A Deep Convolutional Neural Network for Bangla Handwritten Numeral Recognition with Data Augmentation

Umnoon Binta Ali 1713013042
Department of Electrical and Computer Engineering
North South University
Dhaka, Bangladesh
umnoon.ali@northsouth.edu

***Abstract*—Bangla is one of the major languages in the world. Recently a lot of attention has been given to OCR Bangla but the challenges exist due to lack of augmented data. This paper studies the most unbiased and augmented dataset NumtaDB to classify images of isolated Bangla numerals. We preprocessed the data, created synthetic data augmentations using keras library and feed the data to our training model in the pipeline stage. Furthermore, we used popular CNN network ResNet34 to better the performance of the dataset.**

***Index Terms*— Bangla digit, NumtaDB, CNN, synthetic images, augmentation, keras libraries, ResNet34.**

I. INTRODUCTION

Bengali is the fifth most-spoken native language and the sixth most spoken language by total number of speakers in the world with approximately 228 million native speakers and another 37 million as second language speakers [1]. Different people write numerals differently. Such variance often makes the identifying procedure harder due to additional curvature of the Bangla digits. In the areas of computer vision, handwritten digit recognition is important for automatic fields like OCR, license plate recognition etc. For this purpose, many research has been done before on dataset like CMATERDB, ISI, Ekush. But these datasets were biased contained ambiguous digits. The dataset that we chose is NumtaDB which is free from bias [2]. In this paper, we mainly focus on recognizing Bangla digits from a number of scanned or camera captured digital images. Firstly, we create augmentations of the train sets and with that our CNN model is trained. Then we preprocess the test images to fit into our model. And ultimately using transfer learning, ResNet34, we try to better the performance on our dataset.

II. LITERATURE REVIEW

Recent advances in machine learning, especially to the emergence of deep neural networks, have yielded promising results when combined with the power of large data. Detecting characters and numbers for example can be substantially improved with deep learning. Many researches have been done on NumtaDB so far in terms of preprocessing of the image and building up deep learning models like CNN [3]. Preprocessing includes resizing the image, converting the images to grayscale, smoothing for noise removal, filtering out the quadrilateral black spots in the testing set, thresholding the images with a value of 127, cropping, contouring etc. [3]. Bayanno_Net, which has used 10000 images from NumtaDB dataset to train a CNN model that achieved great accuracy [4]. Unconventional transfer learning approaches like VGG16 with accuracy of 97.09% [5], ResNet34 and ResNet50 with accuracy of 99.33% [6], Densenet121 with accuracy 99.46% on augmented dataset and 96.27% on non-augmented dataset have great performance on our NumtaDB dataset. [6] Comparative studies on both augmented and non-augmented dataset using the keras library has inspired us to evaluate our learnings on the dataset NumtaDB.

III. DATASET

Our dataset contains more than 85,000+ Bangla handwritten digit images. It is a standard, unprocessed and reviewed, all real-world data which has been assembled from six different sources. According to NumtaDB, the sources are labelled from 'a' to 'f', both for training and testing. Each image of this dataset is about 180x180 pixels. Some examples from the dataset are shown below.

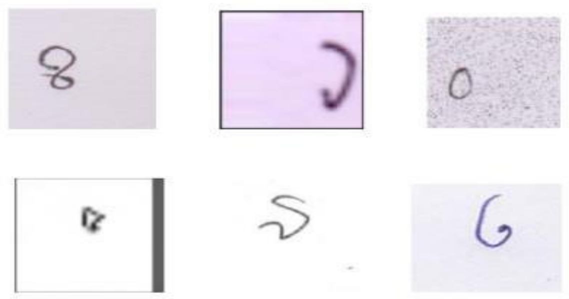


Fig. 1. Dataset sample

In this dataset, the training images do not have many augmentations. So, in this paper we will create synthetic mages which will then be given to our model while training for better performance.

IV. PROPOSED APPROACH

Our approach can be jotted down in four steps. First is creation of data augmentation before training, second formulating deep convolutional neural network, third is preprocessing our test dataset to fit into our model and fourth is comparing our result with a pre-trained model ResNet34.

A. DATA AUGMENTATION

We created data augmentation using keras library. It is because NumtaDB dataset doesn't have many augmentations for training. But to form an OCR, we need more real-world samples which generally will be augmented. For augmentation we created horizontal shift, vertical shift, rotation, zoom, horizontal flip and rescaling. An instance of our augmented dataset in shown here.

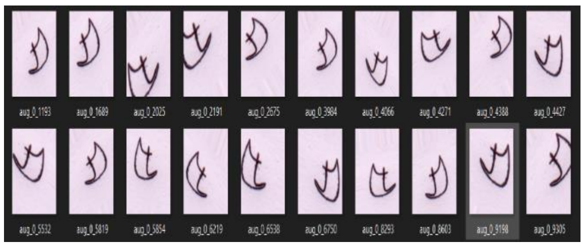


Fig. 2. Data augmentation Example

B. ARCHITECTURE OF MODEL

Our architecture 6 convolutional layers and 2 fully connected dense layer. The first two layers have 32 filters and each filter size is 5×5. The middle two layers have 128 filters and each filter size is 3×3. The last two layers have 256 filters and each filter size is 3×3. RELU. Max pooling 2x2 and batch normalization has been done after every two layers. Among the fully connected layers, the last one has 10 classes which denotes for 10 digits in Bangla (0-9). This architecture is the same as an existing architecture [8]. Our aim is to see how it performs on augmented data on training. Since we are feeding synthetic images during the pipeline stage, 0.2 fraction on those synthetic images will be reserved for validation and the rest of the images are for training. For testing of our model, we use the testing images present in the NumtaDB dataset.

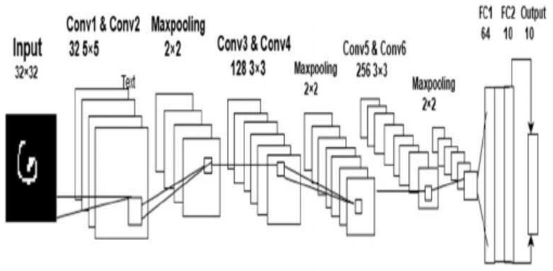


Fig. 3. CNN model

V. PREPROCESSING OF IMAGES

A. Resizing and Gray scaling

The original size of NumtaDB images is 180×180 pixels which are reduced to 32×32 pixels. We also convert all RGB images to GRAY scale images. The color channel is converted to 1 channel from 3 channel.

B. Interpolation

When images are resized, they can lose a lot of detail. For image decimation, inter-area interpolation is the preferred process. The pixel area relation is used to resample in this process. After resizing files, we use inter-area interpolation.

C. Removing Blur

We first add blur with Gaussian blur, then remove the blurred image from the original image. The image is then de-blurred by adding a weighted portion of the mask.

D. *Sharpening Images*

There are a variety of sharpening filters available. The Laplacian filter is used in this paper. Our filer is a three-dimensional matrix.

E. *Salt and Pepper Noise Removed*

We delete salt and pepper noise from NumtaDB images. To remove salt and pepper noise, we use the median filter. The photos become transparent, sharp, and salt and pepper noiseless after preprocessing.

VI. PRETRAINED MODEL

We used transfer learning for our project. We used the pre-trained model Resnet34 with 10 epochs. ResNet is a form of convolutional neural network (CNN). One of the problems ResNet solve is the famous known vanishing gradient. This is because when the network is too deep, the gradients from where the loss function is calculated easily shrink to zero after several applications of the chain rule. This result on the weights never updating its values and therefore, no learning is being performed. With ResNet, the gradients can flow directly through the skip connections backwards from later layers to initial filters. The 34 in ResNet34 indicates 34 convolutional layers in the ResNet model. First layer performs a 7x7 convolution. Each of the next layers follow the same pattern. They perform 3x3 convolution with a fixed feature map dimension (F) [64, 128, 256, 512] respectively, bypassing the input every 2 convolutions. Furthermore, the width (W) and height (H) dimensions remain constant during the entire layer. It follows by a fully connected (FC) layer and the final output.

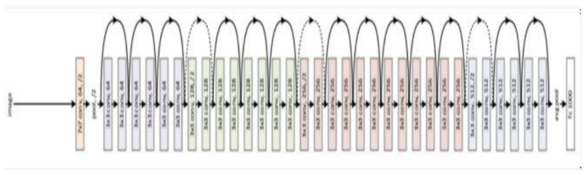


Fig. 4. Resnet34 model

We used about 6000 data from the NumtaDB dataset. The data was split into 80%, 10%, 10% ratio for test, train and validation accordingly. Our images were preprocessed. At first, we resized the images to 64*64 using the OpenCV library to explore and view the images. Then we used TorchVision library to further transform the images. We cropped the images to size 256. We randomly rotated the image by 15 degrees.

Some of the images were horizontally flipped. We also used normalization with some mean and std values. Finally, we trained the model with the pre-processed images.

VII. RESULT ANALYSIS

In our augmented dataset, we got 9.70% accuracy on the kaggle test set after 100 epochs even though it performed really well with our validation set with around 99.73% accuracy. The problem could be on the preprocessing label or in the model itself. For our ResNet34 model we have achieved 98% accuracy after just 10 epochs, which is far better.

We can see the plotted graph of loss function vs epoch number for our ResNet34 model.

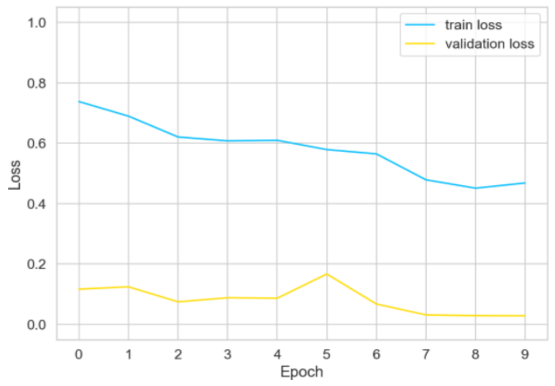
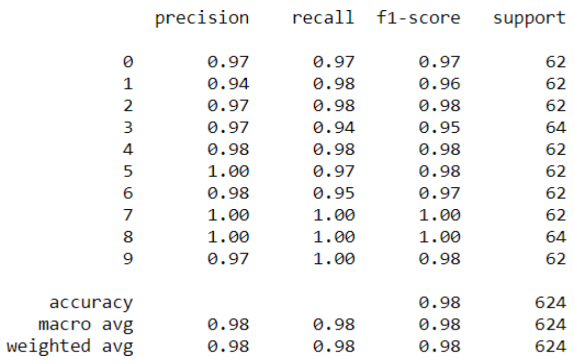


Fig. 5. Loss vs Epoch graph

We can also see the precision, recall, f1 score and overall accuracy from the chart below.



```
              precision    recall  f1-score   support

           0       0.97      0.97      0.97        62
           1       0.94      0.98      0.96        62
           2       0.97      0.98      0.98        62
           3       0.97      0.94      0.95        64
           4       0.98      0.98      0.98        62
           5       1.00      0.97      0.98        62
           6       0.98      0.95      0.97        62
           7       1.00      1.00      1.00        62
           8       1.00      1.00      1.00        64
           9       0.97      1.00      0.98        62

    accuracy                           0.98       624
   macro avg       0.98      0.98      0.98       624
weighted avg       0.98      0.98      0.98       624
```

Fig. 6. Accuracy

We can also see the predicted digit and true digit ratio from the confusion matrix given below.

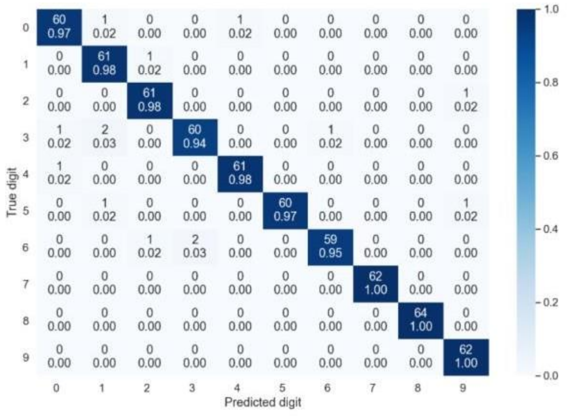


Fig. 7. Confusion Matrix

We can predict that the accuracy will increase nearly 100% if we work with the whole dataset. Between these two approaches, we can conclude that, by transfer learning we get better performance on the given dataset NumtaDB.

VIII. CONCLUSION

We were unable to evaluate large deep neural network architectures such as ResNet34 due to a lack of computer resources. We may investigate this further in the future. These topologies improve the performance of neural network models. Even after the creators' thorough scrutiny, some of the NumtaDB's data was incorrectly labeled. Again, due to time limitation, we could not evaluate our produced model and perform more experimentations to better the result.

REFERENCES

[1] https://en.ikipedia.org/wiki/Bengali_language

[2] S. Alam, T. Reasat, R.M. Doha, and A.I Humayun, “NumtaDB-Assenbled Bengali handwritten digits”, arXiv:1806.02452 [cs.CV], 6 June, 2018.

[3] O. Paul, "Image Pre-processing on NumtaDB for Bengali Handwritten Digit Recognition," 2018 International Conference on Bangla Speech and Language Processing (ICBSLP), 2018, pp. 1-6, doi: 10.1109/ICBSLP.2018.8554910.

[4] M. S. Islam, M. F. A. Foysal and S. R. H. Noori, "Bayanno-Net: Bangla Handwritten Digit Recognition using Convolutional Neural Networks," 2019 IEEE Region 10 Symposium (TENSYMP), 2019, pp. 23-27, doi: 10.1109/TENSYMP46218.2019.8971167.

[5] H. Zunair, N. Mohammed and S. Momen, "Unconventional Wisdom: A New Transfer Learning Approach Applied to Bengali Numeral Classification," 2018 International Conference on Bangla Speech and Language Processing (ICBSLP), 2018, pp. 1-6, doi: 10.1109/ICBSLP.2018.8554435.

[6] M. Mahmudul Hasan, M. Rafid UI Islam and M. Tareq Mahmood, "Recognition of Bengali Handwritten Digits Using Convolutional Neural Network Architectures," 2018 International Conference on Bangla Speech and Language Processing (ICBSLP), 2018, pp. 1-6, doi: 10.1109/ICBSLP.2018.8554753.

[7] M. A. Al Nasim, R. E. Ferdous, M. A. Haque Pantho and A. Islam Chowdhury, "A Comparative Analysis on Bangla Handwritten Digit Recognition with Data Augmentation and Non-Augmentation Process," 2020 International Congress on Human-Computer Interaction, Optimization and Robotic Applications (HORA), 2020, pp. 1-5, doi: 10.1109/HORA49412.2020.9152905.

[8] A. Shawon, M. Jamil-Ur Rahman, F. Mahmud and M. M. Arefin Zaman, "Bangla Handwritten Digit Recognition Using Deep CNN for Large and Unbiased Dataset," 2018 International Conference on Bangla Speech and Language Processing (ICBSLP), 2018, pp. 1-6, doi: 10.1109/ICBSLP.2018.8554900.