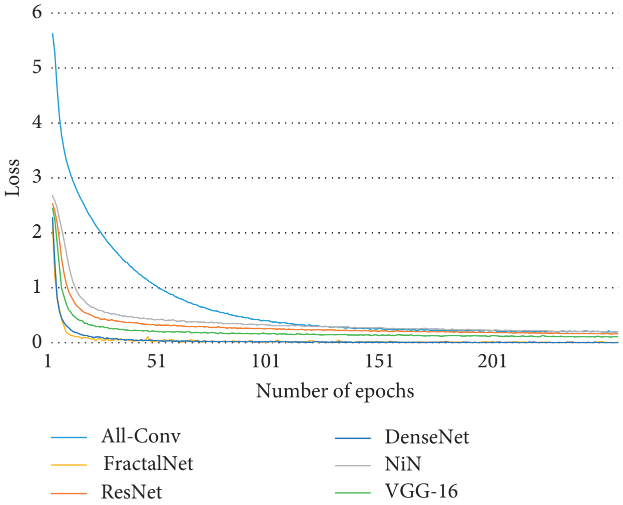


Figure 10: Training loss of different architectures for Bangla handwritten 1–10 digits.

4.2. Bangla Handwritten Alphabet-50. In our implementation, the basic fifty alphabets including 11 vowels and 39 consonants are considered. The samples of 39-consonant and 11-vowel characters are shown in Figures 13(a) and 13(b), respectively. The Alphabet-50 dataset contains 15,000 samples, where 12,000 are used for training and the remaining 3,000 samples are used for testing. Since the dataset contains samples with different dimensions, we rescale all input images to 32 × 32 pixels for better fitting to the convolutional operation. Some randomly selected samples from this database are shown in Figure 14.

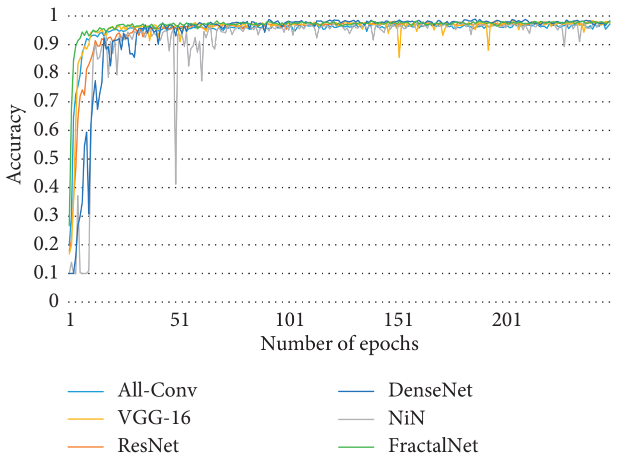


Figure 11: Validation accuracy of different architectures for Bangla handwritten 1–10 digits.

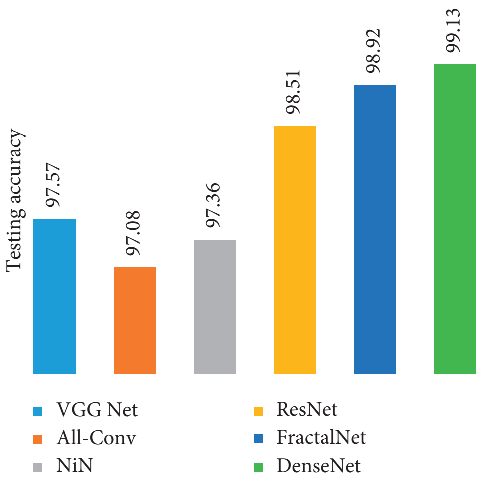
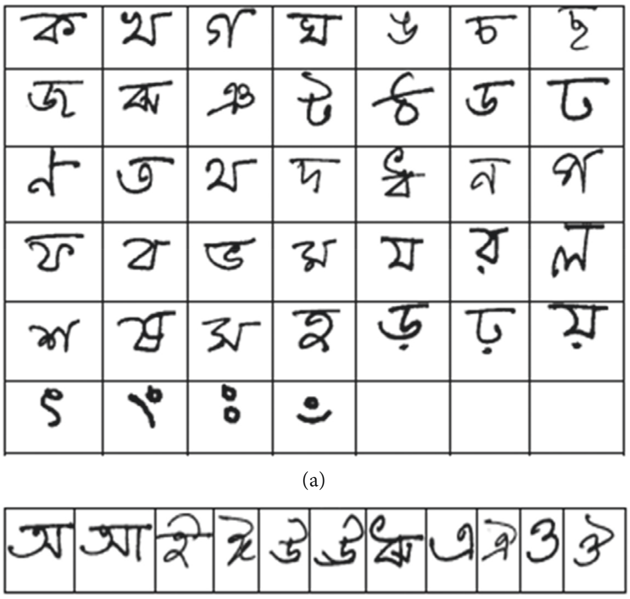


Figure 12: Testing accuracy for Bangla handwritten digit recognition.

The training loss for different DCNN models is shown in Figure 15. It is clear that the DenseNet shows the best convergence compared with the other DCNN approaches. Similar to the previous experiment, All-Conv shows the worst convergence behavior. In addition, an unexpected convergence behavior is observed in the case of NiN model. However, all DCNN models tend to converge after 200 epochs. The corresponding validation accuracy on Alphabet-50 is shown in Figure 16. DenseNet again shows superior validation accuracy compared with other DCNN approaches.

Figure 17 shows the testing results on handwritten Alphabet-50. The DenseNet shows the best testing accuracy with a recognition rate of 98.31%. On the other hand, the All-Conv Net provides around 94.31% testing accuracy, which is the lowest testing accuracy among all the DCNN models.



(a)

(b)

Figure 13: Example images of handwritten characters: (a) Bangla consonant characters and (b) vowels.

4.3. Bangla Handwritten Special Characters. There are several special characters (SpecialChar-13) which are equivalent to representations of vowels that are combined with consonants for making meaningful words. In our evaluation, we use 13 special characters which are for 11 vowels and two additional special characters. Some samples of Bangla special characters are shown in Figure 18. It can be seen that the quality of the samples is poor, and significant variation in the same symbols makes this recognition task even difficult.

The training loss and validation accuracy for SpecialChar-13 are shown in Figures 19 and 20, respectively. From these results, it can be seen that DenseNet provides better performance with lower loss and with the highest validation accuracy among all DCNN models. Figure 21 shows the testing accuracy of DCNN model for SpecialChar-13 dataset. It is observed from Figure 21 that DenseNet shows the highest testing accuracy with lowest training loss and it converges very fast. On the other hand, VGG-19 network shows promising recognition accuracy as well.

4.4. Performance Comparison. The testing performance is compared to several existing methods. The results are presented in Table 3. The experimental results show that the modern DCNN models including DenseNet, FractalNet, and ResNet provide better testing accuracy against the other deep learning approaches and the previously proposed classical methods. In general, the DenseNet provides 99.13% testing accuracy for handwritten digit recognition, which is the best accuracy that has been publicly reported to the best our knowledge. In case of a 50-alphabet recognition, DenseNet yields 98.31% recognition accuracy, which is almost 2.5% better than the method in [55]. As far as we know, this is the highest accuracy for handwritten Bangla 50-alphabet recognition. In addition, on 13 special character recognition task, DCNNs show promising recognition accuracy, especially DenseNet achieves the best accuracy which is 98.18%.

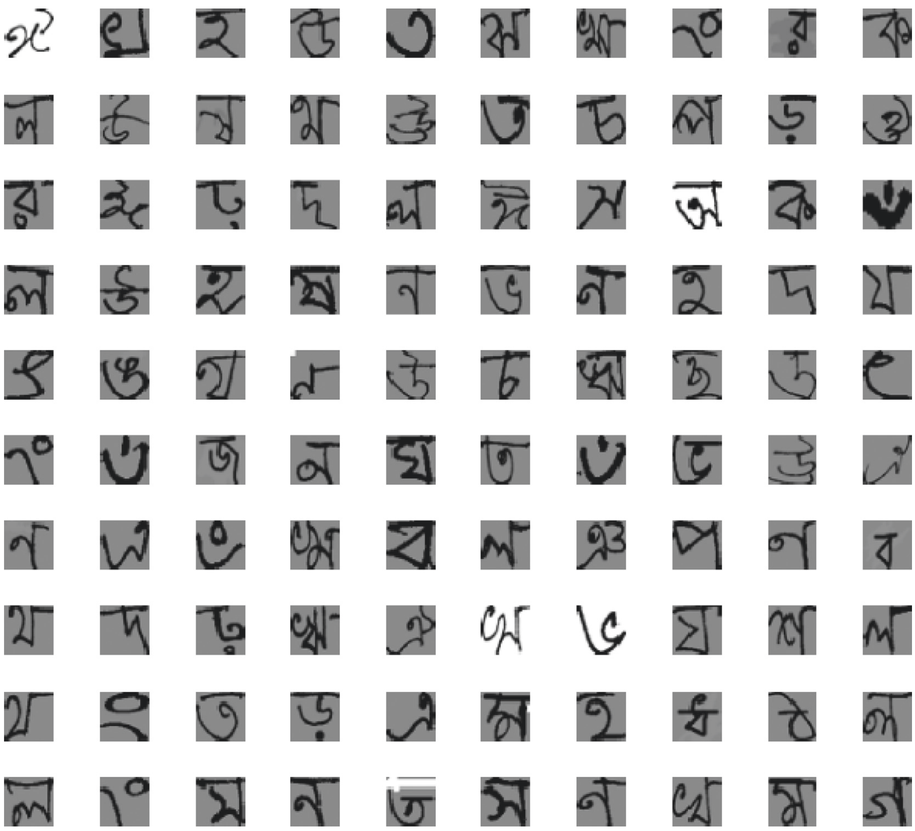


Figure 14: Randomly selected handwritten characters of Bangla alphabets from Bangla handwritten Alphabet-50 dataset.

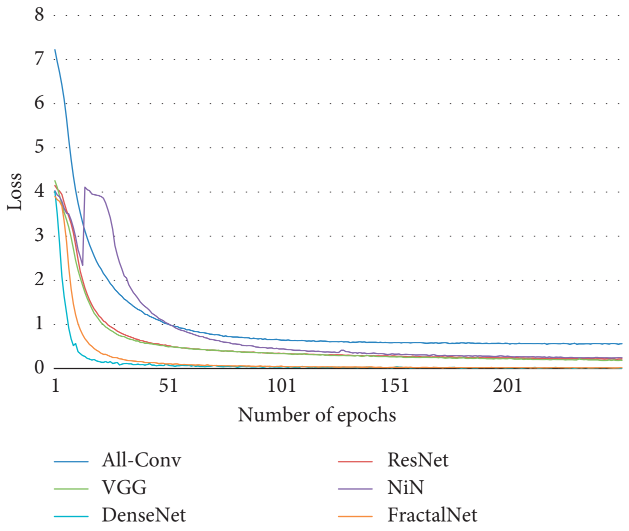


Figure 15: Training loss of different DCNN models for Bangla handwritten Alphabet-50.

4.5. Parameter Evaluation. For impartial comparison, we have trained and tested the networks with the optimized same number of parameters as in the references. Table 4 shows the number of parameters used for different networks for 50-alphabet recognition. The number of network parameters for digits and special character recognition was the same except the number of neurons in the classification layer.

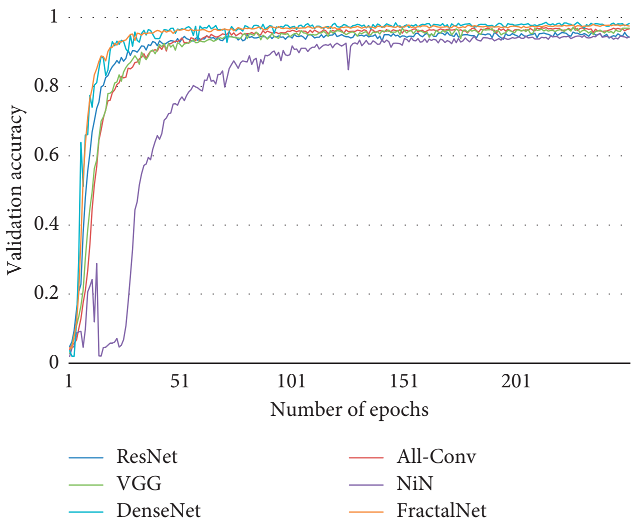


Figure 16: The validation accuracy of different architectures for Bangla handwritten Alphabet-50.

4.6. Computation Time. We also calculate computational cost for all methods, although the computation time depends on the complexity of the architecture. Table 5 presents the computational time per epoch (in second) during training of all the networks for Digit-10, Alphabet-50, and SpecialChar-13 recognition task. From Table 5, it can be seen that DenseNet takes the longest time during training due to its dense structure but yields the best accuracy.

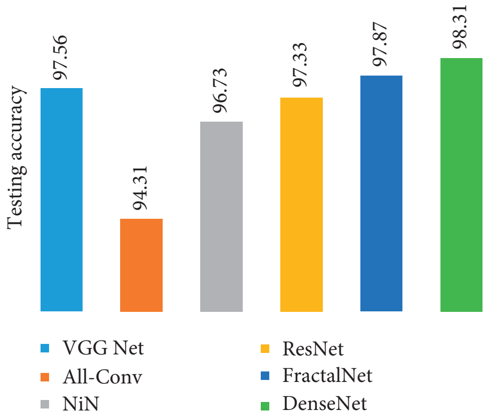


Figure 17: Testing accuracy for handwritten 50-alphabet recognition using different DCNN techniques.

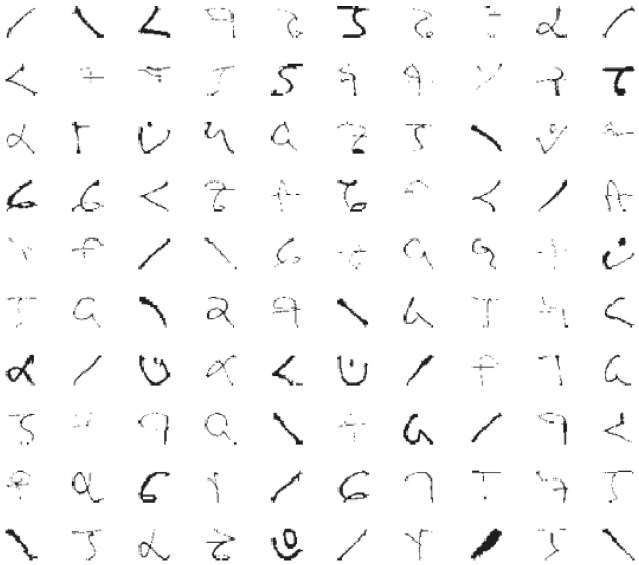


Figure 18: Randomly selected images of special character from the dataset.

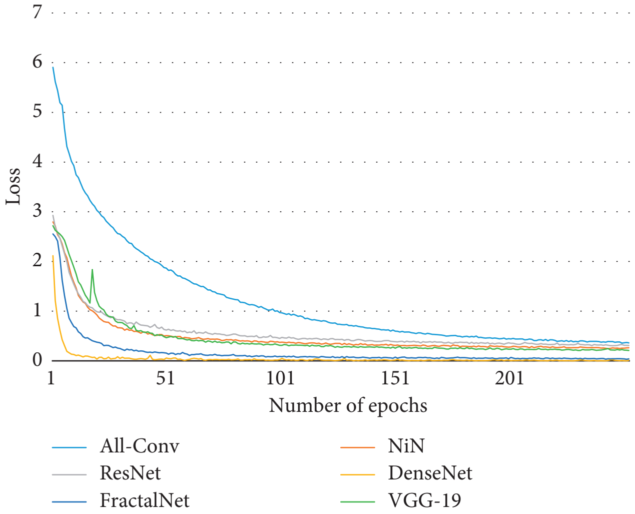


Figure 19: Training loss of different architectures for Bangla 13 special characters (SpecialChar-13).

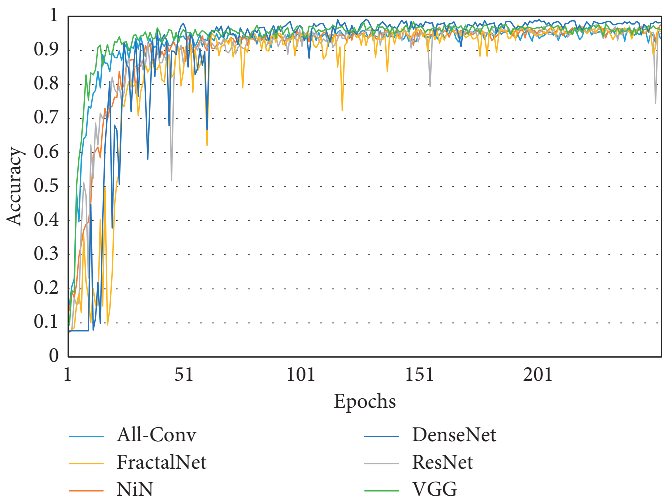


Figure 20: Validation accuracy of different architectures for Bangla 13 special characters (SpecialChar-13).

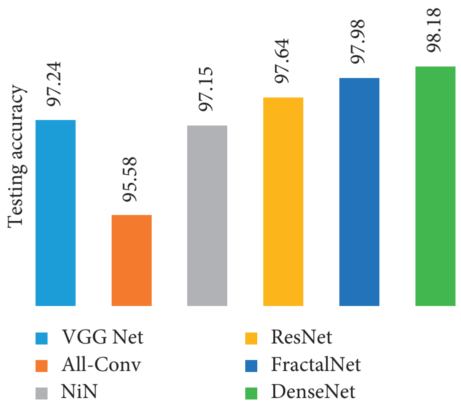


Figure 21: Testing accuracy of different architectures for Bangla 13 special characters (SpecialChar-13).

Table 3: The testing accuracy of VGG-16 Network, All-Conv Network, NiN, ResNet, FractalNet, and DenseNet on Digit-10, Alphabet-50, and SpecialChar-13 and comparison against other existing methods.

Types	Methods	Digit-10 (%)	Alphabet-50 (%)	Special Char-13 (%)
Existing approaches	MLP [7]	96.67	—	—
	MPCA + QTLR [56]	98.55	—	—
	GA [22]	97.00	—	—
	LeNet + DBN [57]	98.64	—	—
	VGG Net [9]	97.57	97.56	96.15
DCNN	All-Conv [10]	97.08	94.31	95.58
	NiN [11]	97.36	96.73	97.24
	ResNet [12]	98.51	97.33	97.64
	FractalNet [13]	98.92	97.87	97.98
	DenseNet [14]	**99.13**	**98.31**	**98.18**

Table 4: Number of parameter comparison.

Models	Number of parameters
VGG-16 [9]	~ 8.43M
All-Conv Net [10]	~ 2.26M
NiN [11]	~ 2.81M
ResNet [12]	~ 5.63M
FractalNet [13]	~ 7.84M
DenseNet [14]	~ 4.25M

Table 5: Computational time (in sec) per epoch for different DCNN models on Digit-10, Alphabet-50, and SpecialChar-13.

Models	Digit-10	Alphabet-50	SpecialChar-13
VGG-16 [9]	32	83	15
All-Conv Net [10]	7	23	4
NiN [11]	9	27	5
ResNet [12]	64	154	34
FractalNet [13]	32	102	18
DenseNet [14]	95	210	58

5. Conclusions

In this research, we investigated the performance of several popular deep convolutional neural networks (DCNNs) for handwritten Bangla character (e.g., digits, alphabets, and special characters) recognition. The experimental results indicated that DenseNet is the best performer in classifying Bangla digits, alphabets, and special characters. Specifically, we achieved recognition rate of 99.13% for handwritten Bangla digits, 98.31% for handwritten Bangla alphabet, and 98.18% for special character recognition using DenseNet. To the best of knowledge, these are the best recognition results on the CMATERdb dataset. In future, some fusion-based DCNN models, such as Inception Recurrent Convolutional Neural Network (IRCNN) [47], will be explored and developed for handwritten Bangla character recognition.

Data Availability

The data used to support the findings of this study are available at https://code.google.com/archive/p/cmaterdb/.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

References

[1] D. C. Cireşan, U. Meier, L. M. Gambardella, and J. Schmidhuber, "Deep, big, simple neural nets for handwritten digit recognition," *Neural Computation*, vol. 22, no. 12, pp. 3207–3220, 2010.

[2] U. Meier, D. C. Ciresan, L. M. Gambardella, and J. Schmidhuber, "Better digit recognition with a committee of simple neural nets," in *Proceedings of International Conference on Document Analysis and Recognition (ICDAR)*, pp. 1250–1254, Beijing, China, September 2011.

[3] W. Song, S. Uchida, and M. Liwicki, "Comparative study of part-based handwritten character recognition methods," in *Proceedings of International Conference on Document Analysis and Recognition (ICDAR)*, pp. 814–818, Beijing, China, September 2011.

[4] H. A. Khan, A. Al Helal, and K. I. Ahmed, "Handwritten Bangla digit recognition using sparse representation classifier," in *Proceedings of 2014 International Conference on Informatics, Electronics and Vision (ICIEV)*, pp. 1–6, IEEE, Dhaka, Bangladesh, May 2014.

[5] U. Pal and B. Chaudhuri, "Automatic recognition of unconstrained off-line Bangla handwritten numerals," in *Proceedings of Advances in Multimodal Interfaces–ICMI 2000*, pp. 371–378, Springer, Beijing, China, October 2000.

[6] N. Das, B. Das, R. Sarkar, S. Basu, M. Kundu, and M. Nasipuri, "Handwritten Bangla basic and compound character recognition using MLP and SVM classifier," 2010, http://arxiv.org/abs/1002.4040.

[7] S. Basu, N. Das, R. Sarkar, M. Kundu, M. Nasipuri, and D. K. Basu, "An MLP based approach for recognition of handwritten Bangla numerals," 2012, http://arxiv.org/abs/1203.0876.

[8] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[9] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2014, http://arxiv.org/abs/1409.1556.

[10] J. T. Springenberg, A. Dosovitskiy, T. Brox, and M. Riedmiller, "Striving for simplicity: The all convolutional net," 2014, http://arxiv.org/abs/1412.6806.

[11] M. Lin, Q. Chen, and S. Yan, "Network in network," 2013, http://arxiv.org/abs/1312.4400.

[12] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 770–778, Las Vegas, NV, USA, June-July 2016.

[13] G. Larsson, M. Maire, and G. Shakhnarovich, "Fractalnet: ultra-deep neural networks without residuals," 2016, http://arxiv.org/abs/1605.07648.

[14] G. Huang, Z. Liu, K. Q. Weinberger, and L. van der Maaten, "Densely connected convolutional networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, vol. 1, no. 2, p. 3, Honolulu, HI, USA, July 2017.

[15] B. Chaudhuri and U. Pal, "A complete printed Bangla OCR system," *Pattern Recognition*, vol. 31, no. 5, pp. 531–549, 1998.

[16] U. Pal, *On the development of an optical character recognition (OCR) system for printed Bangla script*, Ph.D. dissertation, Indian Statistical Institute, Kolkata, India, 1997.

[17] U. Pal and B. Chaudhuri, "Indian script character recognition: a survey," *Pattern Recognition*, vol. 37, no. 9, pp. 1887–1899, 2004.

[18] C.-L. Liu and C. Y. Suen, "A new benchmark on the recognition of handwritten Bangla and Farsi numeral characters," *Pattern Recognition*, vol. 42, no. 12, pp. 3287–3295, 2009.

[19] B. Chaudhuri, "A complete handwritten numeral database of Bangla–a major Indic script,"in *Proceedings of Tenth International Workshop on Frontiers in Handwriting Recognition*, Suvisoft, Baule, France, October 2006.

[20] O. Surinta, L. Schomaker, and M. Wiering, "A comparison of feature and pixel-based methods for recognizing handwritten Bangla digits," in *Proceedings of 12th International Conference on Document Analysis and Recognition (ICDAR)*, pp. 165–169, IEEE, Buffalo, NY, USA, 2013.

[21] C. Cortes and V. Vapnik, "Support-vector networks," *Machine Learning*, vol. 20, no. 3, pp. 273–297, 1995.

[22] 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, pp. 1592–1606, 2012.

[23] J.-W. Xu, J. Xu, and Y. Lu, "Handwritten Bangla digit recognition using hierarchical Bayesian network," in *Proceedings of 3rd International Conference on Intelligent System and Knowledge Engineering*, vol. 1, pp. 1096–1099, IEEE, Xiamen, China, November 2008.

[24] D. E. Rumelhart, J. L. McClelland, P. R. Group et al., *Parallel Distributed Processing*, MIT Press, Vol. 1, MIT Press, Cambridge, MA, USA, 1987.

[25] I.-J. Kim and X. Xie, "Handwritten Hangul recognition using deep convolutional neural networks," *International Journal on Document Analysis and Recognition*, vol. 18, no. 1, pp. 1–13, 2015.

[26] M. M. Rahman, M. Akhand, S. Islam, P. C. Shill, and M. H. Rahman, "Bangla handwritten character recognition using convolutional neural network," *International Journal of Image, Graphics and Signal Processing*, vol. 7, no. 8, pp. 42–49, 2015.

[27] D. Ciregan and U. Meier, "Multi-column deep neural networks for offline handwritten Chinese character classification," in *Proceedings of 2015 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–6, IEEE, Killarney, Ireland, July 2015.

[28] A. Voulodimos, N. Doulamis, A. Doulamis, and E. Protopapadakis, "Deep learning for computer vision: a brief review," *Computational Intelligence and Neuroscience*, vol. 2018, Article ID 7068349, 13 pages, 2018.

[29] G. E. Hinton, S. Osindero, and Y.-W. Teh, "A fast learning algorithm for deep belief nets," *Neural Computation*, vol. 18, no. 7, pp. 1527–1554, 2006.

[30] P. Vincent, H. Larochelle, Y. Bengio, and P.-A. Manzagol, "Extracting and composing robust features with denoising autoencoders," in *Proceedings of the 25th International Conference on Machine Learning*, pp. 1096–1103, ACM, Helsinki, Finland, July 2008.

[31] Y. Bengio et al., "Learning deep architectures for AI," *Foundations and Trends® in Machine Learning*, vol. 2, no. 1, pp. 1–127, 2009.

[32] J. Kim, T.-T.-H. Le, and H. Kim, “Nonintrusive load monitoring based on advanced deep learning and novel signature,” *Computational Intelligence and Neuroscience*, vol. 2017, Article ID 4216281, 22 pages, 2017.

[33] E. Protopapadakis, A. Voulodimos, A. Doulamis, N. Doulamis, D. Dres, and M. Bimpas, “Stacked autoencoders for outlier detection in over-the-horizon radar signals,” *Computational Intelligence and Neuroscience*, vol. 2017, Article ID 5891417, 11 pages, 2017.

[34] H. Greenspan, B. van Ginneken, and R. M. Summers, “Guest editorial deep learning in medical imaging: overview and future promise of an exciting new technique,” *IEEE Transactions on Medical Imaging*, vol. 35, no. 5, pp. 1153–1159, 2016.

[35] D. Shen, G. Wu, and H.-I. Suk, “Deep learning in medical image analysis,” *Annual Review of Biomedical Engineering*, vol. 19, pp. 221–248, 2017.

[36] H.-C. Shin, H. R. Roth, M. Gao et al., “Deep convolutional neural networks for computer-aided detection: CNN architectures, dataset characteristics and transfer learning,” *IEEE Transactions on Medical Imaging*, vol. 35, no. 5, pp. 1285–1298, 2016.

[37] S. Sladojevic, M. Arsenovic, A. Anderla, D. Culibrk, and D. Stefanovic, “Deep neural networks based recognition of plant diseases by leaf image classification,” *Computational Intelligence and Neuroscience*, vol. 2016, Article ID 3289801, 11 pages, 2016.

[38] S. P. Mohanty, D. P. Hughes, and M. Salathé, “Using deep learning for image-based plant disease detection,” *Frontiers in Plant Science*, vol. 7, p. 1419, 2016.

[39] G. Wang, Y. Sun, and J. Wang, “Automatic image-based plant disease severity estimation using deep learning,” *Computational Intelligence and Neuroscience*, vol. 2017, Article ID 2917536, 8 pages, 2017.

[40] L. Zhang, L. Zhang, and B. Du, “Deep learning for remote sensing data: a technical tutorial on the state of the art,” *IEEE Geoscience and Remote Sensing Magazine*, vol. 4, no. 2, pp. 22–40, 2016.

[41] A. Romero, C. Gatta, and G. Camps-Valls, “Unsupervised deep feature extraction for remote sensing image classification,” *IEEE Transactions on Geoscience and Remote Sensing*, vol. 54, no. 3, pp. 1349–1362, 2016.

[42] K. Nogueira, O. A. Penatti, and J. A. dos Santos, “Towards better exploiting convolutional neural networks for remote sensing scene classification,” *Pattern Recognition*, vol. 61, pp. 539–556, 2017.

[43] J. Long, E. Shelhamer, and T. Darrell, “Fully convolutional networks for semantic segmentation,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3431–3440, Boston, MA, USA, June 2015.

[44] V. Badrinarayanan, A. Kendall, and R. Cipolla, “Segnet: a deep convolutional encoder-decoder architecture for image segmentation,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 12, pp. 2481–2495, 2017.

[45] S. Ji, W. Xu, M. Yang, and K. Yu, “3d convolutional neural networks for human action recognition,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 1, pp. 221–231, 2013.

[46] C. Dong, C. C. Loy, K. He, and X. Tang, “Image super-resolution using deep convolutional networks,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 38, no. 2, pp. 295–307, 2016.

[47] C. Farabet, C. Couprie, L. Najman, and Y. LeCun, “Learning hierarchical features for scene labeling,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 8, pp. 1915–1929, 2013.

[48] K. Zhang, Q. Liu, Y. Wu, and M.-H. Yang, “Robust visual tracking via convolutional networks without training,” *IEEE Transactions on Image Processing*, vol. 25, no. 4, pp. 1779–1792, 2016.

[49] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “Imagenet classification with deep convolutional neural networks,” in *Proceedings of Advances in Neural Information Processing Systems*, pp. 1097–1105, Lake Tahoe, NV, USA, December 2012.

[50] M. Matsugu, K. Mori, Y. Mitari, and Y. Kaneda, “Subject independent facial expression recognition with robust face detection using a convolutional neural network,” *Neural Networks*, vol. 16, no. 5-6, pp. 555–559, 2003.

[51] C. Szegedy, W. Liu, Y. Jia et al., “Going deeper with convolutions,” in *Proceedings of 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1–9, Boston, MA, USA, June 2015.

[52] O. Russakovsky, J. Deng, H. Su et al., “Imagenet large scale visual recognition challenge,” *International Journal of Computer Vision*, vol. 115, no. 3, pp. 211–252, 2015.

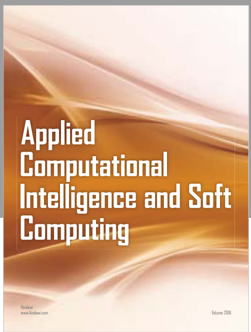
[53] V. Nair and G. E. Hinton, “Rectified linear units improve restricted Boltzmann machines,” in *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, pp. 807–814, Haifa, Israel, June 2010.

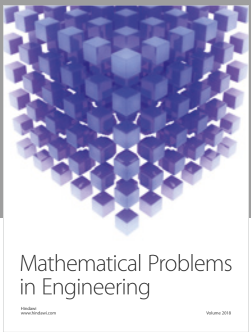
[54] S. Ioffe and C. Szegedy, “Batch normalization: accelerating deep network training by reducing internal covariate shift,” in *Proceedings of International Conference on Machine Learning*, pp. 448–456, Lille, France, July 2015.

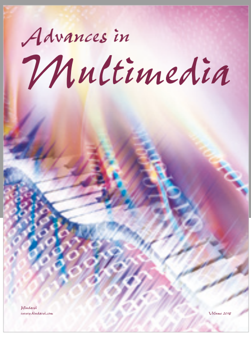
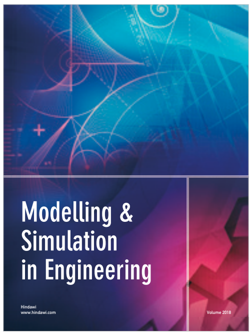
[55] U. Bhattacharya, M. Shridhar, S. K. Parui, P. Sen, and B. Chaudhuri, “Offline recognition of handwritten Bangla characters: an efficient two-stage approach,” *Pattern Analysis and Applications*, vol. 15, no. 4, pp. 445–458, 2012.

[56] N. Das, J. M. Reddy, R. Sarkar et al., “A statistical–topological feature combination for recognition of handwritten numerals,” *Applied Soft Computing*, vol. 12, no. 8, pp. 2486–2495, 2012.

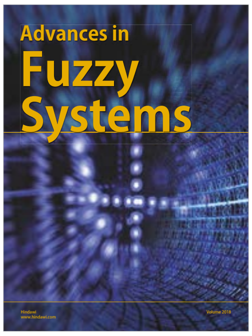
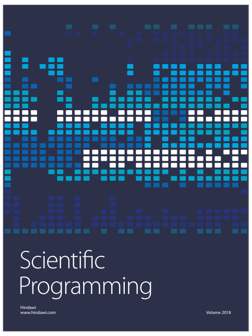
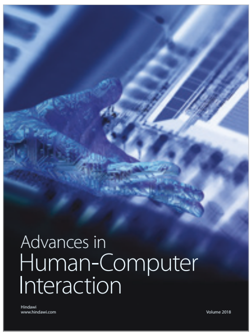
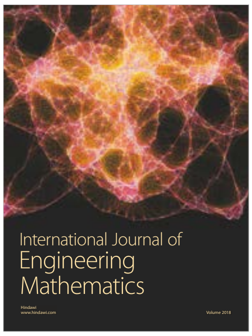
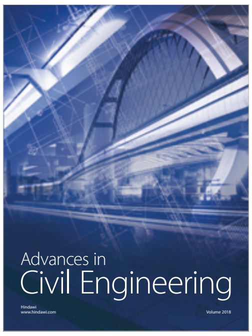
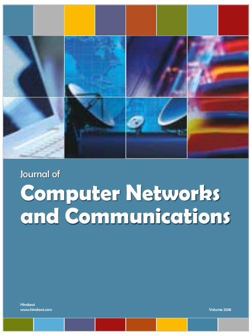
[57] M. Z. Alom, P. Sidike, T. M. Taha, and V. K. Asari, “Handwritten Bangla digit recognition using deep learning,” 2017, http://arxiv.org/abs/1705.02680.


Applied Computational Intelligence and Soft Computing

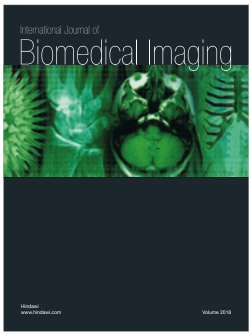
The Scientific World Journal

Mathematical Problems in Engineering

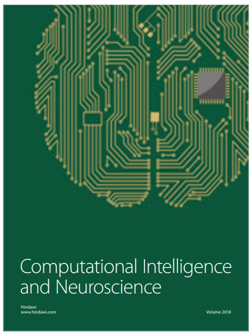
Journal of Engineering

Advances in Multimedia

Modelling & Simulation in Engineering

Advances in Artificial Intelligence

International Journal of Reconfigurable Computing

Advances in Fuzzy Systems

Scientific Programming

Advances in Human-Computer Interaction

International Journal of Engineering Mathematics

Advances in Civil Engineering

Journal of Computer Networks and Communications

Journal of Robotics

International Journal of Computer Games Technology

International Journal of Biomedical Imaging

Journal of Electrical and Computer Engineering

Computational Intelligence and Neuroscience