Bangla Handwritten Character Recognition With Multilayer Convolutional Inception Module

Somania Nur Mahal, *student, BRAC University and* B M Abir *student, BRAC University*

Abstract— **Handwritten character detection and recognition from a natural image has a large set of difficulties. A systematic approach that can automatically recognize character from handwriting, printed books, road signs and also classifies text and non-text blocks from natural image has many significant applications. For instance, visual assistance for visually impaired people, image understanding, classification of text in image, implementing autonomous navigation system. Recent development of deep learning approach has strong capabilities to extract high level feature from a kernel (patch) of an Image. In this paper we will demonstrate a novel approach that integrates a multilayer convolutional neural network (CNN) with supervised feature learning. Unlike previous OCR methods, this CNN based approach does not focuses on some specific feature extraction. 166,105 Bangla handwritten character of different shapes and strokes has been used to train and evaluate the performance of the model. This approach allows a higher recall rate for the character in an image and outperforms some current methodologies.**

Index Terms—**Bangla, Handwritten Character, Neural Network, Inception Module**

I. INTRODUCTION

RECOGNIZING handwritten text from document and scene-text have received a lot of attention as it allows any system using it to be robust and the use of such models in variable real world Scenarios.

In current stage of our world, information is considered to be one of the most significant driving force for our social change. And language is the vessel through which information flows. But it seems there is a clear division between digital and printed and handwritten text that surrounds us. To bridge that gap, computer vision researchers have gone to great lengths such as creating mathematical formulas and using the techniques of hand-engineered features, where the deep & detailed knowledge of the feature is incorporated into models to process the image. So that they can detect and recognize characters, numbers and even symbols. But the formulation and experimentation of each and every filters and feature for each and every characters, and to take account for other various factors, is both hard and a tedious task. But even then, the accuracy of such algorithms like used in common OCR system is not up to the standard of human level accuracy. They are clearly focused on document text detection. Those systems are also plagued by different lighting conditions of the images and various background noise that makes the text recognition systems to underperform [1].

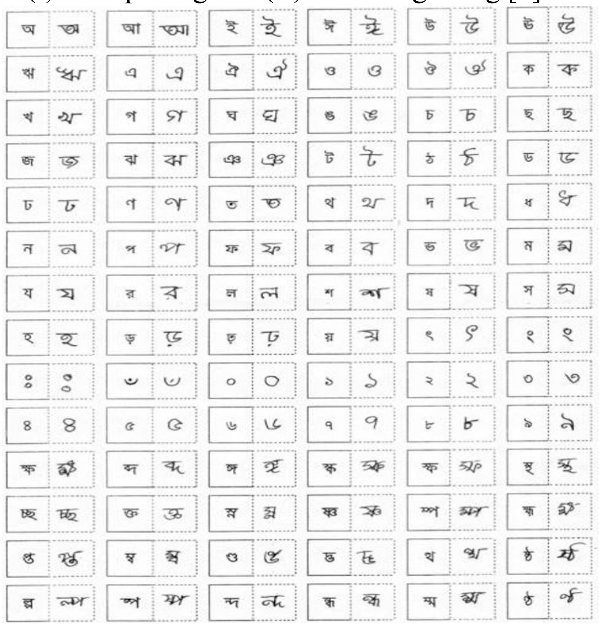
In recent times, the field of computer vision has seen great strides towards expert systems that can be trained using large image dataset due to the rise of high performance computing (HPC) systems. High performing parallel processing GPUs has enabled the computer vision researchers to turn the problem of image recognition into a problem of numerical optimizations problem. So now there exists a wide range of algorithms and architecture to tackle such problems. One such architecture to create models and solve those problems are neural networks. It has been designed on the basis such that it imitates some function of the neural pathways of human brain. And to maximize the performance of such architecture, numerous methodologies have been proposed and one such implementation of a system of detector and recognizer that uses machine learning techniques is convolutional neural network or CNN [2], [3],[4]. In CNN, the state of the art detection performance is 80% on F-measure with ICDAR 2011[4]. And for end-to-end recognition, the result is 57% accurate on SVT dataset [5]. The above mentioned works clearly focuses on detection and recognition of English alphabets.

In this paper we will be showing the results of experimentation on different types of convolutional neural network architecture which will be appropriately evaluated on the context of Bangla alphabet datasets like BanglaLekha-Isolated [6]. By analyzing this dataset we will establish a base ground truth for a superior CNN architecture that can be used in handwritten text recognition in Bangla. And ultimately with the chosen CNN architecture we will implement a Bangla character recognizer that will be able to localize and recognize text from any background image. With the experimentation results we determine which architecture is performing better for our current problem.

In order to improve the accuracy of a CNN-based architecture it is imperative to fine tune the hyper-parameters according to the specific needs of the problem domain. And using the power of hyper-parameters is one of the most powerful advantages the neural network architecture, making any problem set to be solved. This general purpose usage of neural networks makes it so powerful that a complex problems in any field can now be solved with ease. And one of the most popular CNN based architecture is called Inception or GooLeNet [7]. This general purpose architecture contains blocks of CNN-based module called Inception Module. We will use modified version of Inception architecture. Using combination of Inception module and CNN layers. Ultimately we will demonstrate the improvement in accuracy for Bangla Handwritten characters with convolutional.

II. Background Study and Related Works

End-to-end text recognition in natural images has two key components (I) text spotting and (II) text recognizing [3]. The main objective is to locate the specific text oriented regions then recognize the actual character. Many research and work have been done for solving the character recognition problem [3]. Much approach and methods are implemented and showed higher performance. However in a complex image locating and recognizing character still remains a problem. A wide range of approaches are used for text segmentation and recognition. In an end-to-end system text detection system identify the text region which is the input of character or word recognizer [3].Then the recognizer recognizes each character in the word or the whole word. Deep convolutional neural network models has higher performance rate for recognizing and computing high level deep features [1]. LeNet a traditional CNN model, powerful for digit and hand-written character recognition [1[42,43]].In [7] a CNN based approach BHCR-CNN proposed for recognizing Bangla character. Where CNN is employed to classify individual character. In [8] a deep belief network is introduced to recognize Bangla numerals and alphabet. Where an unsupervised feature learning followed by a supervised fine tuning of the network parameters is used.


(a)

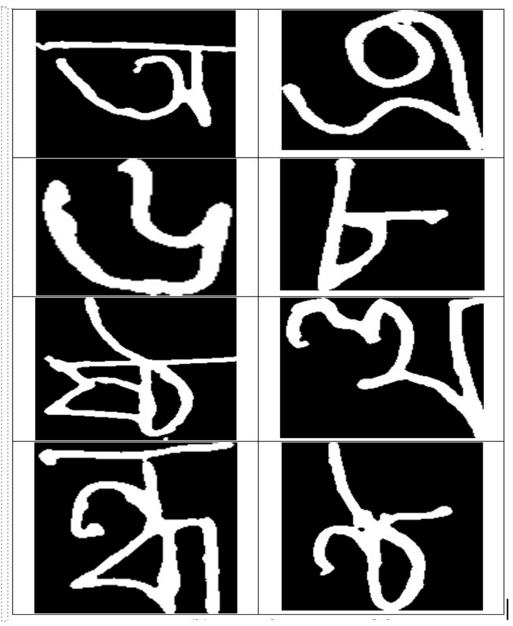

(b)

Figure 1 : Sample figure of Bangla character :(a) handwritten character of a particular age group and geographic location (b) Sample images of dataset

III. Handwritten Bangla character recognition using CNN and inception module

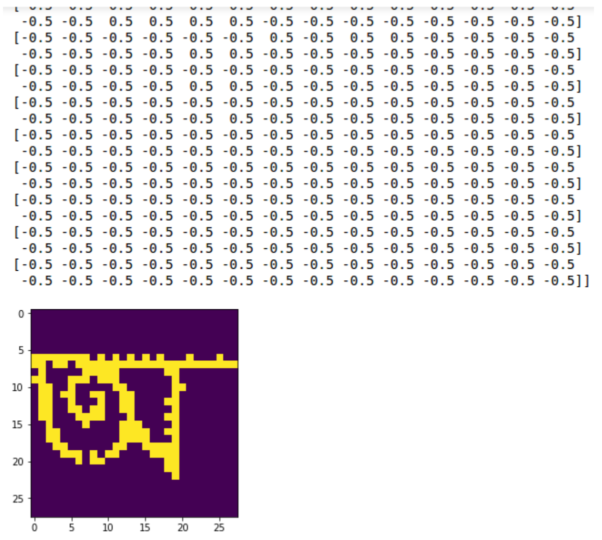
Main objective of our work is to build a Bangla character recognizer which can automatically recognize word or character from any simple or complex image components. In this section our proposed multilayer inception based convolutional model explains in detail. Following subsection gives a brief idea of dataset, image preprocessing, system architecture and classifier.

A. Dataset

The text recognition subsystem is trained on character-level by using BanglaLekha-isolated [8] dataset. In BanglaLekha-isolated has total 1, 66,105 handwritten character images. It consists sample of 84 different Bangla handwritten numerals, basic characters and compound characters [9] which collected from different geographical location of Bangladesh and different age group. In fig. 1(a) collected from [8], a sample of Bangla handwritten character of a particular age is shown. Where first 11 characters are vowel followed by 39 consonant characters then 10 characters are Bangla numerals and rest 24 are Bangla compound character.

The dataset provides multiple labels per character/character group. It contains a wide variation of frequently used in Bangla compound sentences which are very complex shaped. In fig. 1(b) last two character are compound character.

B. Image Preprocessing



```
  -0.5 -0.5  0.5  0.5  0.5  0.5 -0.5 -0.5 -0.5 -0.5 -0.5 -0.5 -0.5 -0.5]
 [-0.5 -0.5 -0.5 -0.5 -0.5 -0.5  0.5 -0.5  0.5  0.5 -0.5 -0.5 -0.5 -0.5
  -0.5 -0.5 -0.5 -0.5  0.5  0.5 -0.5 -0.5 -0.5 -0.5 -0.5 -0.5 -0.5 -0.5]
 [-0.5 -0.5 -0.5 -0.5 -0.5 -0.5 -0.5 -0.5 -0.5 -0.5 -0.5 -0.5 -0.5 -0.5
  -0.5 -0.5 -0.5 -0.5  0.5  0.5 -0.5 -0.5 -0.5 -0.5 -0.5 -0.5 -0.5 -0.5]
 [-0.5 -0.5 -0.5 -0.5 -0.5 -0.5 -0.5 -0.5 -0.5 -0.5 -0.5 -0.5 -0.5 -0.5
  -0.5 -0.5 -0.5 -0.5 -0.5  0.5 -0.5 -0.5 -0.5 -0.5 -0.5 -0.5 -0.5 -0.5]
 [-0.5 -0.5 -0.5 -0.5 -0.5 -0.5 -0.5 -0.5 -0.5 -0.5 -0.5 -0.5 -0.5 -0.5
  -0.5 -0.5 -0.5 -0.5 -0.5 -0.5 -0.5 -0.5 -0.5 -0.5 -0.5 -0.5 -0.5 -0.5]
 [-0.5 -0.5 -0.5 -0.5 -0.5 -0.5 -0.5 -0.5 -0.5 -0.5 -0.5 -0.5 -0.5 -0.5
  -0.5 -0.5 -0.5 -0.5 -0.5 -0.5 -0.5 -0.5 -0.5 -0.5 -0.5 -0.5 -0.5 -0.5]
 [-0.5 -0.5 -0.5 -0.5 -0.5 -0.5 -0.5 -0.5 -0.5 -0.5 -0.5 -0.5 -0.5 -0.5
  -0.5 -0.5 -0.5 -0.5 -0.5 -0.5 -0.5 -0.5 -0.5 -0.5 -0.5 -0.5 -0.5 -0.5]
 [-0.5 -0.5 -0.5 -0.5 -0.5 -0.5 -0.5 -0.5 -0.5 -0.5 -0.5 -0.5 -0.5 -0.5
  -0.5 -0.5 -0.5 -0.5 -0.5 -0.5 -0.5 -0.5 -0.5 -0.5 -0.5 -0.5 -0.5 -0.5]
 [-0.5 -0.5 -0.5 -0.5 -0.5 -0.5 -0.5 -0.5 -0.5 -0.5 -0.5 -0.5 -0.5 -0.5
  -0.5 -0.5 -0.5 -0.5 -0.5 -0.5 -0.5 -0.5 -0.5 -0.5 -0.5 -0.5 -0.5 -0.5]]
```

Figure 2. Converting image into numPy array

As these data from dataset has different image size so it was a bit challenging to train them as we used fixed image size which is 28*28 pixel for each image. Therefore image pre-processing is needed for handling this big size of data. For resizing the images we used python imaging library PIL. Antialiasing filter is used so that quality of the image doesn't degrade while resizing. After setup the train environment we convert images into numPy array and save these images in separate class file. Then images of each class are shuffled to have random validation and training set. Also merge them into a single dataset. Duplicate data between training and test can skew the result. So, we need to remove all the overlap between training and test data. We measured the duplicates data and find 1081 samples overlap between valid and train data, 1278 samples overlap between test and train data, 185 samples overlap between test and valid data. Then we create a sanitized test and valid datasets. This data cleaning process gives better accuracy.

C. Model Architecture

Our system architecture consists of several layers of Convolutional Neural Network. As shown in fig.3, the pre-processed data with a size of 28x28 image is taken as an input into the 1st convolutional neural network layer with patch or kernel size of 3x3 and 'same' padding property. After Max Pooling, the output is again fed into a 2nd convolution layer with patch size of 5x5 and 'same' padding. Its output again fed into Max Pooling layer of same configuration as before. The max pooling from the previous layer is used as the input of the Inception module, where CNN is not only stacked sequentially but also parallel. Then output of the inception module is fed into a fully connected network where every single node is connected with each other. This layer flattens the high level feature outputs from the inception module and converts it into a non-linear combination of these features. After that, another fully connected layer is added to flatten the data again.

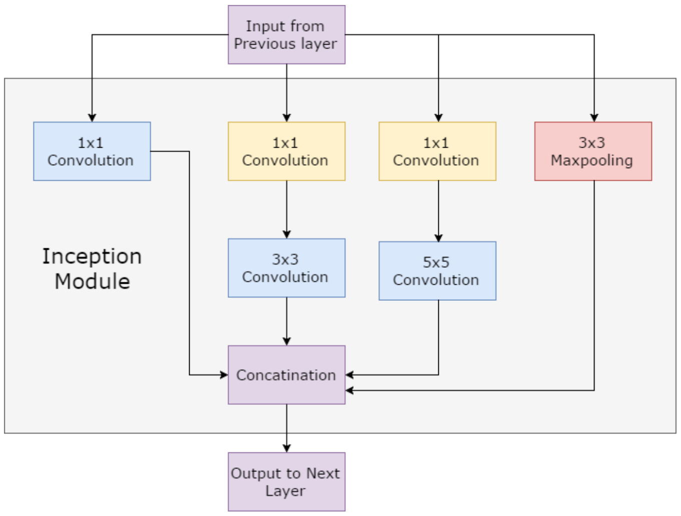


Figure 3: A detailed Inception System Architecture

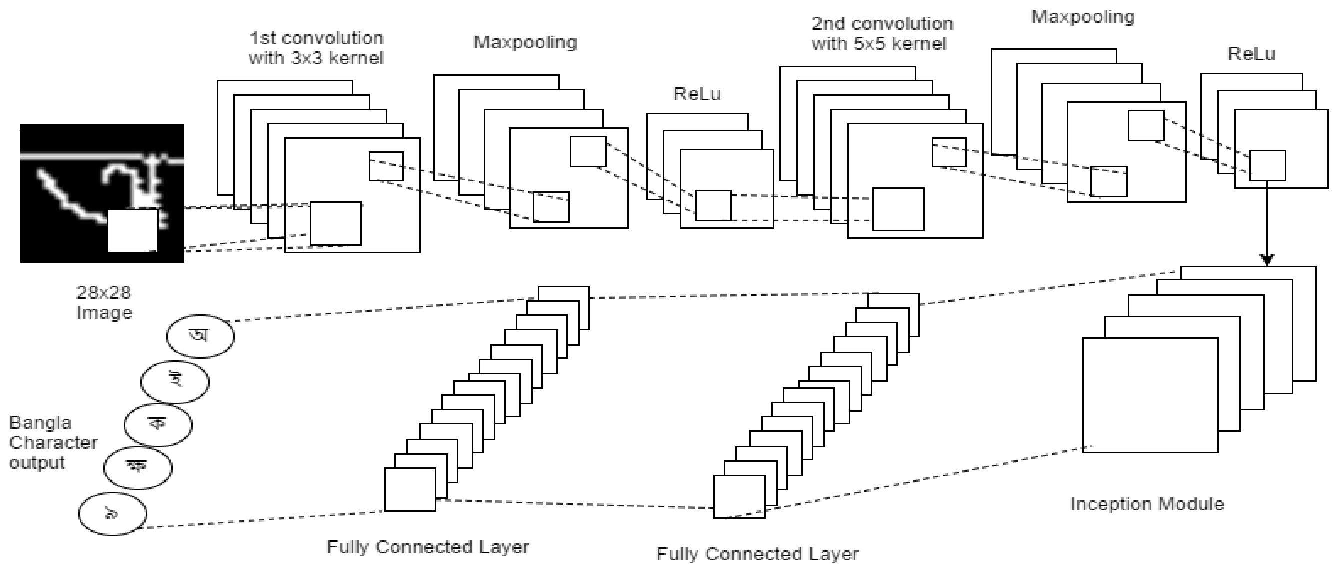


Figure 4: Data Flow between the proposed CNN Architecture for Bangla Character Recognition

D. *Classification Using CNN and Inception Module*

Classifying Bangla handwritten character has high – dimensional complexity which requires a multilayer, hierarchical neural network [7]. For visual recognition three principle factors are very important for instance, local receptive fields, weight sharing, and subsampling layer which makes a difference in CNN from other simple feed forward neural network[4].In our proposed CNN based approach we used two convolution layer followed by one inception module. Unlike other traditional MLP based method, here CNN itself capture local features from the input image by forcing each filter of CNN layer depends on spatially local patch of the previous layer. Therefore a feature map is generated in next layer. In CNN a great advantage of weight sharing is it significantly reduces the number of free parameter [4] by sharing the same set of edge weights across all features in the hidden layers. Subsampling or pooling layer is another powerful building block of CNN. The main objective of pooling layer is to reduce the dimensionality of the response map created by convolution layer [4]. And also allows translation invariance for instance, rotation, scale and other distortions into the model.

In our approach we used an inception module on top of convolutional layers. This module works as filter inputs of multiple convolution which works on same input. For instance, a 1x1, 3x3, and 5x5 convolution layer works on same input and also employ pooling layer at the same time. This module improves the performance of overall model by extracting multi-level feature.

In fig.4 shows structure of convolutional neural network used for Bangla character recognition. Two convolutional layer with 3x3 and 5x5 receptive fields along with two max pooling and ReLu layer. For instance, a 28x28 pixels image is input to the CNN. Applying 1st CNN layer on input produced first level feature maps. These feature maps are produced such a way that it can extract a variation of local features where distinct patches with different weights and biases from other patches are used. After that a max-pooling layer down-sample the input (output matrix from previous layer) representation and also reduce the dimensionality. Input 28x28 pixel image down-sampled to 14x14 feature maps after max-pool operation. The 2nd layer CNN and max-pooling operations are same as 1st layer convolution. Output of 2nd layer CNN is input of inception module. In this inception module 1x1, 3x3, 5x5convulations along with 3x3 max-pooling is used. Here, 1x1 convolution reduces the dimensionality of the input to large convolutions. Therefore it makes the computations reasonable. Filters from all layers in inception module concatenated and converge the output with two fully connected layer. And a dropout activation function is employed to maximize the performance.

IV. Training Experiment Analysis And Results

A. *Architecture for training the model*

The model architecture sometimes also known as logits, takes some placeholder inputs such as weights and biases for each convolution and fully connected layer. The weights and biases are the main tuning parameter that is used to make a model learn from its training dataset. When the training session is first initiated, the weights are randomly generated with a standard deviation of 0.3 and biases with zero. Then the logits use these initial placeholder weights and biases to calculate a predicted value for the initial input of character image. With the Predicted output of the initial values from the logits, a SoftMax function is used to normalize the prediction output between 0.0 and 1.0. Now, with this prediction, we can compare it to the true label of the input data and update the value of weights and biases accordingly.

In order for our system to determine how much of the weights and biases value have to be changed in order to improve the accuracy of the prediction, we have calculated the loss function. To calculate the loss function we have used the cross entropy formula. Cross entropy is a method of comparing the scores predicted from each training or testing step and comparing it with the ground truth label from the dataset. It is considered to be the distance between the predicted scores and Labels.

$$L(s,i) = -\sum L_i \log(S_i)$$

Or Simply, Loss (score, label) =-Labels. Log (Scores). To train the parameters of the logits, an optimizer is used to reduce the loss function of the model in a particular training instance using a technique called gradient descent optimization. First, the gradient of the score is calculated using the loss function and objective function.

$$\theta=\theta-\eta\cdot \nabla J((\mathrm{L(s,l)}),\theta)$$

J() function is the objective function which is considered to be the goal of the gradient descent algorithm. The gradient descent algorithm will try to reduce the loss function by comparing it with the objective function. In order to reduce the loss function, it will change the weights and biases of the logits on each training step. But the gradient descent algorithm is only suitable to reduce the loss function only when the training dataset is small. This is because the training time for systems using gradient descent increases rapidly when the data set size increases. So in order to use large datasets we have followed a modified version of gradient Descent called stochastic gradient descent algorithm. The main difference is that the dataset is randomly shuffled and a mini batch of data is picked that is to be used to train the model.

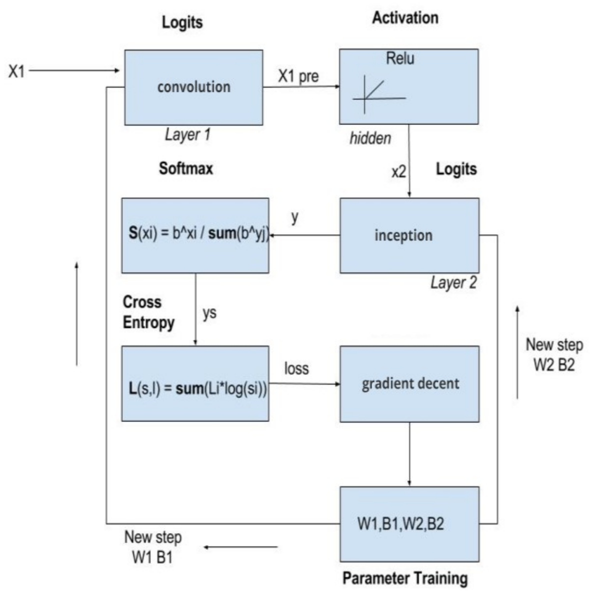


Figure 2A Data Flow chart for our Training Process

Our dataset has been divided into 3 parts, as Training, Validation and Testing. Training set is the sole set of data that is used to train our model. But as we are using stochastic gradient descent, the training set was divided into random minbatch so that to decrease the training time for large datasets.

B. Experimentations

We performed a through comparison on different hyper parameters that can be tuned, to measure the performance of our proposed architecture for Bangla handwritten character recognizer. We started with 128 hidden layers and 0.005 learning rate for the ReLu and got a decent MiniBatch, Validation and Test accuracy of 88.0%, 89.80% and 87.60% respectively. One of the two main hyper parameters that needs most attention are numbers of hidden layers in the subsampling layer or ReLu layer that is found immediately after the convolution layer. Therefore, we intuitively changed the number of hidden layers and trained for the optimum test accuracy. It seems that increasing the hidden layers up to 720 produces the increases the test accuracy results. But after 720 hidden layers, the test accuracy start to decline. But it seems that mini-batch accuracy and Validation accuracy still rises even if the test accuracy falls. This proves that after 720 hidden layers. The network start to generalized the training dataset and thus there is a stark difference between the test accuracy and mini-batch Accuracy. With 900 hidden layer, the network has fully generalized the mini-batch training set.

Hidden Layer	Steps	Mini-Batch Accuracy	Validation Accuracy	Test Accuracy	Minibatch Loss
128	30000	88.00%	89.80%	87.60%	0.587701
384	30000	90.00%	92.70%	88.60%	0.583768
512	30000	94.00%	92.60%	88.80%	0.503671
720	**30000**	88.00%	94.20%	**89.30%**	**0.397282**
750	30000	88.00%	**94.70%**	88.70%	0.348352
800	30000	94.00%	94.20%	88.20%	0.568419
900	30000	**98.00%**	93.80%	89.10%	0.345489

Table 1: Variable Hidden layer Experiments

Therefore with 900 hidden layer hyper parameter, we tested different variations of the learning rate hyper parameter, obtaining the results in Table 2. We started with a high learning rate of 0.055 and it provides a very low testing accuracy compared with the previous value of 0.005 learning rate. Therefore, we significantly decreased the learning rate and found a better testing accuracy with learning rate of 0.006 and best of the class validation accuracy. But increasing the learning rate more than 0.005 yields no more better results.

Learning Rate	Steps	Mini-batch Accuracy	Validation Accuracy	Test Accuracy	Minibatch Loss
0.055	30000	84.00%	93.60%	88.40%	0.480084
0.006	30000	92.00%	**94.70%**	88.80%	0.305358
0.005	**30000**	**98.00%**	93.80%	**89.10%**	**0.345489**
0.003	30000	92.00%	94.60%	88.05	0.367663

Table 2 Variable Learning Rate Experiments

The Fig 5 illustrates the result of increase in validation accuracy with number of steps. The validation accuracy increases sharply around 5000 Steps. And then it increases steadily afterwards. The final validation accuracy is around 89.10% with 900 hidden layers and 0.005 learning rate.

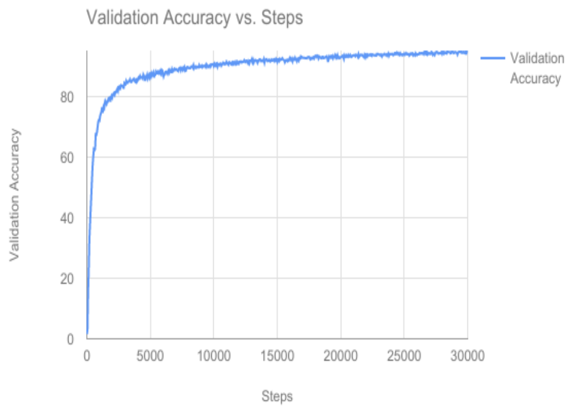


Fig 6

Therefore it is apparent that the whole network has converged with 900 hidden layers and 0.005 learning rate for BanglaLekha dataset with 84 class Bangla character.

Other works	Number of class	Classification	Compound character	Accuracy
BHCR-CNN [7]	50	CNN	No	85.96%
DBNs[8]	60	Unsupervised Deep belief Network	No	90.27%
Proposed method	84	CNN with inception	yes	89.10%

Table 3. Performance comparison with another CNN based approach and unsupervised approach

Table 3 shows comparison of test accuracy among proposed method, CNN based BHCH-CNN [7] and unsupervised deep belief network. The most significant part of BHCR-CNN and proposed method is that no feature selection techniques are used here. In [7] 50 classes (only Bangla basic characters) are only used in classification and the classification gives 85.96% recognition accuracy. In[8] a unsupervised deep belief network is used where 60 classes (10 Bangla numerals and 50 basic character) are used and have 90.27% recognition accuracy. No compound character are used for training the classification. In our proposed method 84 classes are used which consists of Bangla basic character, numerals and compound characters. Compound characters are complex in shape so it's a much challenging for a classifier to recognize compound character. Our proposed method successfully classify commonly used compound Bangla characters. Therefore our proposed classifier shows overall better performance recognizing more classes.

1

V. Acknowledgement

WE WOULD LIKE TO THANK BANGLALEKHA PROJECT FOR PROVIDING THE BANGLALEKHA-ISOLATED DATASET[6].

VI. References

[1] de Campos, Teófilo Emídio, Bodla Rakesh Babu, and Manik Varma. "Character Recognition in Natural Images." In *VISAPP (2)*, pp. 273-280. 2009.

[2] Huang, Weilin, Yu Qiao, and Xiaoou Tang. "Robust scene text detection with convolution neural network induced mser trees." In *European Conference on Computer Vision*, pp. 497-511. Springer International Publishing, 2014.

[3] Wang, Tao, David J. Wu, Adam Coates, and Andrew Y. Ng. "End-to-end text recognition with convolutional neural networks." In *Pattern Recognition (ICPR), 2012 21st International Conference on*, pp. 3304-3308. IEEE, 2012.

[4] Zhang, Zheng, Wei Shen, Cong Yao, and Xiang Bai. "Symmetry-based text line detection in natural scenes." In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2558-2567. 2015.

[5]Yin, Xu-Cheng, Xuwang Yin, Kaizhu Huang, and Hong-Wei Hao. "Robust text detection in natural scene images." *IEEE transactions on pattern analysis and machine intelligence* 36, no. 5 (2014): 970-983.

[6]Biswas, Mithun, Rafiqul Islam, Gautam Kumar Shom, Md Shopon, Nabeel Mohammed, Sifat Momen, and Md Anowarul Abedin. "BanglaLekha-Isolated: A Comprehensive Bangla Handwritten Character Dataset." *arXiv preprint arXiv:1703.10661* (2017).

[7]Szegedy, Christian, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. "Going deeper with convolutions." In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1-9. 2015.

1