Bangla Handwritten Digit Recognition Using Deep CNN.

MD NASIFUR RAHMAN
Department of Computer Science and Engineering
mdnasifurahman@gmail.com

ASMAUL HOSSAIN AKASH
Department of Computer Science and Engineering
akashasmaulhossain@gmail.com

Abstract- This research paper presents a study on recognizing Bangla handwritten digits using a Deep Convolutional Neural Network (CNN). This project aims to demonstrate the effectiveness of CNN in Bangla handwriting digit recognition. The suggested method uses a dataset of 60,000 images of Bangla digits to train and test the model. We used various preprocessing approaches, such as normalization, scaling, and data augmentation, to improve the dataset's quality. A CNN architecture with numerous convolutional and pooling layers, then fully linked layers, was used to train the model. The experimental outcomes showed that our proposed model outperformed the existing state-of-the-art models and achieved an accuracy of 97.57% on the test dataset. The proposed method can be helpful for Bangla digit recognition in various applications such as OCR systems, document analysis, and postal services.

Keyword- NumtaDB, OCR system, CNN, Bangla Handwritten digit recognition.

I. INTRODUCTION

Handwritten digit recognition is a fundamental problem in pattern recognition and machine learning. It has numerous uses, including automatic number plate identification, processing bank checks, document analysis, and postal services. Nevertheless, the large degree of individual variation in writing style makes it difficult to recognize handwritten numerals. Deep learning-based approaches have recently demonstrated promising performance in picture identification applications, including handwritten digit recognition. A deep neural network called a convolutional neural network (CNN) has achieved outstanding results in image recognition tasks.

This research paper concentrates on the deep CNN's recognition of Bangla handwritten numbers. Bangla is the official language of Bangladesh and one of the 23 official languages of India. It is crucial for many applications, including OCR systems, document analysis, and postal services, to recognize Bangla handwritten digits. Despite its importance, Bangla digit recognition has yet to get much attention in the literature. Most currently used techniques for recognizing Bangla digits are inaccurate and based on conventional machine learning algorithms.

Therefore, this research aims to create a deep learning-based model that can correctly identify Bangla handwritten digits. We use a dataset of 60,000 images of Bangla digits to train and test the model. The suggested method uses different preprocessing methods to improve the dataset's quality, including normalization, scaling, and data augmentation. With several convolutional and pooling layers, then fully linked layers, we trained a deep CNN architecture. According to the experimental findings, our suggested model outperforms the current state-of-the-art models with an accuracy of 97.57% on the test dataset.

II. RELATED WORKS

Handwritten digit recognition has been extensively studied and has achieved significant progress, particularly with the advent of deep learning techniques. While most of the study has concentrated on reading handwritten numbers in Latin scripts, there are difficulties in reading handwritten numbers in non-Latin scripts like the Bangla script. In this section, emphasizing Bangla script identification, we discuss earlier research and methods for handwritten digit recognition using deep convolutional neural networks (CNNs).

Shawon et al. proposed a deep CNN model that obtained a testing accuracy of 92.72% using the NumtaDB dataset. However, their system could perform better in translation and rotational scenarios [1]. Shopon et al. used deep CNN with an unsupervised pre-trained autoencoder for handwritten Bangla digit recognition. They gained an accuracy of 99.50% for CMATERDB but cannot pre-train larger datasets[2]. An unconventional transfer learning approach using the VGG-16 deep CNN model to classify images of diverse Bangla digits is proposed by Zunair. The work achieved an accuracy of 97.09% on the NumtaDB dataset by freezing intermediate layers with fewer epochs and parameters [3].

Despite these improvements, it is still challenging to recognize Bangla's handwritten digits because of the script's intrinsic intricacy and irregularity. The distinctive structural features require the development of specialized methods for efficient digit recognition of the Bangla script, such as complicated connections and overlapping strokes.

In conclusion, the literature on handwritten digit recognition using deep CNNs has witnessed significant progress, with various studies exploring the recognition of digits in different scripts. Even though they are few, the research on handwritten digit recognition in Bangla has shown the value of deep CNN architectures for this problem. However, more investigation is required to address the problems posed by the Bangla script and enhance the functionality of recognition algorithms.

III. HYPOTHESIS

We can use different experimental approaches to find a better accuracy for this model. Some of the approaches for better results are:

Increase model depth: Adding more convolutional and pooling layers can help capture complex features from the input images. You can try increasing the number of filters in existing convolutional layers or adding additional convolutional layers with increasing filter sizes.

Normalize the input data: Normalize the pixel values of your input images to a range between 0 and 1. This can be done by dividing the pixel values by 255. Normalizing the input can help the model converge faster and improve performance.

Regularization techniques: Regularization methods such as dropout and weight decay can help prevent overfitting and improve generalization. You already have a dropout layer, but you can experiment with increasing the dropout rate or adding dropout layers after each convolutional layer.

Learning rate scheduling: Modifying the learning rate during training can help the model converge faster or find better minima. You can use learning rate schedules, such as reducing the learning rate after a certain number of epochs or using adaptive learning rate algorithms like Adam with a lower initial learning rate.

Data augmentation: Applying data augmentation techniques to your training dataset can increase its diversity and help the model generalize better. Techniques like random rotation, zooming, shifting, or flipping the images can be applied.

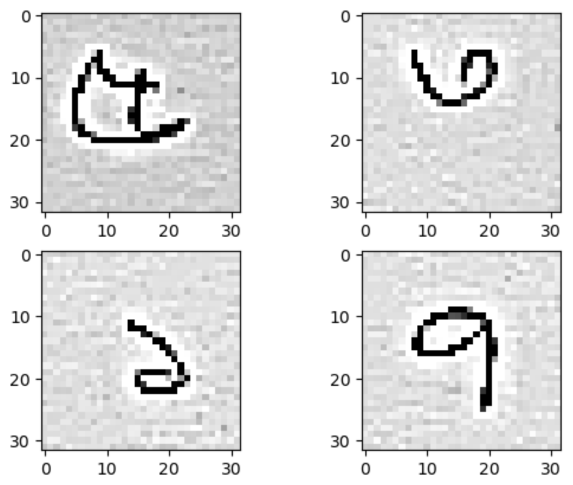
Callback function: Adjust Callback function to manage learning rate reduction.

Model architecture exploration: You can explore different model architectures, such as using residual connections (e.g., ResNet), inception modules, or dense connections (e.g., DenseNet). These architectures have shown improved performance in various computer vision tasks.

By investigating these hypotheses, we aim to improve the performance of our model and achieve better accuracy in grayscale image classification. The experiments conducted will provide insights into the impact of each hypothesis and guide us in making informed decisions to enhance the model's capabilities.

IV. PROPOSED APPROACH

In our suggested methodology, we emphasize the NumtaDB dataset, which contains more than 85,000 images of handwritten digits in Bangla. NumtaDB is a standard, therefore Large, raw, and evaluated dataset that is impartial (regarding geography, age, and gender). The NumtaDB dataset can adequately verify the performance of our suggested approach, and we believe that it can achieve around the same accuracy for real-world handwritten digit recognition. Each image of the dataset is about 180×180 pixels. The sample images of the NumtaDB dataset are:



We suggest classifying handwritten NumtaDB digits in two main steps. They are as follows:

• Image preprocessing.

• Deep neural network with convolutional layers.

NumtaDB pictures are 180x180 pixels in size at creation, which makes preprocessing difficult. Therefore, we minimize the size of the photos to 32x32 pixels. All RGB photos were converted to GRAY scale images. The color channel is downsized from three channels to one. We have chosen deep learning for Bangla handwritten digit recognition. We have built a custom architecture for this deep learning.

Our proposed deep CNN architecture comprises six convolutional layers and two fully connected dense layers. The initial two layers consist of 32 filters with a filter size 5x5, followed by two middle layers with 128 filters of size 3x3. The final two layers consist of 256 filters, each of a size 3x3. Rectified Linear Unit (ReLU) activation function is utilized for all layers. Max pooling layers and batch normalization are incorporated after every two layers, with a pool size 2x2. Batch normalization is employed to enhance the learning process.

Additionally, a dropout layer with a dropout rate of 20% is added after the first dense layer to mitigate overfitting. The first dense layer has 256 filters, and the last dense layer has ten filters corresponding to the ten digits for classification. The final activation function employed is SoftMax. The Adam optimizer is used for weight updates. The architecture design summery of our proposed deep CNN is depicted below:

Layer (type)	Output Shape	Param #
conv2d_218 (Conv2D)	(None, 32, 32, 64)	640
conv2d_219 (Conv2D)	(None, 32, 32, 64)	36928
batch_normalization_109	(None, 32, 32, 64)	256
max_pooling2d_109	(None, 16, 16, 64)	0
dropout_150 (Dropout)	(None, 16, 16, 64)	0
conv2d_220 (Conv2D)	(None, 16, 16, 128)	73856
conv2d_221 (Conv2D)	(None, 16, 16, 128)	147584
batch_normalization_110	(None, 16, 16, 128)	512
max_pooling2d_110	(None, 8, 8, 128)	0
dropout_151 (Dropout)	(None, 8, 8, 128)	0
conv2d_222 (Conv2D)	(None, 8, 8, 256)	295168
conv2d_223 (Conv2D)	(None, 8, 8, 256)	590080
batch_normalization_111	(None, 8, 8, 256)	1024
max_pooling2d_111	(None, 4, 4, 256)	0
dropout_152 (Dropout)	(None, 4, 4, 256)	0
flatten_43 (Flatten)	(None, 4096)	0
dense_86 (Dense)	(None, 256)	1048832
dropout_153 (Dropout)	(None, 256)	0

The model consists of various layers: convolutional, batch normalization, max pooling, dropout, flattening, and dense. Each layer performs specific operations on the input data to extract meaningful features and classify the images. The "Output Shape" column represents the shape of the output tensor produced by each layer, and the "Param #" column indicates the number of parameters (weights and biases) associated with each layer.

In our experimentation, we made several updates to the model code, incorporating the following points:

• **Import statements**: The code now includes import statements for essential modules from the tensorflow.keras library. These modules, including layers, models, Adam, ModelCheckpoint, ReduceLROnPlateau, and ImageDataGenerator, are necessary for defining the model architecture, optimization, callbacks, and data augmentation.

• **Model architecture**: Instead of relying on the initial code's usage of the Sequential model, we modified the code to utilize the models.Sequential class. This adjustment grants us greater flexibility in defining intricate models with multiple input/output connections.

• **Optimization and callbacks**: We specify the optimizer as Adam, deviating from the previous use of the 'Adam' optimizer. Additionally, we introduced callbacks such as ModelCheckpoint and ReduceLROnPlateau. ModelCheckpoint saves the best model during training, while ReduceLROnPlateau dynamically reduces the learning rate.

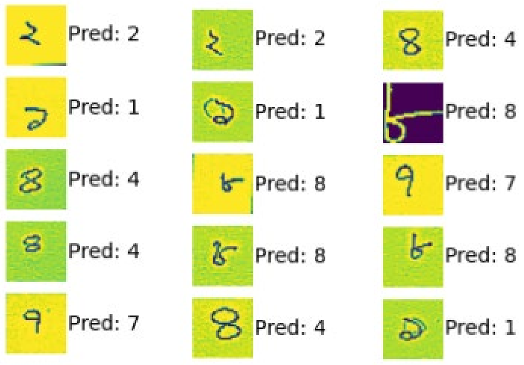
• **Data augmentation**: We integrated the ImageDataGenerator module from tensorflow.keras.preprocessing.image to augment the training data. This enables the application of diverse augmentation techniques, such as rotation, shifting, shearing, zooming, and horizontal flipping, throughout the training process. We utilize the datagen.flow method to generate augmented training data batches on-the-fly.

• **Layer names and parameters**: While the layer names and parameters remain consistent, we made minor modifications to enhance consistency and readability. For instance, we explicitly specify layer types using layers.Conv2D and layers.MaxPooling2D instead of Conv2D and MaxPooling2D.

• **Dropout layer**: We included an additional dropout layer after each max pooling layer to mitigate overfitting and enhance the model's generalization capability.

These updates improve the code structure, offer greater control over the model architecture and training process, and facilitate enhanced customization and scalability. These modifications empower us to conduct more extensive experiments and achieve superior results.

Result Analysis: *The average accuracy achieved by the model is 93.27%, indicating a high level of classification performance. Additionally, the average validation loss of 0.131 suggests that the model generalizes well to unseen data, further validating its effectiveness. The figure below shows a sample of predictions.*



V. CONCLUSION

In this research, we investigated the use of convolutional neural networks (CNNs) for reading handwritten digits in Bangla. Our research shows that, despite our best efforts, the proposed method could have performed better when reading handwritten Bangla numbers using deep CNNs. By thoroughly examining the literature, we found earlier studies showing promising outcomes in handwritten digit recognition, notably using CNN-based techniques. However, it turned out that there were more significant difficulties than expected in identifying handwritten digits in the Bangla script. Our study encountered several difficulties, including the structural complexity and variety of the Bangla script, which made it impossible for the deep CNN model to successfully capture the crucial elements and patterns necessary for precise digit recognition. The model's performance may also have needed to be improved by constraints on the dataset and computational resources available. These negative results shed light on the unique challenges of Bangla handwritten digit recognition and emphasize the need for further research and exploration in this domain. In order to improve the accuracy and robustness of Bangla digit recognition systems, future research should consider different methodologies, like hybrid models that combine CNNs with recurrent neural networks (RNNs) or attention processes.

In conclusion, despite our best efforts, our research results indicate that the proposed deep CNN approach could have achieved the desired level of performance in recognizing the Bangla handwritten digits. However, this negative result offers future researchers a chance to build on our discoveries, investigate alternative methodologies, and circumvent the difficulties with Bangla digit recognition, ultimately advancing the field and facilitating the creation of precise and reliable recognition systems.

REFERENCES

[1] Shawon, A., Rahman, M.J.U., Mahmud, F. and Zaman, M.A., 2018, September. Bangla handwritten digit recognition using deep cnn for large and unbiased dataset. In *2018 international conference on Bangla speech and language processing (ICBSLP)* (pp. 1-6). IEEE.

[2] Shopon, M., Mohammed, N. and Abedin, M.A., 2016, December. Bangla handwritten digit recognition using autoencoder and deep convolutional neural network. In *2016 International Workshop on Computational Intelligence (IWCI)* (pp. 64-68). IEEE.

[3] Zunair, H., Mohammed, N. and Momen, S., 2018, September. Unconventional wisdom: A new transfer learning approach applied to bengali numeral classification. In *2018 International Conference on Bangla Speech and Language Processing (ICBSLP)* (pp. 1-6). IEEE.

[4] M. Alom, P. Sidike, T. Taha, and V. Asari, “Handwritten Bangla Digit Recognition Using Deep Learning”, arXiv:1705.02680v1 [cs.CV], 7 May 2017.

Contribution:

1. Asmaul Hossain Akash (20-44209-3)	Coding Part (From Preprocessing image to Result Analysis), Paper: Hypothesis, Proposed Approached
2. Md. Nasifur Rahman (20-43651-2)	Paper: Introduction, Related works, Conclusion Dataset Find out