Evaluation of Bengali Handwritten Mathematical Equationusing Convolutional Neural Network

Tashfiq Nahiyan Khan Khan (khantashfiq565@gmail.com)
Ahsanullah University of Science and Technology
Rachita Dewanjee Dewanjee
Ahsanullah University of Science and Technology
Nushrat Jahan Shorna Shorna
Ahsanullah University of Science and Technology
MD Mejbah Ur Rahman Sowad Sowad
Ahsanullah University of Science and Technology
Md. Tanvir Rouf Shawon Shawon
Ahsanullah University of Science and Technology

Research Article

Keywords: Handwritten equation, CNN, ResNet18, Mathematical Equation

Posted Date: February 7th, 2023

DOI: https://doi.org/10.21203/rs.3.rs-2553612/v1

Additional Declarations: No competing interests reported.

Evaluation of Bengali Handwritten Mathematical Equation using Convolutional Neural Network

Tashfiq Nahiyan Khan[1*], Rachita Dewanjee[1†], Nushrat Jahan Shorna[1†], MD Mejbah Ur Rahman Sowad[1] and Md. Tanvir Rouf Shawon[1]

[1]Department of Computer Science and Engineering, Ahsanullah University of Science and Technology, Dhaka, Bangladesh.

*Corresponding author(s). E-mail(s): khantashfiq565@gmail.com;
Contributing authors: rachitadewanjee4@gmail.com; nushrat.j01@gmail.com;
mejbahurrahman13@gmail.com; shawontanvir95@gmail.com;
†These authors contributed equally to this work.

Abstract

Humans are naturally capable of solving mathematical expressions, but machines lack the ability to comprehend an issue through a visual context. Computers are gradually becoming more advanced and catching up with the subtlety and inaccuracy of real life. The need for an automated system to check answer scripts of mathematical equations has become unparallel, especially for Bengali handwritten scripts. This study checks each line of the solution of a mathematical equation to evaluate its correctness using a deep learning approach. In contrast to earlier methods, this paper introduces a CNN architecture to verify the accuracy of a handwritten mathematical equation in addition to solving the problem. The model reads a handwritten equation and validates its mathematical symbols and operations. A dataset has been created to evaluate the models performance which is named "BHQED". The experimental result shows that the accuracy of the proposed CNN architecture is 92.25% and the recall is 90.65% on our solely created dataset. To further boost the performance, this study applies the pretrained ResNet18 model and substantially outperforms the CNN with an accuracy of 94.57% and recall of 93.69%.

Keywords: Handwritten equation, CNN, ResNet18, Mathematical Equation

1 Introduction

There are many applications of mathematics in both natural science and human life. People use polynomial equations to draw the curves of roller coasters and bridges, undertake cost analyses, determine the motion of a particle under the effect of gravity, estimate sales trends, and construct profit margins in economics [1]. Recently everything is becoming digitalized and automated. Computers are gradually becoming more intelligent and catching up with the discrepancies and imprecisions of real life as human-computer interaction (HCI) becomes progressively less difficult. [1]. The level of education a person has unquestionably impacted their quality of life. Education enhances information and skills while also fostering the growth of personality and attitude. Exams are typically thoughtful assessment techniques in education. These are efficient techniques

for both assessing and enhancing students' learning. Teachers shall assess the answer sheets succeeding examinations. Teachers lose valuable time since checking the answer scripts takes a lot of time. Because of human error, they can misjudge when evaluating answer scripts. Our inspiration to create an automatic mathematical equation checker came from attempting to resolve these issues. With promising findings, Optical Character Recognition (OCR) problems were studied very early in the development of support vector machines [2]. Optical Character Recognition can enable excellent recognition accuracy when recognizing English characters and digits in electronic books [3]. However, OCR for mathematical documents is significantly more challenging than typical OCR tasks. In equations, the symbols are not always placed in a one-dimensional grid. There are subscripts, superscripts, fractional connections, and accents that can be nested [4]. It is more difficult with handwritten digit images because they differ from natural images. The alignment of a digit, along with the closeness of characteristics across distinct digits, causes confusion. Hence, handwritten mathematical expression recognition remains a difficult task in the field of computer vision [5].

Bengali is among the most widely spoken dialects on the globe, with around 220 million people using it for communication and writing [6]. But English, Greek, and Roman letters are among the font styles used in many articles, and there are a few that worked with Bengali handwritten mathematical equation recognition. Shuvo et. al.[7] represented a new system that takes an input image of a Bengali handwritten mathematical expression, automatically simplifies the problem and generates the answer as an output. They used a CNN architecture called MatheNET for segmented Bengali digits and mathematical symbols. Another paper proposed a task-oriented approach known as Bengali handwritten digit identification using DCCNN(BDNet) [8]. They also constructed a dataset of 1000 images of Bengali handwritten numbers to test their trained model. However, none of the publications we have come across have verified the accuracy of the equations. The majority of the papers we examined that dealt with Bengali mathematical equations employed automated recognition of written mathematical symbols and digits. We, therefore, aim to authenticate the solution of a handwritten Bengali mathematical equation from its image in this study. Our system carefully reviews every line of the solution to see if any line has errors or not. The output reveals which line is correct and which contains the error. We used four different datasets named as NumtaDB [9], Ekush [10], Kaggle Handwritten Math Symbols(1) [11] and Handwritten Math Symbols(2) [12]. In this paper, we primarily contributed the following:

- We have proposed a workflow to evaluate and validate Bengali handwritten mathematical equations. A workspace detection system has been developed to serve this purpose.
- Two distinct model architectures are proposed, one of which is CNN-based and the other of which is ResNet18 which is a pretrained CNN architecture. We have incorporated four different widely used datasets to train the models.
- We created our own dataset named **BHQED** to evaluate the models which can be found at https://github.com/tashfiq333/Bangla-Handwritten-Math-Equation and we also used a range of performance metrics to demonstrate how well the models performed.

The rest of the paper is organized as follows: Section 2 includes literature reviews of related papers and section 3 reviews the background study which includes the model description. Section 4 introduces the dataset description and collection. The methodology part is presented in Section 5. The results of our research are explained in section 7. We conclude this paper with a future research plan and conclusion in Section 8.

2 Literature Review

There has been substantial progress in the recognition of handwritten mathematical equations. The literature review section is divided into 3 different sub-sections. In the first subsection, works based on optical character recognition are described, whereas the second one describes papers on digit recognition. The third subsection holds the works based on handwritten mathematical expression recognition.

2.1 OCR Based Papers

The study [13] by Jamshed Memon et al. thoroughly reviewed 176 papers on handwritten OCR that were published between 2000 and 2019 in order to conduct a structured literature study. According to this study, the OCR gives priority to six different languages. After executing a thorough review approach using electronic resources, it has been found that, while published research papers have proposed a number of alternative OCR systems, a significant area still required progress like data analysis commercialization. A great deal of helpful information may be transformed into usable, digital data with the development of practical and economical OCR technology. This research [14] by Noman Islam et al. incorporates data from several previously published OCR studies. The pre-processing, symbol segmentation, feature extraction, classification, and other processes required for a good OCR system were all outlined. In the study [15] by Mujibur Rahman Majumderet al. showed a full OCR system that was successful in handling English characters, particularly handwritten Calibri typeface. Mostly in event that the digital image was not properly oriented, this approach first corrects the image's imbalance, then reduces noise, separates the individual image into rows and characters, and lastly recognizes the characters within the inputs. Researchers experimented until they found a median for both the Calibri font around 92% to 98%. They came up with a technique that outperforms current OCR technologies in speed and effectiveness. In order to improve Chinese handwriting recognition, the research [16] by Yi-ChaoWu et al. evaluates the effects of two symbol-level NNLMs, notably (FNNLMs) as well as (RNNLMs). BLMs were additionally combined between both to produce hybrid LMS. Research on the Chinese handwriting dataset named CASIA-HWDB shows that NNLMs improve detection capability, whereas hybrid RNNLMs surpass other LMS.

2.2 Digit Recognition Based Papers

The study [17] by Muntarin Islam et al. proposed a handwritten character system consisting of ResNet50, Inception-v3 and EfficientNet-b0 with NumtaDB dataset consisting of 17,000 examples of handwritten Bengali digits having 10 classes. The result of their models is quite impressive, achieving 97% accuracy mostly in 10 categories. This study also contrasts various pre-trained algorithms for Bengali handwritten digit recognition systems. Testing accuracy they achieved for InceptionV3, EfficientNet-b0 and RestNet50 are 88%, 96%, and 94%. The Support Vector Machine, Conditional Random fields, Decision Tree, and K nearest Neighbor, in addition to a CNN, were utilized as just the classification algorithm in the study [18] by Ayushi Sharma et al. The CNN they have used outperforms some other classifiers, identifying handwritten characters with a testing accuracy of only 98.83%. The main resource for separating learning seems to be the Standard MNIST dataset [19], which was released by York University. Throughout the study [20] by Khalil Ahammad et al., the concept of convolutional neural networks were used to recognize single numbers in Bengali Sign Language via gestures. The suggested approach makes use of 160 of the 180 BdSL images that were directly gathered from state residents. This CNN architecture was subsequently developed using these images following extensive pre-processing. To accurately identify different classes, CNN uses several layers. Whenever the model has been trained on images without spinning, it achieves an accuracy of 94.17%, although when spinning is applied, it achieves an accuracy of 99.75%.

2.3 Handwritten Mathematical Expression Recognition Based Papers

A method comparable to ours was used in the project[1] to automatically analyze the English mathematical equation. The report the contributor submitted states that they used a sample worksheet with mathematical calculations to which several writers contributed. Line width, word style, ballpoint pen tip size, word alignment, etc. can all vary. This box was placed on the worksheet after each row had previously been adjusted. Workstation detection and evaluation modules are process components. This method of workspace recognition finds several workspaces on something

[1] https://github.com/divyaprabha123/Autograding-handwritten-mathematical-worksheets

similar to a single piece of paper. This is presuming that the worksheet contribution has precise rectangular borders. Because they are designated working areas, the three biggest rectangular boxes on the sheet were included on purpose. Starting with rectangular blocks, disregard all numerals, symbols, and even other writings on that worksheet. These boxes are made up of these two horizontal and vertical planes. Possibly all of the rectangles have been found; put them in the spreadsheet from top to bottom in the order they were found. Then appropriate workstations were chosen from each of the available squares. The regions that might be used for activities appear to be the three largest rectangles. Rectangles' dimensions should be computed, and the biggest three with the largest areas should be chosen. The workplaces from the datasheet have been extracted using these chosen rectangles. Finding the line is the first step in the analysis process. This method of line identification makes the assumption that there is enough space between lines and that lines with exponential digits would eventually cross. After line identification, it locates contours to classify and arrange the characters, anticipating the numbers and exponential for each row. To create the model, the authors used the Handwritten Arithmetic Symbol sample as the dataset for signs and the MNIST dataset [19] as the dataset for numbers. Over 60,000 images of processing symbols and digits were used to train this DCCNN. The accuracy of the aforementioned model was 96% at its maximum level. The model was evaluated during the last phase, and only the results from this stage were used to create the boxes. The primary objective of the paper [21] was to identify HME utilizing CNN. The images were preprocessed utilizing grayscale conversion, noise reduction that uses the average, binarization using cutoff, and flattening to maintain the depth of the foreground after the HME database was assembled by a number of researchers. Single pixel to ensure that the picture may be processed clearly. They used CNN's 83 different classifications as the classifier. The reliability of just this model was roughly 87.72 %.

This article [22] by Md Bipul Hossain et al. shows a process of identifying and dealing with a single mathematical problem as well as a group of scrawled math questions. They created several samples and also utilized a customized MNIST database. One grayscale image of 32*32 pixels was utilized for the dataset. Initial preparation of such dataset photos included grayscale transformation, noise removal, and binarization with just an adjustable threshold. Integrated connected method, as well as horizontal compacted project analysis, are two segmentation approaches. A resource for these CNN input layers was a 32x32 picture dataset that was used during the work's learning phase. The categorization model is based on CNN. The system correctly identified 39.11% of something like the formulas in the set of 1000 images, whereas symbol segmentation accuracy reached 91.08%. CNN architecture is used in this formula identification system in the article [23]by Rajwardhan Shinde et al. The proposed method recognizes and handles polynomial equations as well as basic operations (-, +, /, *) with many digits. MNIST sample and a manually produced collection of signs ("-","+", "/", "*", "(")) were both utilized to build the model. This system further used RNN to handle the recognized activities. Having 96% accuracy, this RNN model has been used to predict the consequences of the current operation's outcomes. Throughout this study, the CNN architecture is utilized to recognize equations of different degrees with an accuracy of nearly 85%.

Since the original problem is essentially a classification task, the Deep CNN was applied in the study [24] to recognize handwritten symbols in Bengali. They used 2 well-known databases, BanglaLekha-Isolated [25] with NumbtaDB [9], including both numerals and signs. They expanded their dataset by applying a shifting approach while running several tests on vowels, numerals, as well as symbols. As a consequence, BanglaLekha's accuracy level rose to 96.42%. Here on the NumtaDB sample, their system also achieved an accuracy of 98.92%.

3 Background Study

In this study, a pretrained CNN-based architecture called ResNet18 and a CNN architecture are both utilized. On the previously mentioned datasets, we first applied CNN. Then, to further improve the model's accuracy, we applied ResNet18. This section provides a description of these models along with a description of the performance metrics we used in our work.

3.1 Convolutional Neural Network

The Convolutional Neural Network (CNN) is one of the most used deep neural networks. The word "convolution" refers to a mathematical linear operation between matrices[26]. In machine learning, CNN performs quite well. Particularly the programs that work with image data. Pooling and non-linearity layers don't have parameters, but convolutional and fully connected layers do. In the literature, there are a variety of different CNN architectures. However, they share a lot of fundamental elements. Convolutional, pooling, and fully connected layers make up the majority of its layers. The convolutional layer's objective is to learn input feature representations. Several convolution kernels make up the convolution layer, which is utilized to compute various feature maps. To detect nonlinear features of multi-layer networks, CNN uses the activation functions that introduce nonlinearities. Typical activation functions are sigmoid, tanh and ReLU[27]. By lowering the resolution of the feature maps, the pooling layer seeks to achieve shift-invariance. Normally, it sits between two convolutional layers. One or more fully-connected layers intend to carry out high-level reasoning after numerous convolutional and pooling layers. To provide global semantic information, they connect all of the neurons from the previous layer to every single neuron in the current layer[27].

3.2 Transformer model : Resnet18

The residual network has multiple variations, such as ResNet16, ResNet18, ResNet34, ResNet50, ResNet101, ResNet110, ResNet152, ResNet164, ResNet1202, and so on. The ResNet stands for residual networks[28]. A 7x7 kernel serves as the first layer of RESNET18 which has 18 layers. It has identical convolutional layers. Each layer is made up of two residual blocks. Each block has two weight layers and a skip connection that is connected to the second weight layer's output by a ReLU activation function. The identity connection is used if the outcome matches the convolution network layer's input. However, a convolutional pooling is performed on the skip connection if the input and output are not identical[29]. There is an issue of accuracy degradation in deeper networks to resolve this problem. Kaiming He et al.[30] proposed a technique using residual learning to train deeper networks, which are practically challenging to train. To learn residual functions in relation to the layer inputs, network layers were reformulated. The findings show that residual learning-based deep networks can achieve superior optimization and high accuracy. In simple networks, the desired mapping is immediately learned by stacking several layers together. In contrast, the layers in residual networks are stacked to learn a residual mapping. A few stacked layers are used to fit the mapping function. According to the theory behind residual learning, if numerous nonlinear layers can asymptotically estimate a challenging mapping function, they can do the same for the residual function [31]. The architecture of the residual 18 network has eighteen layers in total, including 17 convolutional layers, a fully-connected layer, and an additional softmax layer for classification tasks. The convolutional layers use 3×3 filters, and the network is designed so that if the output feature map is the same size, the layers have the same number of filters, but if the output feature map is halved, the filters in the layers are doubled. Convolutional layers with a stride of 2 perform the down sampling. Finally, an average pooling is used, followed by a fully-connected layer with a Softmax layer. Between layers, residual shortcut connections are inserted throughout the network. There are two kinds of connections. When the dimensions of the input and output are the same, ResNet18's first type of connection is utilized. When dimensions increase, the second type of connection is used. Identity mapping is still carried out using this kind of connection, but with padding of zeros for larger dimensions and a stride of two.

3.3 Performance Metrics

The most logical performance metric is accuracy, which is just the proportion of properly predicted observations to all observations. In terms of positive observations, precision is the proportion of accurately anticipated observations to all predicted positive observations. A good classifier's precision should preferably be 1 (high). Recall (sensitivity) is the proportion of accurately predicted positive observations to all of the actual class observations. The ideal recall for a good classifier is 1 (high). The weighted average of Precision and Recall is the F1 Score. Therefore, both

false positives and false negatives are considered while calculating this score. Although F1 is generally more beneficial than accuracy, especially if you have an uneven class distribution, it is not intuitively as simple to understand as accuracy.

4 Dataset

We had to deal with expressions that contain several characters as variables, mathematical operators, and mathematical symbols in order to solve an equation. The Bengali numerical digits are represented in the NumtaDB[9] and Ekush[10] dataset, but no mathematical symbols are included. We have used the Kaggle Handwritten Math Symbols(1)[11] and Handwritten Math Symbols(2)[12] datasets to obtain handwritten mathematical symbols.

4.1 Ekush

Ekush dataset is the biggest Bengali handwritten character dataset for computer vision research on Bengali handwritten character recognition. It not only contains Bengali handwritten digits but also contains Bengali handwritten vowels, consonants, and compound letters. There are $367,018$ images of handwritten characters written by 3086 unique writers. Among these images, there are $30,687$ Bengali handwritten digit images that we used for our work. In digit images, there are 10 numbers of classes $(0-9)$ which can be seen in Fig. 1a. Since the model had trouble detecting 5 and 6 after being trained with those photos, we did not include them in our study.

4.2 NamtaDB

NumtaDB [9] is a widely used dataset for Bengali digit recognition. It consists of more than $85,000$ images of Bengali handwritten digits. Among these $72,044$ of them are training images and $13,552$ of them are testing images. The images were mainly collected from university students. There is no fixed dimension of the images the width differs from 29 to 879 pixels and the height differs from 73 to 765 pixels. There are 10 numbers of classes (0-9). Fig. 1b shows some samples from the NamtaDB dataset.

(a) (b)

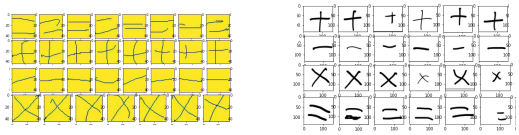
Fig. 1: Sample images from Ekush dataset (*left*) and NumtaDB dataset (*right*).

4.3 Handwritten math symbols(1)

The dataset named Handwritten math symbols(1) was collected from an open source website named Kaggle[2]. The dataset contains all kinds of math symbols except Hebrew alphabet. In this dataset, there are all the math symbols like $log, lim, cos, sin, tan, \int, \sum, \sqrt{}, +, -$, and more. For our work we used only $+, -, =, (,), \{, \}, [,], \times$ total 10 number of classes Fig. 2a. The image dimension is 45×45 pixels. Fig. 2b shows some samples of the dataset.

4.4 Handwritten math symbols(2)

This dataset named Handwritten math symbols(2) was also collected from an open source dataset called Kaggle[3]. The Dataset contains 9000 images of handwritten digits and arithmetic operators. The size of the images is mostly 400×400 and 155×155. The data sets consist of 19 classes labeled 0 to 9 having symbols like $(+, -, \times, =$, $\div$,x,y,z). For our work, we only used $(+, -, \times, =$,) signs which are necessary for mathematical equations. Some samples of the dataset can be seen in Fig. 2b.



(a) (b)

Fig. 2: Sample images from Handwritten math symbols(1) (*left*) and Handwritten math symbols(2) (*right*)

[2] https://www.kaggle.com/datasets/xainano/handwrittenmathsymbols

[3] https://www.kaggle.com/datasets/sagyamthapa/handwritten-math-symbols

4.5 BHQED (Our Created Dataset)

Since there isn't a dataset[4] for Bengali handwritten equations, we had to develop one for testing purposes. We have named our dataset as **BHQED** which stands for **Bengali Handwritten Quadratic Equation Dataset**. To serve this purpose, we have gathered Bengali handwritten math equations from a variety of writers with various handwriting styles. The equations are written in the format of $a * b^2 + c * d$. The dataset contains both successfully calculated equations and erroneously calculated equations for testing purposes. We chose equations with comprehensible text. Fig 3 shows some images from our dataset.

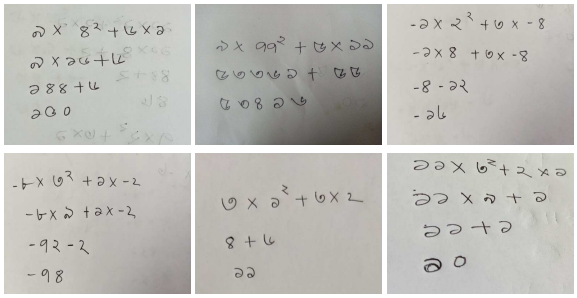


Fig. 3: Sample of our collected Bengali handwritten mathematical equation dataset.

5 Methodology

This section discusses our proposed methodology for evaluating Bengali handwritten math equations. The overall process is executed in seven major steps. This section briefly discusses the dataset preparation process to train our model and the overall process of achieving the final output. A graphical representation of our proposed methodology can be seen in Fig. 4.

5.1 Dataset Preparation

The format of the data was not the same for each data because it was compiled from various sources. Our dataset required to be formatted using just one style. So, the subsequent actions were taken.

- The labels of NumtaDB[9] dataset were given in a csv file, however, the labels of the data from Handwritten math symbols(1), Handwritten math symbols(2) and Ekush[10] dataset were not given in a csv file. As a result, accessing the labels could be troublesome. We had to write the symbols and Ekush[10] data sets labels to a csv file in order to avoid these problems.
- The image format of NumtaDB was PNG but all the other datasets were in JPG format. So, we converted the image format of all the other dataset to PNG.
- While all the previous datasets used 24-bit color photos, the NumtaDB dataset used 8-bit images, thus we transformed all of the images to 8-bit.
- We also converted all the images to 28×28 pixels before feeding the models.
- However, after training the model with Ekush dataset we found that the model was having a problem identifying the digit five (5) and six (6) as both the digits are written in a similar manner in the Ekush dataset. So we excluded the images of digit five and six.

5.2 Model Training & Prediction

The mathematical symbols for our proposed work were originally obtained from Handwritten Math Symbols(1), while the Bengali mathematical digits were obtained from NumtaDB (0-9) dataset. Following that, additional mathematical symbols using Handwritten Math Symbols(2) and additional Bengali math digits using Ekush were added to increase accuracy. Our work mainly consists of two major steps. First, extracting the math equation from the given image of the handwritten math equation Fig. 5a and then recognizing the math digits and symbols from the extracted equation Fig. 5b. Secondly generating green or red lines based on whether the extracted line of the equation is correct or not (5c) where green lines mean the correct solution and red lines mean the wrong solution.
The major steps for performing the proposed work are as follows:
Step 1) Image input: Firstly an image of the handwritten equation is taken and uploaded for background removal.

[4] https://github.com/tashfiq333/Bangla-Handwritten-Math-Equation

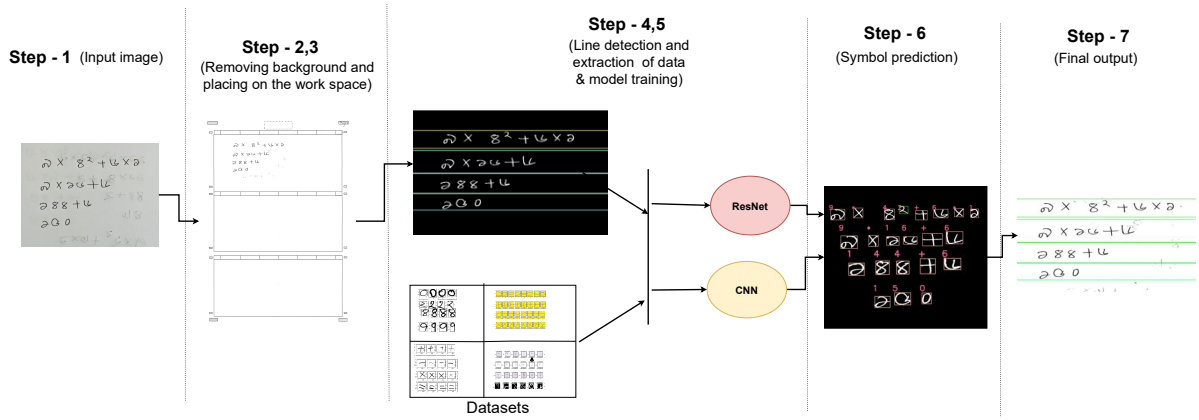


Fig. 4: Workflow of the proposed approach.

(a) (b) (c)

Fig. 5: A sample of line extraction (*left*), a prediction of the symbols and the digits (*middle*) and a final output of the model (*right*).

Step 2) Image background removal: The background of the input image is removed using the API provided by removebg[5].

Step 3) Placing the image in workspace : The image with the transparent background was then added to the workspace we had already set up. Fig. 6.

Step 4) Extracting equation: The process for the extraction of the equation is inspired by the previously mentioned project[6]. The steps required for the extraction of the equation are as follows:

- **Workspace Detection:** The task of detecting numerous workspaces on a single piece of paper falls to the Workspace Detection module. Valid workplaces must be found in order to locate the rectangle that contains the equations.
- **Finding Rectangular Boxes:** Identifying all the horizontal and vertical lines is the initial step, keeping in mind that two horizontal and two vertical lines intersect to make rectangles. When looking for the horizontal and vertical lines on the sheet, any numbers, symbols, and other contents are ignored. Then two different binary images are created, having vertical and horizontal lines respectively. The two images are combined to get the final image. The next step is to extract the image's contours (The line connecting all the points along an edge of the image with the same intensity is known as a contour).
- **Sorting and selecting the rectangles:** Now considering that every rectangle has been found, we may order them by their coordinates from top to bottom. Although there are many rectangles, just the largest three are needed. We can decide which three rectangles are the largest by measuring the dimensions of each one and then selecting the top three with the largest area. The analysis module receives the work locations after they have been obtained from the worksheet.
- **Segmentation:** The majority of image processing and computer vision applications use segmentation to identify objects or other important information in digital images. The fragmentation of one image into multiple pieces is what it is. In our suggested strategy, there are two basic steps for performing the segmentation which is as follows:

[5] https://www.remove.bg/api

[6] https://github.com/divyaprabha123/Autograding-handwritten-mathematical-worksheets

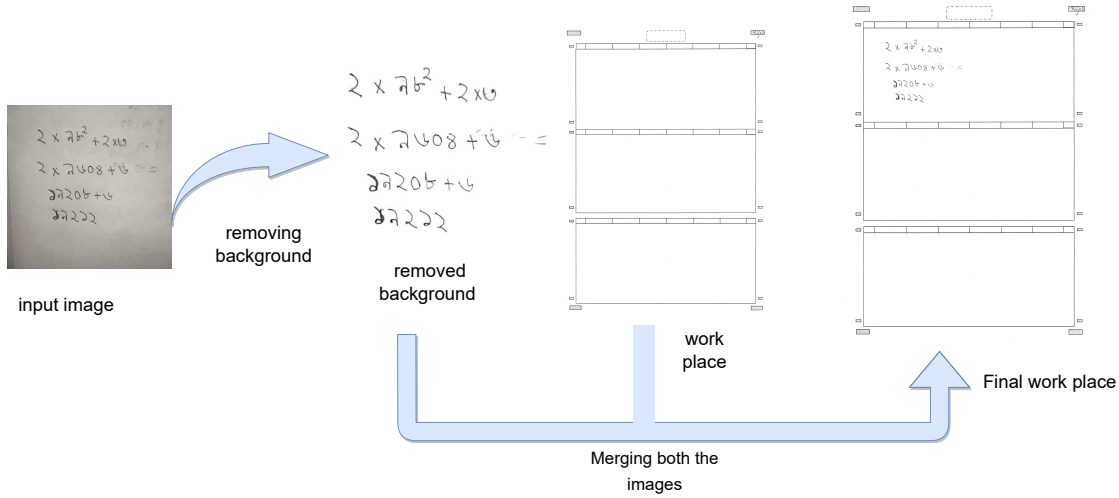


Fig. 6: Process of placing the handwritten equations on the template after removing the background.

Equation Line Segmentation: The division of the several lines of characters that are present in the image is known as equation segmentation. A minimal vertical space separates the characters on each line from those above and below it. This helps to clearly distinguish each line. This space can be used to identify and separate different lines of characters.

Character Segmentation: The goal of character segmentation is to separate an image of a group of characters into smaller representations of individual symbols. For the segmentation of certain characters from the image in our suggested methodology, the idea of connected component analysis is employed.

Step 5) Model Training: For predicting the mathematical digits and symbols, two distinct models are trained on the aforementioned dataset using a pretrained model called "ResNet18" and a manually constructed model based on CNN.

Step 6) Symbol Prediction: Then the previously trained model recognizes the characters in each line, and finally, shows the predicted characters before assessing them by putting boxes in the characters Fig. 5b.

Step 7) Evaluation and Drawing boxes: The evaluation is the last and most important step. After proper character segmentation, the given math problem is solved line by line, and the answers are stored in memory. Then, the handwritten equation lines are compared with the result of the previously saved solution. A green line is drawn once the correct line in the solution has been found otherwise the line is drawn as red. An equation is surrounded by a blue line if the line has incorrect mathematical syntax also.

6 Proposed Models

First, CNN was used to identify the mathematical symbols and digits. The model was first trained using Handwritten Math Symbols(1) and NumtaDB, and the accuracy was 97%. When we used the manually constructed dataset BHQED to test the model, we discovered that it was underperforming. It was having trouble recognizing the multiplication symbol as well as a few numerals. We utilized ResNet18 to solve this problem and trained the model on the same dataset. The accuracy this time was 98%. The model was then put to the test with BHQED, and we discovered that it was still having trouble recognizing some multiplication signs. So, for additional math symbols, we used Handwritten Math Symbols(2) dataset, and for additional Bengali math digits, we used Ekush dataset. We obtained 99.14% accuracy from ResNet18 and 99.10% accuracy from CNN for

BFQED dataset after adding additional math digits and symbols. The architecture of CNN and ResNet18 can be seen in Fig. 8 and 7 respectively.

7 Results and Experiments

This section talks about the experiment's results and overall outcome. Here, the performance of both models, their hyperparameters, and a comparison of the models are all covered.

7.1 Hyperparameter Settings

The learning rate and batch size for training CNN and ResNet18 were identical for both models. The batch size was 64, and the learning rate was 0.001. We applied 10 epochs for CNN and obtained an accuracy of 99.10%, while we used 73 epochs for ResNet18 and obtained an accuracy of 99.18%.

7.2 Performance of the Models

We present the overall performance of the CNN and ResNet18 model by using the confusion matrix also the accuracy and loss graph. And performance matrix of ResNet18 and CNN are given in Table 1.

Table 1: Performance matrix of ResNet18 and CNN

Model Name	Accuracy	Precision	Recall	F1-score
ResNet-18	94.27	1	.9369	.9674
CNN	92.25	1	.9065	.9510

7.2.1 CNN

After using CNN to test the handwritten equations, the confusion matrix shown in the accompanying Fig. 9 was produced. The confusion matrix indicates that the CNN's accuracy was 92.25% and that its recall was estimated to be 90.65%. We attained a validation accuracy of 92% while training the CNN. As shown in Fig. 11, 10 iterations were used to train the model, and after 10 iterations, validation accuracy had been found to be 99.10% and validation loss to be 0.051. Because there was a possibility of overfitting, the model was not taken further. The precision value, which can be seen as 1, indicates how carefully the model can avoid false positive cases but The .9065 recall value reveals CNN's incapacity in the event of false negative cases.

7.2.2 ResNet18

We utilized the ResNet18 model to try to improve the CNN result even though it is already satisfactory. The confusion matrix in Fig. 10 was formed by the ResNet18 performance. The ResNet18 model's accuracy and recall, as determined by the confusion matrix, are 94.57% and 93.69% respectively. Results from the ResNet18 model outperformed those from the CNN architecture. This is due to ResNet18's ability to train huge numbers of layers quickly without influencing the training error percentage. ResNet18 resolves the vanishing gradients problem. While testing the data on **BHQED** with ResNet18, 99.14% validation accuracy was attained. The ResNet18 model was trained over 73 iterations, and after 73 iterations, 99.14% validation accuracy was discovered and the validation loss was 0.152, as shown in Fig. 12. The model was not advanced further due to the possibility of overfitting. For this model, the precision parameter is also 1, making it identical to CNN. Recall in this instance is almost 3% greater than CNN, which shows an improvement in CNN's false negative cases.

7.3 Experimental Analysis

Here are the results from the experiments we conducted for our proposed study. We also discussed the many challenges we faced and how we overcame them.

7.3.1 Red Line

When a user makes a mathematical error in any of the steps of the mathematical equation, the model will surround that specific error step with a red line. If there are errors in multiple steps in the calculation, then all of the steps containing errors will be surrounded by separate red boxes also. This scenario arrived in a few cases. Some samples are shown in Fig. 13 where the red lines are clearly visible marking the mistakes. In the leftmost image, the output should be $18 + 6$ which is 24 but the prediction is 20 which is wrong. So the equation is marked by the red line.

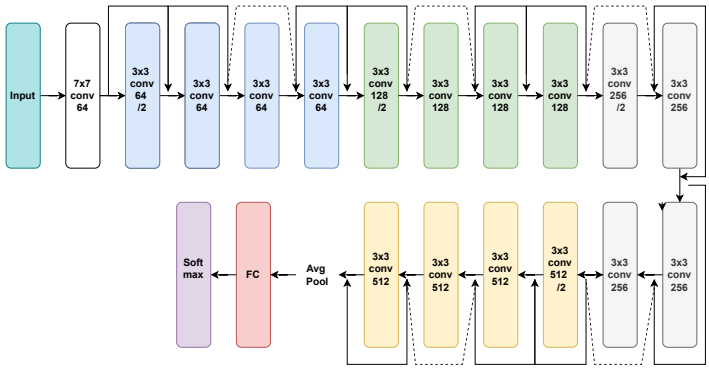


Fig. 7: The architecture of ResNet18

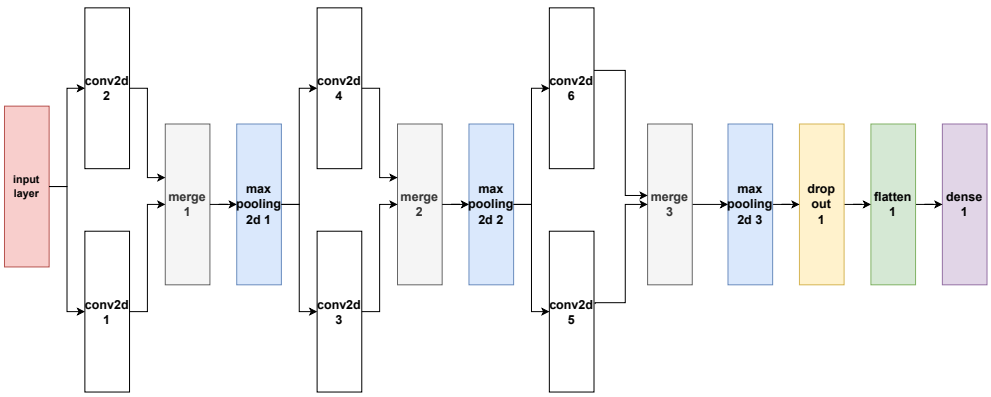


Fig. 8: A diagram of the proposed CNN architecture.

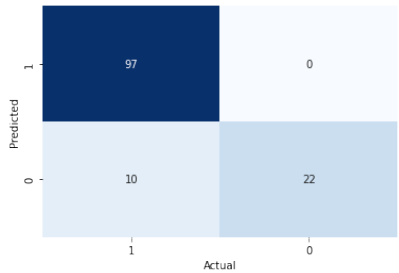


Fig. 9: The confusion matrix of CNN

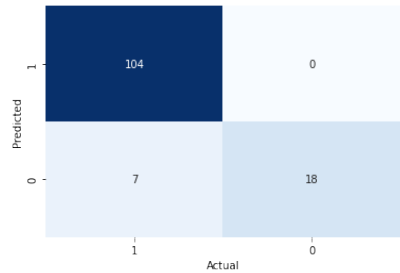


Fig. 10: The confusion matrix of ResNet18

7.3.2 Green Line

The model will indicate the mathematical step with a green line indicating that the step is mathematically valid when the computation is accurate. The model does this by calculating the answer of the equation separately using the predefined values of the variables and then comparing it with the input handwritten equation. Some samples are depicted in Fig. 14 where all the correctly performed equations are marked by green lines.

7.3.3 Blue Line

An equation is surrounded by a blue line if the line has incorrect mathematical syntax. Syntax errors

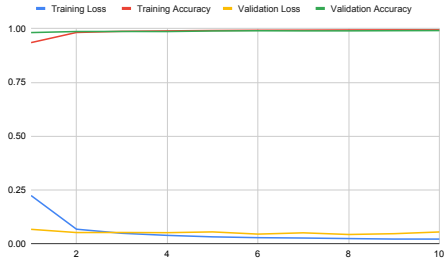


Fig. 11: Accuracy & loss graph of CNN

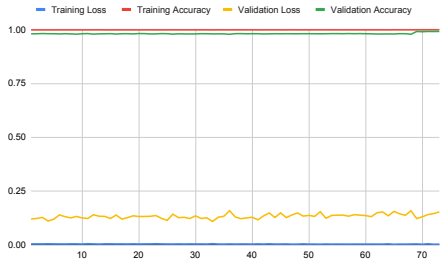


Fig. 12: Accuracy & loss graph of ResNet18

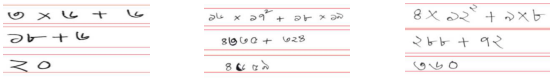


Fig. 13: Sample images of wrong equations marked by red lines.

like when there are two or more than two multiplication signs between two digits (Fig. 15a) or after a multiplication sign if there are no digits (Fig. 15b) in the equation. This blue line does not indicate that the calculation is wrong, but it means that the line does not follow the correct mathematical syntax of an equation.

7.3.4 Limitation of Dataset

Previously, we trained the CNN using the Numtadb dataset and the Handwritten mathematical symbols(1) datasets, however, the accuracy and recall values were unsatisfactory. Most of the multiplication signs were put in the wrong classification. After the ResNet18 model had been trained, the situation was almost identical. The majority of the handwritten equations were improperly validated as a result of a significant number of miss-classifications, which decreased accuracy and recall values. The model was then fed

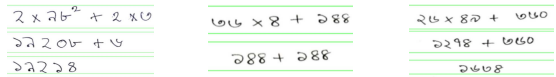
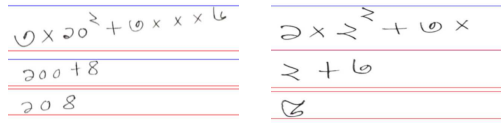


Fig. 14: Sample images of correct equations marked by green lines.



(a) (b)

Fig. 15: Sample images of syntactically wrong equations by blue lines.

2 new distinct datasets, Ekush and Handwritten mathematical symbols(2), to help solve the miss-classification issue. Accuracy and recall both increased after utilizing the Ekush and Handwritten mathematical symbols(2) datasets. Misclassification of the symbols happened far less frequently.

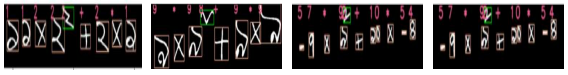


Fig. 16: Misclassified mathematical characters and digits

In a few instances, the model was unable to recognize specific mathematical characters as seen in Fig. 16. It was due to various reasons e.g there was some bad data in the dataset that we used, illegible handwritten characters or scribbling, etc. In Fig. 16, we can see the minus sign, and some of the digits are incorrectly recognized. Hence it continues with the calculation taking the incorrectly classified signs and digits. In a relatively small number of situations, this miss-classification occurred.

7.3.5 Line and character spacing Problem

We faced a few problems while reading characters and exponents from the handwritten equations. When the characters touched one another in the equation Fig. 17, the model failed to separate

the characters and mistaken the characters for another character. To avoid this problem we set a constraint that the handwritten characters should have an adequate amount of space between them.

Fig. 17: A visual demonstration of character and line spacing problem

7.4 Comparison between CNN and ResNet18:

After conducting the testing and training with both models, we compared the results and statistics obtained from them. The results are shown with a graphical representation in Fig. 18. From the graphs, we can infer that the sensitivity and F1 score for ResNet18 is higher than CNN. This is because the ResNet18 model correctly identified a higher number of the mathematically correct and incorrect steps from the handwritten mathematical equations and there were fewer false negatives i.e the mathematical steps were correct but the model identified them as incorrect. We can also see that the precision value is 100% for both models. This is because there were no false positives i.e neither of the models has identified an incorrect mathematical step as correct, and all the incorrect mathematical steps were identified as incorrect by both models.

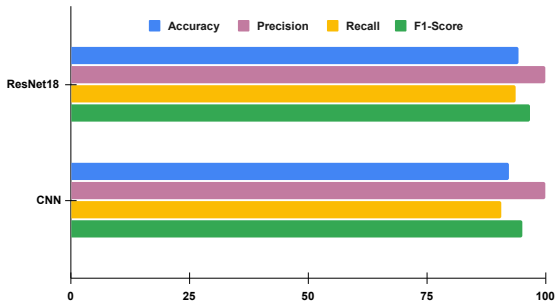


Fig. 18: A graphical comparison between ResNet18 and CNN

8 Conclusion and Future Work

The detection of Bengali handwritten characters and signs is indeed a difficult study topic because of the language's significant number of intricate characters and signs. Our study assessed how successfully handwritten Bengali equations are recognized by the state-of-the-art CNN and Restnet-18 architectures. Compared to the CNN architecture, the ResNet18 approach has done substantially better. Studies employing 4 large datasets for Bengali handwritten digit recognition show the effectiveness of these 2 models for Bengali HCR. This segmentation technique may be used more effectively in the next work to considerably enhance the outcome. But if there is not enough gap between digits and lines, it will not be properly identified. Additionally, if somehow the exponential digit is not written explicitly, then it is mistakenly identified. We hope this research can contribute to the OCR industry to a great extent. The proposed method may be developed into complicated problems, like trigonometric equations, utilizing the same detection and segmentation procedure,

9 Declarations

9.1 Ethical Approval and Consent to participate

Not Applicable.

9.2 Consent for publication

Not Applicable.

9.3 Human and Animal Ethics

Not Applicable.

9.4 Competing interests

The authors declare that they have no competing interests.

9.5 Funding

No funding was received for conducting this study.

9.6 Authors' contributions

Tashfiq and Shawon set the research scope, coordinated this research, coded, ran a few experiments, and drafted the manuscript. Nushrat and Rachita collected data and ran most experiments. Sowad ran a few experiments.

9.7 Data Availability

The data used in this study is publicly available at https://github.com/tashfiq333/Bangla-Handwritten-Math-Equation. The data is accessible to anyone and can be used for replication or further analysis.

References

[1] Lokhande, M.: Handwritten Math Problem Solver Using Convolutional Neural Network - IJARCCE — ijarcce.com. [Accessed 17-Nov-2022]. https://ijarcce.com/wp-content/uploads/2021/03/IJARCCE.2021.10307.pdf

[2] Malon, C., Uchida, S., Suzuki, M.: Support vector machines for mathematical symbol recognition. In: Yeung, D.-Y., Kwok, J.T., Fred, A., Roli, F., de Ridder, D. (eds.) Structural, Syntactic, and Statistical Pattern Recognition, pp. 136–144. Springer, Berlin, Heidelberg (2006)

[3] Hossain, M.B., Naznin, F., Joarder, Y., Islam, M.Z., Uddin, M.J.: Recognition and solution for handwritten equation using convolutional neural network. In: 2018 Joint 7th International Conference on Informatics, Electronics & Vision (ICIEV) and 2018 2nd International Conference on Imaging, Vision & Pattern Recognition (icIVPR), pp. 250–255 (2018). IEEE

[4] Eto, Y., Suzuki, M.: Mathematical formula recognition using virtual link network, pp. 762–767 (2001). https://doi.org/10.1109/ICDAR.2001.953891

[5] Lu, C., Mohan, K.: Recognition of online handwritten mathematical expressions using convolutional neural networks. cs231n project report stanford (2015)

[6] Hasan, F., Shuvo, S.N., Abujar, S., Mohibullah, M., Hossain, S.A.: Bangla continuous handwriting character and digit recognition using cnn. In: Innovations in Computer Science and Engineering, pp. 555–563. Springer, Singapore (2020)

[7] Rabby, A., Abujar, S., Haque, S., Hossain, S.A.: Bangla handwritten digit recognition using convolutional neural network. In: Emerging Technologies in Data Mining and Information Security, pp. 111–122. Springer, Singapore (2019)

[8] Sufian, A., Ghosh, A., Naskar, A., Sultana, F., Sil, J., Rahman, M.M.H.: Bdnet: Bengali handwritten numeral digit recognition based on densely connected convolutional neural networks. Journal of King Saud University - Computer and Information Sciences **34**(6, Part A), 2610–2620 (2022). https://doi.org/10.1016/j.jksuci.2020.03.002

[9] Alam, S., Reasat, T., Doha, R.M., Humayun, A.I.: Numtadb-assembled bengali handwritten digits. arXiv preprint arXiv:1806.02452 (2018)

[10] Rabby, A.K.M.S.A., Haque, S., Islam, M.S., Abujar, S., Hossain, S.A.: Ekush: A multipurpose and multitype comprehensive database for online off-line bangla handwritten characters. In: Santosh, K.C., Hegadi, R.S. (eds.) Recent Trends in Image Processing and Pattern Recognition, pp. 149–158. Springer, Singapore (2019)

[11] Nano, X.: Handwritten math symbols dataset (2017). https://www.kaggle.com/datasets/xainano/handwrittenmathsymbols

[12] Thapa, S.: Handwritten math symbols (2021). https://www.kaggle.com/datasets/sagyamthapa/handwritten-math-symbols

[13] Memon, J., Sami, M., Khan, R.A., Uddin, M.: Handwritten optical character recognition (ocr): A comprehensive systematic literature review (slr). IEEE Access **8**, 142642–142668 (2020)

[14] Islam, N., Islam, Z., Noor, N.: A survey on

optical character recognition system. arXiv preprint arXiv:1710.05703 (2017)

[15] Majumder, M.R., Mahmud, B.U., Jahan, B., Alam, M.: Offline optical character recognition (ocr) method: An effective method for scanned documents. In: 2019 22nd International Conference on Computer and Information Technology (ICCIT), pp. 1–5 (2019). IEEE

[16] Wu, Y.-C., Yin, F., Liu, C.-L.: Improving handwritten chinese text recognition using neural network language models and convolutional neural network shape models. Pattern Recognition **65**, 251–264 (2017)

[17] Islam, M., Shuvo, S.A., Nipun, M.S., Sulaiman, R.B., Nayeem, J., Haque, Z., Shaikh, M.M., Sourav, M.S.U.: Efficient approach of using cnn based pretrained model in bangla handwritten digit recognition. arXiv preprint arXiv:2209.13005 (2022)

[18] Sharma, A., Bhardwaj, H., Bhardwaj, A., Sakalle, A., Acharya, D., Ibrahim, W.: A machine learning and deep learning approach for recognizing handwritten digits. Computational Intelligence and Neuroscience **2022** (2022)

[19] Cohen, G., Afshar, S., Tapson, J., van Schaik, A.: Emnist: Extending mnist to handwritten letters. In: 2017 International Joint Conference on Neural Networks (IJCNN), pp. 2921–2926 (2017). https://doi.org/10.1109/IJCNN.2017.7966217

[20] Ahammad, K., Shawon, J.A.B., Chakraborty, P., Islam, M.J., Islam, S.: Recognizing bengali sign language gestures for digits in real time using convolutional neural network. Int J Comput Sci Information Security (IJCSIS) **19**(1) (2021)

[21] D'souza, L., Mascarenhas, M.: Offline handwritten mathematical expression recognition using convolutional neural network. In: 2018 International Conference on Information, Communication, Engineering and Technology (ICICET), pp. 1–3 (2018). IEEE

[22] Hossain, M.B., Naznin, F., Joarder, Y., Islam, M.Z., Uddin, M.J.: Recognition and solution for handwritten equation using convolutional neural network. In: 2018 Joint 7th International Conference on Informatics, Electronics & Vision (ICIEV) and 2018 2nd International Conference on Imaging, Vision & Pattern Recognition (icIVPR), pp. 250–255 (2018). IEEE

[23] Shinde, R., Dherange, O., Gavhane, R., Koul, H., Patil, N.: Handwritten mathematical equation solver. International Journal of Engineering Applied Sciences and Technology (IJEAST) **6**(10), 146–149 (2022)

[24] Huda, H., Fahad, M.A.I., Islam, M., Das, A.K.: Bangla handwritten character and digit recognition using deep convolutional neural network on augmented dataset and its applications. In: 2022 16th International Conference on Ubiquitous Information Management and Communication (IMCOM), pp. 1–7 (2022). IEEE

[25] Biswas, M., Islam, R., Shom, G.K., Shopon, M., Mohammed, N., Momen, S., Abedin, A.: Banglalekha-isolated: A multi-purpose comprehensive dataset of handwritten bangla isolated characters. Data in brief **12**, 103–107 (2017)

[26] Albawi, S., Mohammed, T.A., Al-Zawi, S.: Understanding of a convolutional neural network. In: 2017 International Conference on Engineering and Technology (ICET), pp. 1–6 (2017). Ieee

[27] Gu, J., Wang, Z., Kuen, J., Ma, L., Shahroudy, A., Shuai, B., Liu, T., Wang, X., Wang, G., Cai, J., *et al.*: Recent advances in convolutional neural networks. Pattern recognition **77**, 354–377 (2018)

[28] Chandola, Y., Virmani, J., Bhadauria, H.S., Kumar, P.: Chapter 4 - end-to-end pretrained cnn-based computer-aided classification system design for chest radiographs. Primers in Biomedical Imaging Devices and Systems, pp. 117–140. Academic Press (2021). https://doi.org/10.1016/B978-0-323-90184-0.00011-4

[29] Allena Venkata Sai Abhishek, D.V.R.G.: Resnet18 model with sequential layer for computing accuracy on image classification dataset. International Journal of Creative Research Thoughts **10** (May 2022)

[30] Ramzan, F., Khan, M.U., Rehmat, A., Iqbal, S., Saba, T., Rehman, A., Mehmood, Z.: A deep learning approach for automated diagnosis and multi-class classification of alzheimer's disease stages using resting-state fmri and residual neural networks. Journal of Medical Systems **44** (2019). https://doi.org/10.1007/s10916-019-1475-2

[31] He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 770–778 (2016)