Convolutional neural networks performance comparison for handwritten Bengali numerals recognition

Md. Moklesur Rahman¹ · Md. Shafiqul Islam¹ · Roberto Sassi² · Md. Aktaruzzaman³

Received: 10 July 2019 / Accepted: 13 November 2019 / Published online: 23 November 2019
© Springer Nature Switzerland AG 2019

Abstract

Handwritten recognition has drawn profound attention since decades ago due to its numerous potential applications in real life. Research on unconstrained handwritten recognition in some languages has achieved attractive advancement, but it lags behind for Bengali even though it is the major language spoken by about 230 million people in the Indian subcontinent, and even the first and official language of Bangladesh. Recently, the use of convolutional neural network (CNN) has been reported with high accuracy in pattern recognition and computer vision problems. The main purpose of this study is to provide an architecture of a CNN to improve the accuracy of handwritten Bengali numerals recognition (HBNR) and compare its performance with the existing ones. We proposed a new CNN architecture, VGG-11M, which improves an existing one (VGG-11). The normalized and rescaled images of each numeral were augmented by different transformation operations to increase the training samples and to add diversity in the dataset. Then, the images were used to train the proposed VGG-11M model. The recognition accuracy of the developed system was tested on both training and test sets of three publicly available handwritten Bengali numerals database at different resolutions. Finally the performance of the model was compared with four other architectures (LeNet-5, ResNet-50, VGG-11, and VGG-16). The highest accuracy 99.80%, 99.66%, and 99.25% was obtained using the proposed architecture on the test set of ISI, CMATERDB, and NUMTADB dataset, respectively, at resolution 32×32 . The proposed VGG-11M outperformed the existing architectures of CNN on HBNR.

Keywords Bengali numerals · Handwritten recognition · Comparison · Deep learning

1 Introduction

Handwritten recognition has gained much attention to the researchers because of its numerous potential applications in real life. Some major applications include automatic bank cheque processing, form data entry, and postal sorting [1]. Bengal is a region of eastern South Asia, which comprises Bangladesh and the West Bengal of India. Bengali is the 7th most widely spoken language in the World and the mother language of Bangladesh. It is used as the official language of Bangladesh and several Indian states

including Assam, Jharkhand, Tripura, and West Bengal [2]. Research on handwritten numerals has made impressive progress in some languages such as Arabic, Chinese, and English [1, 3–5]. The automatic recognition of printed Bengali numerals is also very high; however, the progress of handwritten Bengali numeral recognition (HBNR) is far behind these languages [6–10].

Recognizing Bengali handwritten numerals is challenging as for Arabic numerals, because of their varied sizes and critical shapes. But more importantly, there are Bengali numerals whose shapes are very similar and this

✉ Md. Aktaruzzaman, aktaruzzaman@iu.ac.bd | ¹Department of Computer Science and Engineering, The People's University of Bangladesh, Dhaka 1207, Bangladesh. ²Dipartimento di Informatica, Università degli Studi di Milano, Via Celoria 18, 20133 Milan, Italy. ³Department of Computer Science and Engineering, Islamic University, Kushtia 7003, Bangladesh.

shape similarity makes the recognition more challenging. Therefore, efforts with new tools and methods for better HBNR are a timely demand.

Handwritten recognition of numerals is a classical problem of pattern recognition and machine learning, where proper choice of classification tool and datasets play an important role in this kind of problem. To get optimum accuracy of a classifier, it should be trained properly with sufficiently large dataset. There are very few handwritten Bengali numeral datasets available in the literature. Among them, the most commonly used datasets are CMATERdb3.1.1 [11] and ISI [12], consisting of 6000 and 23392 handwritten samples, respectively. On the other hand, MNIST [13], the largest dataset of Hindu–Arabic numerals (i.e., the so-called English numerals set) consists of 60,000 samples, and the best accuracy reported in the literature on this dataset is 99.79%. The number of samples of the Bengali datasets is small compared to the English dataset. In general, many samples have diversity which implies improved training of the classifier. The limitation of small datasets of Bengali handwritten numerals was recently resolved by the publication of a large dataset called NUMTADB by Alam et al. [14] consisting of 85596 samples. Training with larger dataset might not always means better training of a classifier. It depends on how much varieties in shape, size, resolution, writing styles, papers quality, etc have been added in the samples of the dataset. So, a comparison of the publicly available dataset is required to observe which of the dataset contains more versatility irrespective of dataset dimension.

The more important part of classification is the proper selection of classification methods. Every classification depends on the features extraction. In some classification tools, features are manually extracted and then fed to the classifier. On the other hand, some classification methods do not depend on the manually extraction of features. Indeed, they extracted features automatically by themselves. Among methods in the first category, a few of them have reported accuracies larger than 95%. Bhattacharya et al. [8] proposed handwritten Bengali numerals recognition using an artificial neural networks (ANNs). Some descriptive features like loops, junctions, etc. from the skeletal shape were extracted and fed to a multilayer perceptron (MLP) neural network to classify the numerals. They reported recognition accuracy of about 90% on the test dataset of 3440 samples in 2006. Pal et al. [9] proposed water reservoir overflow-based features, in addition to some topological and structural features of the numerals. The overall recognition accuracy, they reported on a test dataset of 12000 samples was about 92.8%. Later in 2007, Pal and his co-authors [15] further published another work on the recognition of handwritten Bengali numerals with five other major Indian scripts. In this study, they reported

the maximum recognition accuracy of 98.99% on a test set of 14650 samples using the quadratic classifiers based on 16-direction gradient histogram features using Robert masks.

In 2009, Bhattacharya et al. [12] proposed an improved version of their previous work [8]. This time, they proposed a multistage recognition of the numerals at three resolution levels: 16×16 , 32×32 , and 64×64 . There were three distinct MLPs for three resolution levels organized in a cascaded fashion in the first stage, and a single MLP in the second stage. The classification started with MPL at the coarser resolution level. The numeral rejected by the MPL at the coarser resolution was attempted to recognize by the MPL at the higher level of resolution. If the decision about the possible class of a numeral could not be reached even by the MPL at the final level of resolution, likelihood estimates, obtained from the classifiers in the first stage, were fed into the MPL at the second stage. Finally, the numeral was either recognized or rejected by this MLP according to a selected precision index. They reported recognition accuracies of 98.20% and 99.14%, respectively, for test and training sets. To the best of our knowledge, the highest accuracy 98.69% on the ISI dataset was reported by Liu et al. [16] using 8-direction gradient features and a MLP with one hidden layer of 100 nodes.

Although this traditional shallow neural network is computationally less expensive, the main challenge with this manual feature extraction technique is the extraction and selection of best representative features of a class. Recently, convolutional neural networks (CNNs) have been found more efficient than the existing non-deep learning-based machine learning methods for image classification [17]. They automatically provide some degree of translation invariance and do not depend on any feature extraction methods. By this time, CNNs have been recognized as the best classification method for handwritten (English or Hindu–Arabic) digits as claimed in the survey done by [18]. However, a few researchers [7, 10, 19, 20] have explored the power of CNNs for handwritten Bengali numerals recognition. One major requirement of using CNNs is that it requires huge number of samples for training; however, none of the existing works explored the power of CNNs on the recently published large NUMTADB dataset. A brief description of related works found in the literature is now presented.

CNNs were employed first by [21] for handwritten Bengali character recognition, in 2015. The experimental results showed that the CNNs outperformed the standard shallow learning methods with predefined features. The authors of [10] used autoencoders and deep CNNs for recognition of handwritten Bengali numerals. The deep network was trained with 19313 samples of the ISI training dataset and tested with images of the CMATERdb3.1.1

dataset, and they leading to an accuracy of 99.50%. Akhand et al. employed CNNs for handwritten Bengali isolated numerals recognition on a dataset of 17,000 numeral images. They used 13,000 samples for training the CNNs and the remaining 4000 to test the performance of the model. They reported recognition accuracies of about 99.40% and 97.93%, respectively, on the train and test sets. Later, Akhand and the coauthors of [20] have proposed a different CNNs-based method for HBNR. Three datasets were used for the experiments. Each CNN was trained with a different training set prepared from the samples of handwritten images, and the final decision was made by combining the decisions of the CNNs. One CNN was trained with the ordinary images of the handwritten numerals. A simple rotation-based approach was used to generate training sets for other CNNs. A set of 18000 numeral images, out of the 19392 of the ISI dataset, was considered for training one CNN. The performance of the developed system was tested on the separated test set, of 4000 samples, of the ISI database. Their method with multiple CNNs reported accuracies of 98.80% and 99.51% on the test and training sets, respectively, and it is the best performance ever found in the literature on HBNR. It is definitely a great achievement that seems closer to human perception, on this test set of 4000 samples only.

Although a number of works have been done in the literature using both regular and deep machine learning architectures [12, 15, 20, 21], they suffer from one or more of the following vital limitations: (1) low recognition accuracy to adopt in practical, (2) private dataset with small number of samples, and (3) not sufficiently good performance. The performance of a recognition system may vary for different test sets. To test how much it generalizes to completely unknown samples, we observed its performance on a large set of data. There are different architectures of CNN from very simple to complex, and the accuracy of a recognition system can depend on which architecture of CNN is used for what kind of problem. The applications of CNNs with same architecture on different problems may provide different performances, and a simple modification in the architecture of a CNN can change its performance. CNNs are becoming more popular in the computer vision and pattern recognition fields, and several attempts have been made to improve the original architecture [22] with the aim of achieving better accuracy. For instance, the best model [23] submitted to the ImageNet Large-Scale Visual Recognition Challenge (ILSVRC-2013) for image recognition proposed the use of smaller receptive window size with smaller stride at the first convolutional layer. Another improvement [24] was done by densely training the network over the whole image. Other line of improvements in the architectural design of CNN were done by Simonyan and Zisserman

[25], who increased the depth of the architecture by increasing the number of convolutional layers and fixing other parameters. The changes in the architectural design improve the performance of the recognition model. VGG is an innovative object recognition model and has been placed first and second in the open ImageNet competitions held in 2013 and 2014, respectively. There are several architectures of VGG model: from very simple VGG-8 to very complex VGG-19 including VGG-11. Inspired by the works of [22, 24, 25], we have proposed a model (VGG-11M) by modifying VGG-11 (as a model with moderately low complexity) with the aim of improving the accuracy of the existing recognition systems for handwritten Bengali numerals recognition.

The proper selection of CNN architectures with a sufficient amount of diversity in the training set with enough training samples can improve the accuracy of the HBNR system to the level of human perception. So, the objectives of this study are (1) to study and compare the performance of some prominent models of CNN (e.g., VGG) and its architectural modification for recognition of handwritten Bengali numerals, (2) to compare the performance of the newly proposed model with the already published works represent the highest accuracy in the literature, (3) to test the generality of the system so that it stands robust with completely unknown samples of numerals, (4) to observe how much the training is affected by the image resolution, and finally (5) to compare the existing datasets with respect to the sufficient training of an artificial neural network.

2 Dataset

In this study, three datasets of handwritten Bengali numerals have been considered: the handwritten ISI Bengali numeral database [26], the CMATERdb3.1.1 (CMATERDB) [11] dataset and the NUMTADB dataset [14]. The ISI train and test dataset consists of 23, 392 and 4000 samples, respectively, for training and testing. The training and test sets are completely different. The CMATERDB dataset consists of 6000 handwritten Bengali numerals. Each numeral has 600 images of resolution 32×32 pixels. The size of the dataset is small with quite high variation [7]. Since there are no separate training and test sets, the dataset was divided randomly into training (5000) and test set (1000, 100 samples of each class of numeral). The very recently introduced NUMTADB [14] dataset consists of 85596 samples of handwritten Bengali numerals, which is actually an assembly of six different datasets. A separate test set is not available, so we randomly prepared a training and test sets by randomly dividing the dataset with a ratio of 80%:20% (train:test). Moreover,

10% samples from the training set, selected randomly of each dataset, were used to construct the validation set for the corresponding dataset. Some samples of the handwritten Bengali numerals are shown in Fig. 1.

3 Methods

Deep learning-based methods often provide better performance as compared to shallow learning based methods (e.g., classical dense artificial neural network) [27]. So, we have considered in this study deep learning-based methods for handwritten Bengali numerals recognition. Each numeral from the training set was preprocessed and rescaled first, then fed to a deep convolutional neural network for training. After training, we evaluated both the individual and cross-validation performances of the proposed VGG-11M model on the three datasets at three resolutions: 16×16 , 32×32 , and 64×64 . There are many different CNN models from simple to complex. The recognition performance can depend on the selection of the proper model. Here, we have considered five models: (1) LeNet-5, (2) ResNet-50, (3) VGG-16, (4) VGG-11, and (5) the proposed VGG-11M architectures. The basic structure of a CNN consists of two main modules which are responsible for features extraction and classification. The features extraction module is actually a collection of a number of layers, where operations are performed at each layer. Finally, the set of extracted features are passed to a fully connected neural network (FC). Some more details about deep learning are presented in section 3.1.

Arabic digits (numerals)	Bengali numerals (in word)	Bengali numerals	Some samples of handwritten Bengali numerals									
0	Shyanya	০	০	০	০	০	০	০	০	০	০	০
1	Ek	১	১	১	১	১	১	১	১	১	১	১
2	Dui	২	২	২	২	২	২	২	২	২	২	২
3	Tin	৩	৩	৩	৩	৩	৩	৩	৩	৩	৩	৩
4	Chaar	৪	৪	৪	৪	৪	৪	৪	৪	৪	৪	৪
5	Panch	৫	৫	৫	৫	৫	৫	৫	৫	৫	৫	৫
6	Chhoy	৬	৬	৬	৬	৬	৬	৬	৬	৬	৬	৬
7	Shat	৭	৭	৭	৭	৭	৭	৭	৭	৭	৭	৭
8	Aat	৮	৮	৮	৮	৮	৮	৮	৮	৮	৮	৮
9	Noi	৯	৯	৯	৯	৯	৯	৯	৯	৯	৯	৯

Fig. 1 Few samples of Bengali numerals. The left most column shows the symbols of Arabic numerals. The Bengali numeral corresponding to each Arabic numeral is shown, respectively, in both word and symbolic forms in the second and third columns. A set of 10 randomly selected samples of each handwritten Bengali numerals has been shown on the right most column

3.1 Deep learning background

Deep learning is a subset of machine learning. Usually, when we use the term deep learning, we are referring to deep artificial neural networks. The term deep refers to the number of layers in an artificial neural network. A shallow neural network has one (or very few) so-called hidden layer, while simplifying, a deep network has more than one hidden layers. In traditional shallow machine learning, a set of relevant features are first extracted manually from the input object and then applied to the input of the classifier. Deep neural networks are actually a set of algorithms that automatically extract many features by themselves from the input object and performs classification (or recognition). Convolutional neural networks (CNNs) are the most commonly used deep neural networks for classification.

CNNs automatically extract features of an object using 10 to 100s hidden layers. The complexity of learned features increases with increasing hidden layer, e.g., the first hidden layer could learn a shape, and the more complex shape of an object is learned at the last hidden layer. This automatic features extraction might lead to high accuracy for computer vision and object recognition tasks using deep learning models. A brief description of the architecture of some deep learning models such as LeNet-5, ResNet-50, and VGG is given here.

3.1.1 LeNet-5

LeNet-5 is a classic CNN model which was proposed by the Yann LeCun, Leon Bottou, Yosuha Bengio, and Patrick Haffner in 1998 for optical character recognition [28]. The LeNet-5 is structured on seven layers. In its architecture, there are two sets of pooling and convolutional layers followed by a flattening convolutional layer and two fully connected layers. Finally, there is a softmax classifier at the end of its architecture.

3.1.2 Residual network-50

The residual network (ResNet) model was introduced by Microsoft [29] and won the ILSVRC (ImageNet Large Scale Visual Recognition Challenge) award in 2014 [30]. The key concept of ResNet was to increase the number of layers of LeNet and introducing a residual unit (a collection of a number of consecutive layers) with an identity layer. In this model, the input of each residual unit (RU) is merged with the output of the unit and acts as an input to the next RU. Thus, in ResNet unlike LeNet and VGG, the input of every RU, in addition to its output, is passed to the input of the next RU. ResNet-50 is a 50 layer convolutional neural network and can classify images into 1000 object categories,

such as mouse, keyboard, pencil, and animals of different classes.

3.1.3 VGG

Simonyan and Zisserman [25] in 2014 introduced a model of CNN to investigate the effect of convolutional depth on its accuracy in large-scale image recognition problem and named it VGG according to the name of their research group: Visual Geometry Group. Two special architectures of VGG, VGG-11 [31] and VGG-16 [32] were used for image segmentation and handwritten Bengali characters recognition, respectively. The architecture of VGG-11 consists of 8 convolutional layers and 3 fully connected layers followed by a single *Softmax* layer, and the architecture of VGG-16 consists of 13 convolutional layers and 3 fully connected layers followed by a single *Softmax* layer.

4 Proposed method

The major steps of the proposed method for handwritten Bengali numerals recognition have been summarized by the block diagram in Fig. 2.

4.1 Preprocessing

The images of raw numerals are of 256 levels grayscale at different resolutions. They were normalized in the range [0, 1]. Handwritten numerals have no specific resolution and patterns are typically transformed to the same dimension (or resolution). In this study, we have considered three fixed resolutions: 16×16 , 32×32 , and 64×64 . After pre-processing, each numeral is fed to the CNN model.

Fig. 2 Overall procedures processed in handwritten Bengali numeral recognition

4.2 VGG-11M model

We have proposed a modified version of the VGG-11 [25, 31] convolutional neural network model. In this study, this modified version of VGG-11 model has been denoted by VGG-11M. In VGG-11M, we have performed optimization of some functions of VGG-11 to minimize over-fitting problems, and hence to improve the classification accuracy. The structure of the proposed CNN model for Bengali handwritten numerals recognition is described in Fig. 3. A brief description of every layer of the model is given here.

4.2.1 Convolution layer

Convolution layer by definition serves as the primary component of a CNN. It is the first layer of CNN that is responsible for extracting features from the input data using a convolution filter or kernel. Kernels (or filters) actually extract the learnable parameters (the weights) from the input image of size $M \times N$ through forward and backward propagation. This operation is accomplished by sliding the filter of size 3×3 over the input image matrix at stride 1. At every move, we perform element-wise matrix multiplication and sum the results. This sum results in a feature map.

Fig. 3 Architecture of the VGG-11M model for handwritten Bengali numerals recognition system. In the diagram, the terms Conv, BN, GAP, FC stand for convolutional layer, batch normalization, Rectified Linear Unit activation function, global average pooling layer, and fully connected layers, respectively

We choose a kernel size for the convolution layers of 3×3 , as proposed by other similar studies [22, 29, 33].

The performance of CNN depends on the number of kernels used in the convolutional layers. There is no universally accepted rule for selecting the number of kernels in a convolution layer. In the literature [22], the author has used a different number of kernels in different convolution layers starting at 64 to 512. In this study, we tried the number of kernels starting at 64 to 512 with step size 32, and we selected on the validation set the combination of kernels which maximize the accuracy.

Stride defines the number of pixels shifts over the input matrix, and we selected stride of length one to move the filters one pixel at a time. Batch normalization layer [28] is followed by every convolution layer, which enables greatly accelerating the learning process.

4.2.2 Zero-padding layer

The term zero padding refers to the process of adding extra rows or columns of zeros at four sides of the actual image. Zero padding (P) refers to adding P rows or columns of zeros at each side. This is an optional layer of the convolutional neural network. When the input image resolution is too small ($\leq 16 \times 16$), the border information (pixels) gets more importance than the internal pixels. Thus, the convolution operation seems biased to the border pixels. Another problem with the images of small resolution is that when more convolution layers are added to increase the depth of the CNN, the image dimension reduces to zero due to convolution (pooling) layer, so further operation cannot be performed. For example, when a filter of size $f \times f$ is applied to an image of size $n \times n$, with zero padding (p) and stride (s), then after convolution, the input dimension reduces to $m \times m$, where m is given by

$$\frac{n + 2 \times f - f}{s} + 1. \quad (1)$$

So, when the value of $n + 2 \times p$ becomes $\leq f$, the dimension of resultant images becomes 0, and hence cannot further proceed. To tackle this problems with small resolution images, we added zero-padding layers in our proposed model.

4.3 Activation function

Activation functions play a key role in every CNN architecture. Whether a neuron will be activated or not is determined by the activation function that verifies if the information received by a neuron is "relevant." The relevant information is then passed to the next level for further operation. There are two types of activation functions:

linear and nonlinear. The nonlinear functions are mostly used in the artificial neural networks. The most commonly used activation functions used in CNN models are rectified linear units (ReLU), leakage rectified linear units (LReLU), and exponential linear units (ELU) [34]. In our proposed model, we used ReLU activation function, which is nonlinear in nature. The output of this activation function is a continuous variable. It returns 0, if the input is a negative number, otherwise the output is equal to the input. Mathematically, The ReLU function can be expressed as:

$$\text{ReLU}(x) = \max(0, x), \quad (2)$$

where x denotes the input to a neuron. Another important reason of selecting ReLU is that it tends to be several times faster [35] than their equivalents (sigmoid, tanh, etc.) in training CNN. ReLU assigns all negative inputs to zero, which means ignoring a large number of nodes that will never be considered in the future training.

4.3.1 Regularization

Regularization, also called normalization, is a supplementary technique that tries to more generalize a model, i.e., produce better results on the previously unseen data [36]. Typical normalization methods are $L1$ (expressed as the sum of the absolute values of the weights) and $L2$ (expressed as the sum of the square of the weights). Generally, in deep learning, dropout and batch normalization are applied, as the network gets deeper a small change in parameter in a previous layer can have a large effect on the input distribution of the next layers. This phenomenon is called internal covariate shift, which can be reduced by applying batch normalization. The operation, batch normalization, is applied after the convolution layer and before the activation function. In the proposed mode, input batches were standard normalized by subtracting the mean of the batches and divided by the standard deviation. The normalized batch was then scaled and shifted.

In a fully connected layer, neurons become codependent among each other. This dependency curbs the individual power of each neuron, and hence leads to over-fitting of training data. The over-fitting can be avoided by reducing the interdependency between layers, called dropout. This is obtained by dropping out individual neurons of the network with probability $1 - p$. There is no standard rule about which value should be chosen for p . Different researchers [37–39] have selected different values of p for this purpose, e.g., in [38], they have used $p = 0.5$. In our study, we used dropout three times in our model with $p = (0.2, 0.25, 0.4)$. We placed it after each activation function. These values were selected based on providing maximum accuracy on a validation set.

4.3.2 Pooling layer

A pooling layer is a non-trainable layer of CNN which is inserted in between successive convolutional layers of a CNN. The purpose of using pooling layer is to reduce the number of parameters, and hence to limit the computational cost in the network. This layer spatially resizes the input by working independently on every slice of the input. The most commonly used form of a pooling layer is filters of size 2×2 and stride 2. In every step, it downsamples every input depth slice by 2, along with both height and width. There are several types of pooling operations: max-pooling, average pooling, min-pooling, etc. In our proposed model, we have used max-pooling which was computed over every 2×2 small regions in some depth slice. The output dimension of the max-pooling operation can be calculated using the following mathematical expression:

$$n_{\text{out}} = \text{floor} \left\{ \frac{n_{\text{in}} - f}{s} \right\} + 1, \quad (3)$$

where n_{in} , f and s denote the dimension of the input image, the filter size, and the stride size, respectively.

4.3.3 Global average pooling (GAP) layer

Global average pooling (GAP) layers dramatically reduce the total number of parameters in the model. They carry out a more extreme dimensionality reduction, where a tensor with dimensions of size height $\times$ width $\times$ depth is reduced to $1 \times 1 \times \text{depth}$, i.e., the GAP layer maps each feature matrix of dimension height $\times$ Width to a single number by simply taking the average of all its values.

4.3.4 Fully connected layer

The convolution and pooling layers discussed so far contain information regarding local features in the input image such as edges, blobs and shapes. These feature matrices are squashed into a vector which is handed over to a fully connected (FC) layer. In FC layers, neurons in one layer are connected to every neurons in another layer. Its classification performance depends on the features extracted from the previous layers, which contains an activation function (e.g., *Softmax*, *Sigmoid*, *ReLU*) like a traditional shallow neural network. The performance of the classifier can be varied due to the various activation functions used in this layer. In this study, we used a *Softmax* activation function, which has been applied in most of the research works for image recognition.

4.3.5 Output layer

The output of a *Softmax* activation function is basically the normalized exponential probability of class observations. It is simply the exponential of each input divided by the sum of the exponential. In our study, it results in a vector of length 10, where each scalar is the probability to belong to one of the 10 category of numerals. Note that the sum of the output is 1. The probability of each class in the *Softmax* layer can be expressed as:

$$\sigma(X)_j = \frac{e^{X_j}}{\sum_{k=1}^k e^{X_k}}, \quad (4)$$

for $j = 1, 2, 3, \dots, k$, where k is the number of classes and X_j (for each value of j) are the inputs from the previous fully connected layer applied to each node of the *Softmax* layer.

4.3.6 Cost and optimizer function

The cost function of a neural network quantifies the error of the network's performance. There are different functions of estimating the performance error of a neural network; a cross-entropy function is typically used in deep learning neural network. The purpose of optimization algorithm is to minimize (or maximize) the error function. It is simply a mathematical function that depends on the model's internal learnable parameters which are used in computing the target values (Y) from the set of predictors (X) used in the model. There are different cost optimization algorithms such as *Adagrad* [40], *Adadelta* [41], and *Adam* [42]. In this work, we have used a gradient descent-based optimizer to minimize the cost function:

$$c = \frac{1}{m} \sum [y \ln a + (1 - y) \ln(1 - a)] \quad (5)$$

where m is the size of training data, y is an expected value, and a is the actual value from the output layer. In our tests, the final performance difference between the above functions was not large. The optimal point is reached earlier if *Adam* was used as optimizer function. Therefore, we used *Adam* optimizer [42] with learning rate 0.001. The number of training epochs was a hyperparameter, and we train our model for up to 200 epochs (batch size: 128, steps per epoch: 64). We included early stopping, to define that we wanted to monitor the validation (test) loss at each epoch. After the test loss has not improved for thirty epochs, training was interrupted. The learning rate was updated to 75% of its value if our model validation accuracy did not improve for six consecutive epochs. The weights of the network were randomly initialized with small numbers from a normal distribution, and we used a weight decay of 1×10^{-6} .

5 Results

We studied four different architectures (LeNet-5, ResNet-50, VGG-11, and VGG-16) and our proposed VGG-11M model for handwritten Bengali numerals recognition at three different resolutions: 16×16 , 32×32 , and 64×64 . In addition to batch normalization and zero-padding layers, the number of filters in the convolutional layers and neurons in the fully connected layers have been reduced in the proposed VGG-11M than in VGG-11. This changes in VGG-11 model result in the drastic reduction of total number of parameters from 28, 148, 235 (for VGG-11) to 7, 717, 258 (for VGG-11M). However, the VGG-11M model among all five architectures of CNN was found more efficient, and it provides the maximum accuracy (99.80%) for the ISI dataset at resolution of 32×32 , which is the best performance on Bengali handwritten numeral recognition system till date. Moreover, it also provides maximum accuracy for the other two datasets (CMATERDB and NUMTADB). The performance of the VGG-11M model for the ISI dataset is shown in Fig. 4.

The training and validation accuracies are 100% and 99.90%, respectively. The more detail results about the selection of CNN architectures, the dataset, and resolutions are described in Tables 1, 2, and 3. The accuracy of the recognition system was not affected so much for resolutions of size 32×32 or more. The accuracy of the proposed model was always larger than 99%, irrespective of the dataset and resolutions ($\leq 32 \times 32$).

The best accuracy 99.80% was achieved for ISI (separate train and test sets) dataset, and the worst accuracy (99.25%) was obtained for NUMTADB (separate train and test sets) dataset, at resolution 32×32 . The authors of [22] reported that ResNet performed better than VGG-11, which is also proved in our study. However, the proposed VGG-11M provided better accuracy than all models including ResNet (please see Tables 1, 2, 3).

Fig. 4 The performance curve (training and validation accuracies) of the VGG-11M architecture on the ISI dataset for each iteration

Table 1 Results on the ISI dataset for different models at three different resolutions

Dataset	Model	Resolution	Accuracy (%)
ISI	LeNet-5	16×16	96.9
		32×32	98.5
		64×64	98.6
	ResNet-50	16×16	99.65
		32×32	99.13
		64×64	99.63
	VGG-11	16×16	97.38
		32×32	98.25
		64×64	99.40
	VGG-16	32×32	99.02
		64×64	99.15
	VGG-11M	16×16	98.98
		32×32	99.80
		64×64	99.43

The cross-validation (i.e., when training is done using ISI training set and evaluated with the CMATERDB dataset, and when training is done using the CMATERDB dataset and evaluated with the ISI test dataset) accuracies of the proposed VGG-11M model at resolution 32×32 were 99.72% and 94.75%, respectively. The effect of data augmentation to increase the number of samples and variability (or diversity) in the train set was also investigated in our study. The adoption of data augmentation on the ISI training set increased the recognition accuracy from 99.53 to 99.80%. Thus, it slightly improves the result obtained without augmentation. The accuracy (99.55%) of the proposed method exceeds the best accuracy (99.35%) reported in the literature

Table 2 Results on the CMATERDB dataset for different resolutions and models

Dataset	Model	Resolution	Accuracy (%)
CMATERDB	LeNet-5	16×16	98.05
		32×32	99.16
		64×64	98.72
	ResNet-50	16×16	98.78
		32×32	99.22
		64×64	98.72
	VGG-11	16×16	98.28
		32×32	98.56
		64×64	99.00
	VGG-16	32×32	99.16
		64×64	98.80
	VGG-11M	16×16	98.94
		32×32	99.66
		64×64	99.06

Table 3 Results on the NUMTADB dataset for different resolutions and models

Dataset	Model	Resolution	Accuracy (%)
NUMTADB	LeNet-5	16 × 16	93.03
		32 × 32	98.04
		64 × 64	97.24
	ResNet-50	16 × 16	94.25
		32 × 32	98.3
		64 × 64	98.85
	VGG-11	16 × 16	91.45
		32 × 32	98.00
		64 × 64	99.06
	VGG-16	32 × 32	98.87
		64 × 64	99.18
	VGG-11M	16 × 16	95.3
		32 × 32	99.25
		64 × 64	99.40

True level of numeral	0	399	0	0	0	0	0	0	0	0
	1	0	400	0	0	0	0	0	0	0
	2	0	0	399	0	0	0	0	0	2
	3	1	0	0	400	0	0	0	0	0
	4	0	0	0	0	400	0	0	0	0
	5	0	0	0	0	0	400	0	0	1
	6	0	0	0	1	0	0	400	0	0
	7	0	0	0	0	0	0	0	400	0
	8	0	0	0	0	0	0	0	0	400
	9	0	0	0	0	0	0	0	0	397
Predicated level of numeral										
	0	1	2	3	4	5	6	7	8	9

Fig. 5 The confusion matrix represents the classification performance on test set of the ISI database. Columns correspond to true label, and rows correspond to the predicated label

at resolution 29×29 on ISI dataset [43]. Finally, the confusion matrix of the VGG-11M model on the test set (each class of numeral contains 400 samples) of the ISI database at resolution 32×32 is shown in Fig. 5.

It is observed from Fig. 5 that the model predicted numerals two, five, six, seven, and eight with 100% accuracy. The lowest accuracy (99.25%) was achieved for numerals one and nine, whose patterns are very similar (some numerals of these classes are very challenging to recognize even for humans).

6 Discussion

In this study, we have proposed a model of convolutional neural network for Bengali handwritten numerals recognition. This model of CNN has been used extensively

in some image recognition problems since it was introduced in 2014 by [25]. The authors of [32] have then explored its performance for Bengali handwritten character recognition. In our model, we have incorporated some extra layers, like batch normalization, zero padding, dropout and a favorable selection of the number of convolution layers, max-pooling layer, and the number of filters that boosted the classification performance by a significant margin. The application of more convolutional operations on low-dimension images (e.g., 16×16 or less) may become Critical. For this reason, the image was kept sufficiently large and zero padding was used. Dropout prevents model overfitting by forcing the model to learn more robust features that are useful in aggregation with many different random subsets of the other neurons.

Our results show that the recognition accuracy is affected by the resolution (if it is less than 32×32) of the resampled image. This is because the quality of the resampled image is degraded with reducing the resolution. However, the image resolution, when reaches to a certain level (32×32 or more), does not significantly affect the classifier performance. The classifier showed the best performance (99.80%) when it was trained by the ISI handwritten numeral dataset.

The cross-validation results reveal the generalizability of the classifier. It is clear from the result that the classifier becomes more robust when it is trained by the ISI dataset. The individual accuracy (i.e., when train with training set and test with the test set of the same database) is always better than that obtained with cross-validation (i.e., when train and test set comes from the different datasets). This can be due to the fact that the train and test sets are different, and they may still share some similarity (image format, quality, etc.). When our model is trained on the ISI and tested on CMATERDB, the accuracy rate achieved is 99.72%, which is better than the previously published results on this dataset. On the other hand, a moderately good result (94.75%) was obtained when the CMATERDB dataset was used for training and tested with the ISI test dataset. This might be due to the fact that the ISI dataset is larger than the CMATERDB dataset, so the more diversity of samples made the training more perfect. Interestingly, although the dimension of NUMTADB is about three times more than the ISI dataset and about twelve times more than the CMATERDB dataset, the performance achieved (98.3%) is lower than when trained with ISI. This result meant that more diversity or variability exists in the ISI dataset than into the NUMTADB dataset, even though its larger dimension.

The accuracy of a classifier can be affected by both the image resolution and the sample used to test the classifier.

The comparison of the proposed method and the method [43] (best accuracy 99.35%) on the same test set showed that our method is about 0.45% more accurate greater than the previous best method. The development of the VGG-11M model opens the doors for the future research on Bengali handwritten character recognition and related problems. The recognition error rate, even though very close to 0%, is mainly due to the misclassification between numerals nine and one, which are somewhat difficult to recognize even for a human. This is a possible area for improvement by developing a robust CNNs model.

7 Conclusion

A handwritten Bengali numeral recognition system with the highest accuracy has been proposed. In this paper, we studied four architectures of convolutional neural network on three publicly available datasets. One of the datasets is very large in dimension, and no work can still be found on it in the literature. The validity of the new dataset has been compared with the other two datasets commonly used for this purpose. How much the resolution affects the classification performance was also studied. The comparison of the proposed VGG-11M model with the image augmentation by blocky artifact and deep convolutional neural network [43] model (i.e., the method with the highest accuracy) was in favor of the new method. The ISI dataset proved more robust even with small sets of a sample. Thus, the dataset dimension does not matter always if the training set contains a sufficient amount of diversity or variability.

Compliance with ethical standards

Conflict of interest The authors declare that they have no conflict of interest.

References

- Plamondon R, Srihari SN (2000) On-line and off-line handwritten recognition: a comprehensive survey. *IEEE Trans PAMI* 22(1):62–84
- Chatterji SK, Wagner R (2015) The origin and development of the Bengali language. *Indogermanische Forschungen* 47(1):370–380
- Azmi AN, Nasien D, Shamsuddin SM (2013) A review on handwritten character and numeral recognition for Roman, Arabic, Chinese and Indian scripts. [arXiv:1308.4902](https://arxiv.org/abs/1308.4902)
- Abdleazeem S, El-Sherif E (2008) Arabic handwritten digit recognition. *Int J Doc Anal Recognit* 11(3):127–141
- Wang JJ, Hu S, Zhan X, Yu Q, Liu Z, Chen TP, Yin Y, Hosaka S, Liu Y (2018) Handwritten-digit recognition by hybrid convolutional neural network based on HfO₂ memristive spiking-neuron. *Sci Rep* 8(1):12546
- Aktaruzzaman M, Khan MF, Ambia A (2013) A new technique for segmentation of handwritten numerical strings of Bangla Languagesuk. *Int J Inf Technol Comput Sci* 65(5):38–43
- Alom MZ, Sidike P, Taha TM, Asari VK (2017) Handwritten Bangla digit recognition using deep learning. *CoRR*, [arXiv:1705.02680](https://arxiv.org/abs/1705.02680)
- Bhattacharya U, Das TK, Datta A, Parui SK, Chaudhuri BB (2002) Recognition of Handprinted Bangla numerals using neural network models. In: *Proceedings of AFSS international conference on fuzzy systems. Advances in Soft Computing*, Calcutta, pp 228–235
- Pal U, Belaid A, Chaudhuri BB (2006) A system for Bangla handwritten numeral recognition. *IETE J RES* 3:444–457
- Shopon M, Mohammed N, Abedin MA (2016) Bangla handwritten digit recognition using autoencoder and deep convolutional neural network. In: *International workshop on computational intelligence (IWCI)*, pp 64–68
- Basu S, Sarkar R, Das N, Kundu M, Nasipuri M, Basu DK (2005) Handwritten Bangla digit recognition using classifier combination through DS technique. In: *International conference on pattern recognition and machine intelligence*, pp 236–241
- Bhattacharya U, Chaudhuri BB (2009) Handwritten numeral databases of Indian scripts and multistage recognition of mixed numerals. *IEEE Trans Pattern Anal Mach Intell* 31(3):444–457
- Deng Li (2012) The-MNIST-database-of-handwritten-digit-images-for-machine-learning-research. *IEEE Signal Process Mag* 29(6):141–142
- Alam S, Reasat T, Doha RM, Humayun AI (2018) NumtaDB-Assembled Bengali handwritten digits. *CoRR*, [arXiv:1806.02452](https://arxiv.org/abs/1806.02452)
- Pal U, Sharma N, Wakabayashi T, Kimura F (2007) Handwritten numeral recognition of six popular Indian scripts. In: *Ninth international conference on document analysis and recognition*, vol 2, pp 749–753
- Liu C, Suen CY (2009) A new benchmark on the recognition of handwritten Bangla and Farsi numeral characters. *Patt Recognit* 42(12):3287–3295
- Sharif S MA, Mohammed N, Mansoor N, Momen S (2016) A hybrid deep model with HOG features for Bangla handwritten numeral classification. In: *9th international conference on electrical and computer engineering (ICECE)*, BUET, Dhaka, Bangladesh, pp 463–466
- Kamavisdar P, Saluja S, Agrawal S (2013) A survey on image classification approaches and techniques. *Int J Adv Res Comput Commun Eng* 2(1):1005–1009
- Akhand MAH, Rahman MM, Shill PC, Islam S, Rahman MMH (2015) Bangla handwritten numeral recognition using convolutional neural network. In: *International conference on electrical engineering and information communication technology (ICEEICT)* May 21–23, Dhaka, Bangladesh, pp 1–5
- Akhand MAH, Ahmed M, Rahman MMH (2016) Multiple convolutional neural network training for Bangla handwritten numeral recognition. In: *International conference on computer and communication engineering (ICCCE)*, October, 28–30, Durrës, Albania, pp 311–315
- Rahman MM, Akhand MAH, Islam S, Shill PC, Rahman MMH et al (2015) Bangla handwritten character recognition using convolutional neural network. *Int J Image Graph Signal Process* 7(8):42–49
- Krizhevsky A, Sutskever I, Hinton GE (2012) Imagenet classification with deep convolutional neural networks. In: *Advances in neural information processing systems*, pp 1097–1105
- Zeiler MD, Fergus R (2013) Visualizing and understanding convolutional networks. *CoRR*, [arXiv:1311.2901](https://arxiv.org/abs/1311.2901)
- Sermanet P, Eigen D, Zhang X, Mathieu M, Fergus R, LeCun Y (2014) Overfeat: integrated recognition, localization and

- detection using convolutional networks. In: Proceedings of the international conference on learning representations, May, 07–09, San Diego, USA, pp 1–14. [arXiv:1312.6229](#)
25. Simonyan K, Zisserman A (2014) Very deep convolutional networks for large-scale image recognition. In: International conference on learning representations, May, 07–09, San Diego, USA, pp 1–14. [arXiv:1409.1556](#)
 26. Chaudhuri BB (2006) A complete handwritten numeral database of Bangla—A Major Indic Script. In: 10th international workshop on frontiers in handwriting recognition, October, 23–26, Atlanta Congress Center, France, La Baule, France, 379–38423
 27. Mhaskar H, Liao Q, Poggio T (2017) When and why are deep networks better than shallow ones. In: Proceedings of the thirty-first AAAI conference on artificial intelligence, February, 4–9, San Francisco, California, USA, pp 2343–2349
 28. LeCun Y, Bottou L, Bengio Y, Haffner P (1998) Gradient-based learning applied to document recognition. *Proc IEEE* 86(11):2278–2324
 29. He K, Zhang X, Ren S, Sun J (2016) Deep residual learning for image recognition. In: IEEE conference on computer vision and pattern recognition, June, 27–30, Las Vegas, Nevada, USA, pp 770–778
 30. Russakovsky O, Deng J, Su H, Krause J, Satheesh S, Ma S, Huang Z, Karpathy A, Khosla A, Bernstein M, Berg AC, Fei-Fei L (2015) ImageNet large scale visual recognition challenge. *Int J Comput Vis* 115(3):211–252
 31. Iglovikov V, Shvets AA (2018) Terausnet: U-net with VGG11 encoder pre-trained on imagenet for image segmentation. [arXiv:1801.05746](#)
 32. Alom MZ, Sidike P, Hasan M, Taha TM, Asari VK (2018) Handwritten Bangla character recognition using the state-of-the-art deep convolutional neural networks. In: Computational intelligence and neuroscience. *CoRR*, [arXiv:1712.09872](#)
 33. Huang G, Liu Z, Maaten VDL, Weinberger KQ (2017) Densely connected convolutional networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 4700–4708
 34. Clevert D, Unterthiner T, Hochreiter S (2016) Fast and accurate deep network learning by exponential linear units (ELUs). In: Proceedings of international conference on learning representations, May, 02–04, San Juan, Puerto Rico. [arXiv:1511.07289](#)
 35. Weng T, Zhang H, Chen H, Song Z, Hsieh C, Boning DS, Dhillon IS, Daniel L (2018) Towards fast computation of certified robustness for ReLU networks. In: Proceedings of thirty fifth international conference on machine learning, July, 10–15, Stockholm, Sweden. [arXiv:1804.09699v4](#)
 36. Kuckacka J, Golkov V, Cremers D (2017) Regularization for deep learning: a taxonomy. *CoRR*, [arXiv:1710.10686](#)
 37. Ko B, Kim H, Oh K, Choi H (2017) Controlled dropout: a different approach to using dropout on deep neural network. In: IEEE international conference on big data and smart computing, February, 13–16, Jeju Island, Korea, pp 358–362
 38. Jun TJ, Nguyen HM, Kang D, Kim D, Kim Y (2018) ECG arrhythmia classification using a 2-D convolutional neural network. [arXiv:1804.06812](#)
 39. Park S, Kwak N (2016) Analysis on the dropout effect in convolutional neural networks. In: Computer Vision—ACCV 13th Asian conference on computer vision Taipei, Taiwan, November 20–24, Revised Selected Papers, Part II, pp 189–204
 40. Duchi J, Hazan E, Singer Y (2011) Adaptive subgradient methods for online learning and stochastic optimization. *J Mach Learn Res* 12:2121–2159
 41. Zeiler MD (2012) ADADELTA: an adaptive learning rate method. [arXiv:1212.5701](#)
 42. Kingma DP, Ba J (2015) Adam: a method for stochastic optimization. *CoRR*, [arXiv:1412.6980](#)
 43. Shopon M, Mohammed N, Abedin MA (2017) Image augmentation by blocky artifact in Deep Convolutional Neural Network for handwritten digit recognition. In: IEEE international conference on imaging, vision & pattern recognition, February, 13–14, Dhaka, Bangladesh, pp 1–6

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.