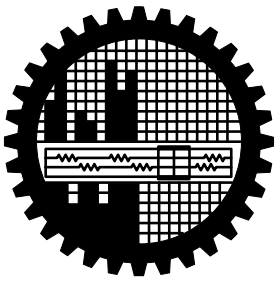


CSE 472
Machine Learning Sessional

Report on Assignment 4
Bangla Character Recognition Challenge

Asif Ajrof
Student ID: 1705092

Department of Computer Science and Engineering,
Bangladesh University of Engineering and Technology.

1 Introduction

In this assignment, we implemented a Convolutional Neural Network (CNN) to perform image classification.

1.1 Components of the CNN

The implementation of the CNN in this assignment consists of six key components, as described below:

1. Convolution: The convolutional layer is responsible for detecting features and patterns in the input image. Four parameters are used:
   (a) Number of output channels
   (b) Filter dimension
   (c) Stride
   (d) Padding
2. ReLU Activation: The Rectified Linear Unit (ReLU) activation function is applied to add non-linearity to the network, allowing it to model complex relationships between input and output.
3. Max-pooling: The max-pooling layer is used to down-sample the feature maps generated by the convolutional layer. This layer is configured with two parameters:
   (a) Filter dimension
   (b) Stride
4. Flattening: The flattening layer transforms the feature map into a one-dimensional column vector, which is then fed into the fully-connected layer.
5. Fully-Connected Layer: The fully-connected layer, also known as a dense layer, is responsible for classifying the input image. This layer is configured with one parameter:
   (a) Output dimension
6. Softmax: The final component of the network is the softmax function, which converts the output of the fully-connected layer into normalized probabilities, indicating the likelihood of each class.

1.2 The Dataset

We trained our model on the **NUMTA Handwritten Bengali Digits** dataset[1]. The data description and additional information can be found at the following Kaggle link: `https://www.kaggle.com/competitions/numta/data`.

We combined the *training-a*, *training-b*, and *training-c* datasets to form the training and validation set. And we used the *training-d* dataset as the independent test set.

2 Data Preprocessing

Before training our Convolutional Neural Network (CNN), it was necessary to preprocess the **NUMTA Handwritten Bengali Digits** dataset. The following steps were performed to prepare the data for analysis:

- Inversion: The images were inverted as they were originally black ink on a white paper background.
- Erosion and Dilation: The images went through morphological operations such as erosion and dilation to remove any noise and enhance the visibility of the digit patterns.
- Greyscale Conversion: The images were converted to greyscale in order to reduce the dimensionality of the data, as the color values hold little significance in digit recognition. Additionally, converting the images to greyscale made it easier to apply thresholding.
- Thresholding: The greyscale images were thresholded to convert them to binary images, making it easier for the CNN to identify the digit patterns.
- Resizing: The images were resized to a standard size to ensure consistent input to the CNN.
- Normalization: The images were normalized to ensure that their pixel values fall within a specific range, making it easier for the CNN to process and learn from the data.

After evaluating various preprocessing techniques, the better performing approach (see Section 4.1) was chosen to transform the **NUMTA Handwritten Bengali Digits** dataset into a clean and standardized form (see Figure 1).

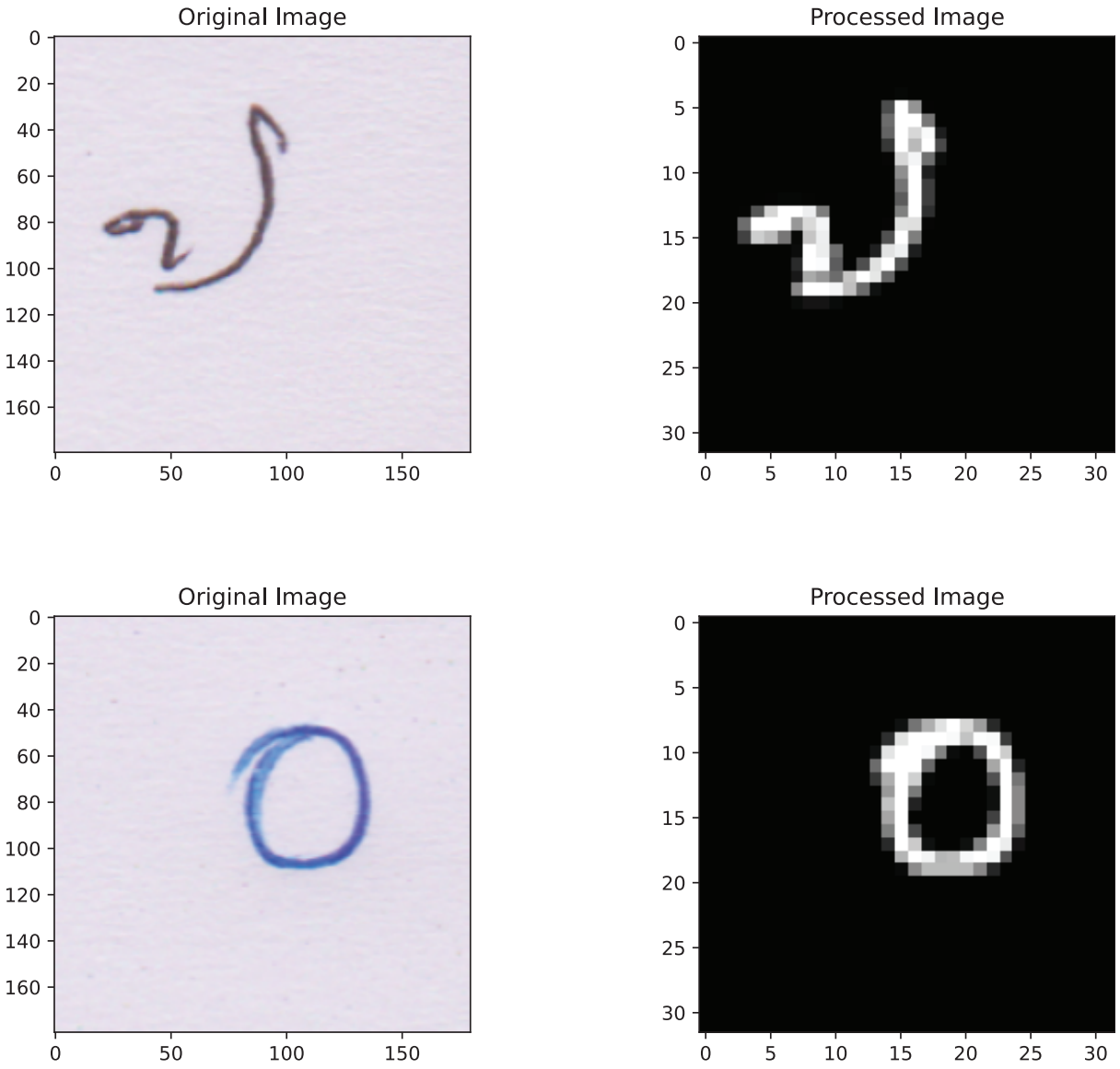


Figure 1: Example of image preprocessing

3 Model Architecture

Due to time constraints, our exploration of various neural network architectures was limited. We have not included the performance results of the custom architectures that we experimented with due to unsatisfactory results.

Ultimately, we selected the simple LeNet-5 [2] model as our final choice.

3.1 Description

Input	Image dimension 32×32 (RGB or Greyscale)
Convolution	Number of output channels 6, Filter dimension 5×5, Stride 1, Padding 0
ReLU Activation	
Max-pooling	Filter dimension 2×2, Stride 2
Convolution	Number of output channels 16, Filter dimension 5×5, Stride 1, Padding 0
ReLU Activation	
Max-pooling	Filter dimension 2×2, Stride 2
Flattening	
Fully-Connected Layer	Output dimension 120
ReLU Activation	
Fully-Connected Layer	Output dimension 84
ReLU Activation	
Fully-Connected Layer	Output dimension 10
Softmax	

4 Results

4.1 Preprocessing

Experiment environemnt

Training Data source	*training-a*, *training-b*, and *training-c*
Training Data count	$19702 + 359 + 24298 = 44359$ (Full)
Train-Validation split	$8 : 2$
Testing Data source	*training-d*
Testing Data count	10908
Preprocessing	varying
Batch size	64
Epoch	10
Learning rate	0.001

Preprocessing 1	Inversion, Resizing (28×28), Normalization
Preprocessing 2	Inversion, Erosion, Dilation, Greyscale Conversion, Thresholding, Resizing (32×32), Normalization
Preprocessing 3	Greyscale Conversion, Inversion, Dilation, Thresholding, Resizing (32×32), Normalization

We chose Preprocessing 3 as it is the better performing method.

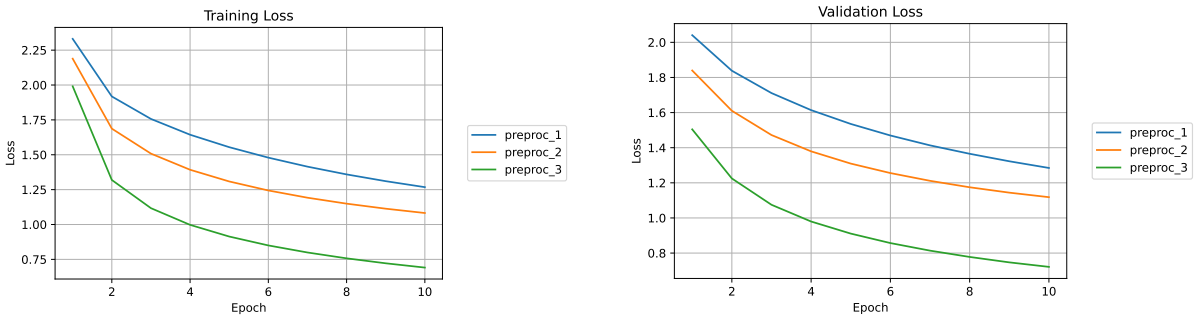


Figure 2: Training-Validation loss varying preprocessing

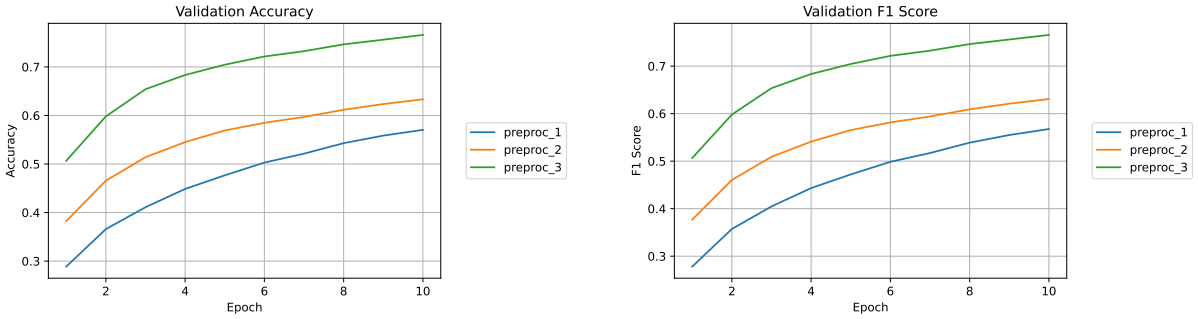


Figure 3: Training-Validation scores varying preprocessing

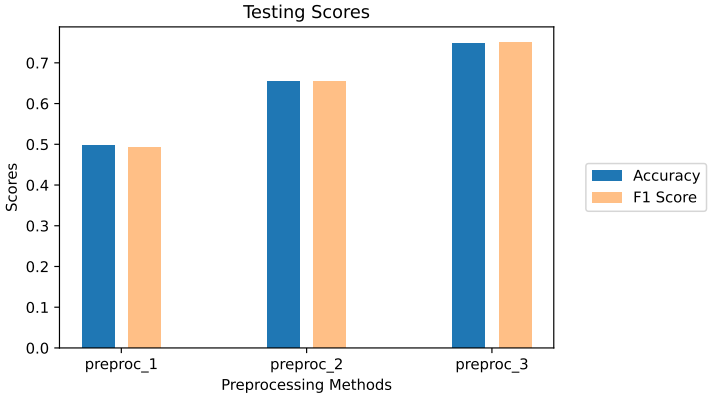
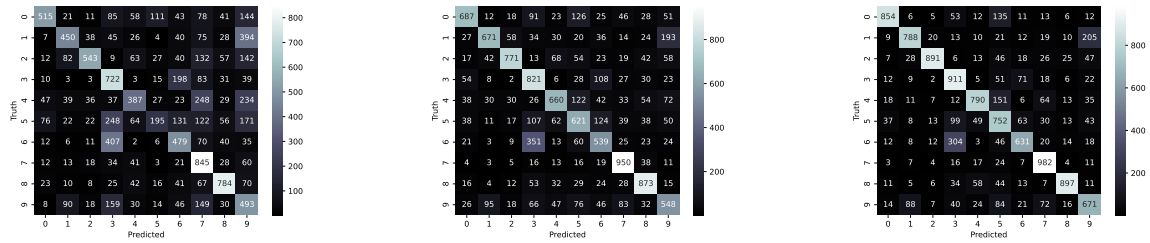


Figure 4: Testing scores varying preprocessing



(a) Preprocessing 1 (b) Preprocessing 2 (c) Preprocessing 3

Figure 5: Testing Confusion Matrix varying preprocessing

4.2 Learning Rate

Experiment environemnt

Training Data source	*training-a*, *training-b*, and *training-c*
Training Data count	10000
Train-Validation split	8 : 2
Batch size	128
Epoch	30
Learning rate	varying

We chose 0.001 as the Learning Rate based on the performances.

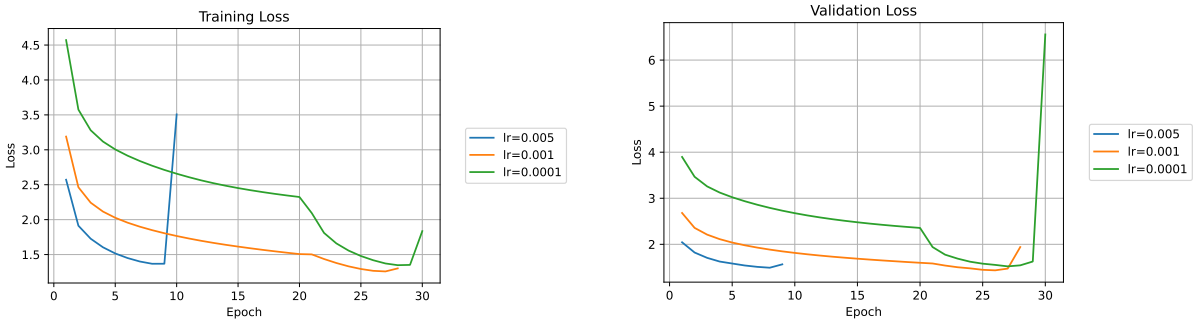


Figure 6: Training-Validation loss varying learning rate

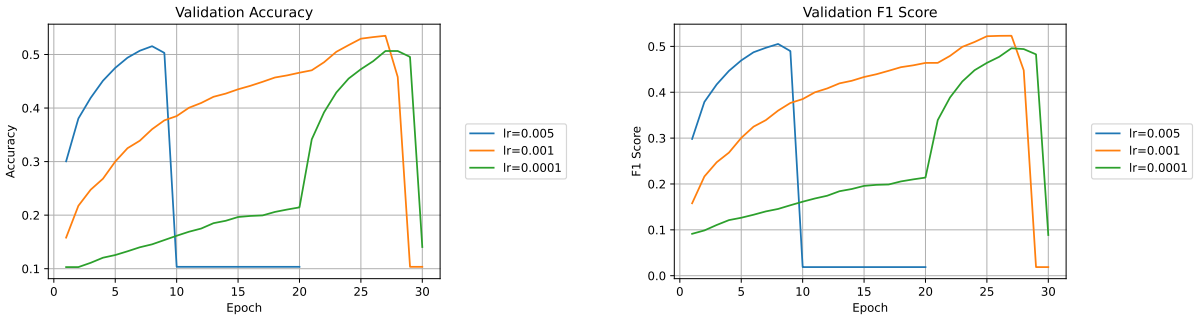


Figure 7: Training-Validation scores varying learning rate

4.3 The Final Model

Training Data source	*training-a*, *training-b*, and *training-c*
Training Data count	$19702 + 359 + 24298 = 44359$ (Full)
Train-Validation split	$8 : 2$
Testing Data source	*training-d*
Testing Data count	10908
Preprocessing	Preprocessing 3
Batch size	64 (higher batch size leads to a little better time performance but lower scores)
Epoch	10 (higher epoch overfits on the training data)
Learning rate	0.001

Independent Test Performance

Accuracy	74.87165383204987%
F1 Score	75.0704863697553%

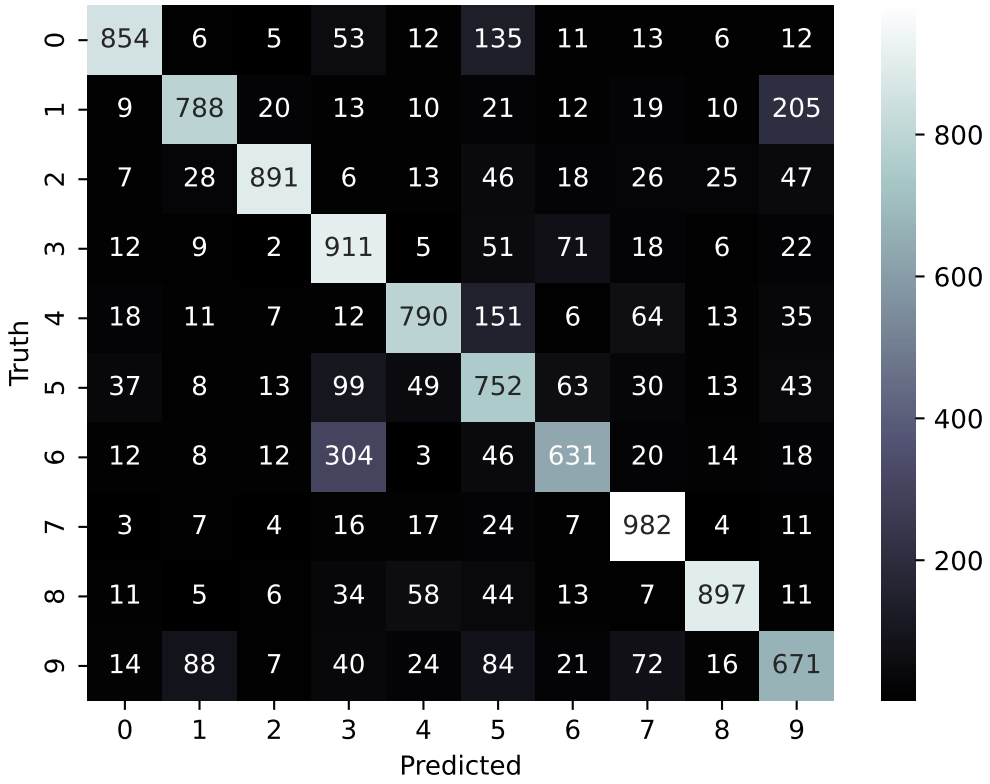


Figure 8: Confusion Matrix of Final Model on Testing Data

5 Discussion

Developing a Convolutional Neural Network from scratch proved to be a challenging yet fulfilling experience for us. Had we been able to experiment with more architectures, it is likely that the performance would have improved.

The test results reveal that the model frequently misinterprets the Bangla digit ৬ as ৩ and ১ as ৯, due to their similar pattern. Furthermore, the handwriting in the dataset can sometimes be too complex even for human comprehension, which may have contributed to suboptimal performance.

References

[1] Samiul Alam, Tahsin Reasat, Rashed Mohammad Doha, and Ahmed Imtiaz Humayun. Numtadb-assembled bengali handwritten digits. *arXiv preprint arXiv:1806.02452*, 2018.

[2] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998. doi: 10.1109/5.726791.