Handwritten Bangla Character Recognition using Inception Convolutional Neural Network

Md. Adnan Taufique
Department of Computer Science and Engineering
Ahsanullah University of Science and Technology,
Dhaka, Bangladesh

Farhana Rahman
Department of Computer Science and Engineering
Ahsanullah University of Science and Technology,
Dhaka, Bangladesh

Md. Imrul Kayes Pranta
Department of Computer Science and Engineering
Ahsanullah University of Science and Technology,
Dhaka, Bangladesh

Nasib AL Zahid
Department of Computer Science and Engineering
Ahsanullah University of Science and Technology
Dhaka, Bangladesh

Syeda Shabnam Hasan
Assistant Professor
Department of Computer Science and Engineering
Ahsanullah University of Science and Technology
Dhaka, Bangladesh

ABSTRACT
With the advancement of modern technology the necessity of pattern recognition has increased a lot. Character recognition it's part of pattern recognition. In last few decades there has been some researches on optical character recognition(OCR) for so many languages like - Roman, Japanese, African, Chinese, English and some researches of Indian language like -Tamil, Devanagari, Telugu, Gujratietc and so many other languages. There are very few works on handwritten Bangla character recognition. As it is tough to understand like Bangla language because of different people handwritten varies in fervidity or formation, stripe and angle. For this it's so much challenging to work in this field. In some researches SVM, MLP, ANN, HMM, HLP & CNN has been used for handwritten Bangla character recognition. In this paper an attempt is made to recognize handwritten Bangla character using Convolutional Neural Network along with the method of inception module without feature extraction. The feature extraction occurs during the training phase rather than the dataset preprocessing phase. As CNN can't take input data that varying in shape ,so had to rescaled the dataset images at fixed different size. In total final dataset contains 100000 images of dimension 28x28. 85000 images is used for training and 3000 images is used for testing. After analyzing the results a conclusion is derived on the proposed work and stated the future goals and plans to achieve highest success and accuracy rate.

Keywords
Handwritten Bangla character, Shallow convonet, CNN, Inception, Data Normalization

1. INTRODUCTION
Optical Character Recognition(OCR) is the process of automatic recognition of handwritten character by computer. In the areas of pattern recognition, OCR is one of the most alluring and challenging technology with numerous practical applications. It can play a vital role to the advancement of an automation process and can improve the interface between man and machine in many applications. Character recognition provides an elegant solution for this sort of problem because it can process large amounts of data with minimal computational cost. This problem is a non-trivial one because of Handwritten characters by different people is varies in size, shape, angle, style. Due to this wide range of variability, it is difficult to recognize by a machine. For this handwritten character recognition is very much challenging. Most of the handwritten character recognition problems are complex and deal with the large number of classes. A lot of research have been done on several languages of handwritten character & digit and applied successfully on various real life application like -postal codes, bank checks etc. In many character & digit recognition research SVM, HMM, MLP,ANN etc has been used for the language of English, Roman, Chinese, Indian etc[1][2][3]. Using different Kernel based SVM Classifier and MLP Neural Network was used for English character where the accuracy was 94.8% and 80.96% respectively[4].

Bangla is the 1st language of Bangladesh, 2nd language of India and 5th language in the world. It is derived from the ancient Brahmin script through various transformation. Almost 160 million people in Bangladesh use Bangla and over 300 million people use Bangla to express emotion as speaking and writing purpose. The writing style is horizontal and left to right. The concept of upper or lower case is absent. There are 10 digits and 50 characters including vowel and constant. Bangla is also consists of many shaped characters. Two or more consonant characters combine to form Compound characters. 260 compound characters are present in the literature. But, according to 'Barnaparichaya' there are 194 Bangla compound characters. Bangla script has a very rich and complex alphabet set. Despite the complexity problem of handwritten Bangla character recognition and the popularity of Bangla Script evidences of research on OCR of handwritten Bangla characters, as observed in the literature, are few in number[5][6][7][8]. There exists limited work on the Bengali character set and most of these achieved recognition accuracy below 90%.

Convolutional Neural Network(CNN) is responsible for major breakthroughs in image classification and the core of computer vision system. There is no feature extraction in CNN unlike other approaches [14][15]. CNN has not yet been used much in handwritten Bangla character recognition. There exists a paper by Akhand et al. (2015) that has employed CNNs to the bangla character set, and that, too, achieved 85.96% accuracy.

Inception is used because the inception module helps to overcome a weakness of CNN which fixed the window sized [16]. It generates feature map using different window size. This allows to extract more features than usual which in turn improves the performance of the network gradually.

This thesis is oriented in 6 sections. Sections 1 presents introduction. Sections 2 officially defines the way of approaching the work and the architecture of that method which are used in this paper. In section 3, the information of data source and the way of preparing the dataset to train and test for the model is presented. Section 4 presents implementation and the result analysis. Section 5 contains conclusion.

2. PROPOSED METHOD

As input dimensions of a CNN cannot be varied at the time of training [17] , dataset images is rescaled to a fixed size. But It was not clear which input size would yield highest accuracy and be computationally feasible at the same time. The complexity of a CNN depends on the input size as complexity is proportional to the input size. As proposed approach is already a complex network have to find out the optimal input image size to reduce computational cost right from the beginning to do so. After preprocessing the data, a strategy have devised that might give an idea about the optimal image size. Strategy was to:

- Create multiple dataset of varying image size with same input images .
- Build a CNN that can be trained in short time for lots of dataset.
- Train the model for datasets varying some parameters of the dataset.
- Analyze the results and decide on the dataset size with respect to accuracy and time.
- Take the existing CNN and build upon that to train a model to achieve highest possible accuracy for the selected dataset.

The inception v3 module is not sufficient for the dataset . So, the below architecture is used for the research.

2.1 Shallow Convonet Architecture

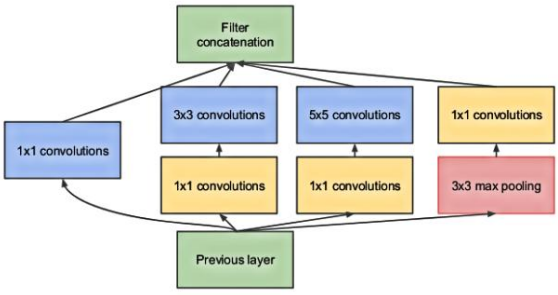

Fig-1: Inception Architecture.

To train the dataset shallow convonet have been designed - a barebones convnet with 2 convolutional layers, an inception module and a 1 fully-connected layer following them. The kernel of filter size of both of the Convolutional layer 5x5 depth of the convolutional layers or the number of convolutional layer feature map was 32 and depth of the fully connected layer was 64. These numbers were selected from general standard of CNN architecture recognizing MNIST dataset. Stride of 1 and odd number of zero padding was used thus the output produced was the same size as the inputs. For Activation or As cost function, softmax cross entropy function was used, Gradient descent optimizer[18] was used to minimize the loss with a learning rate of 0.2 and Softmax function was used to classify the images. The model was trained for 10 epochs for every dataset. As the weights and biases were initialized randomly for every training, the test accuracy varies each time within a small range. So, for every dataset the model is trained 5 times and took the mean values of the result. This procedure was performed for every experiment mentioned later in the paper. To establish a benchmark accuracy for the dataset. To analyze the result for different dataset parameter to select optimum set of parameters that will be used for training deeper convnet in future.

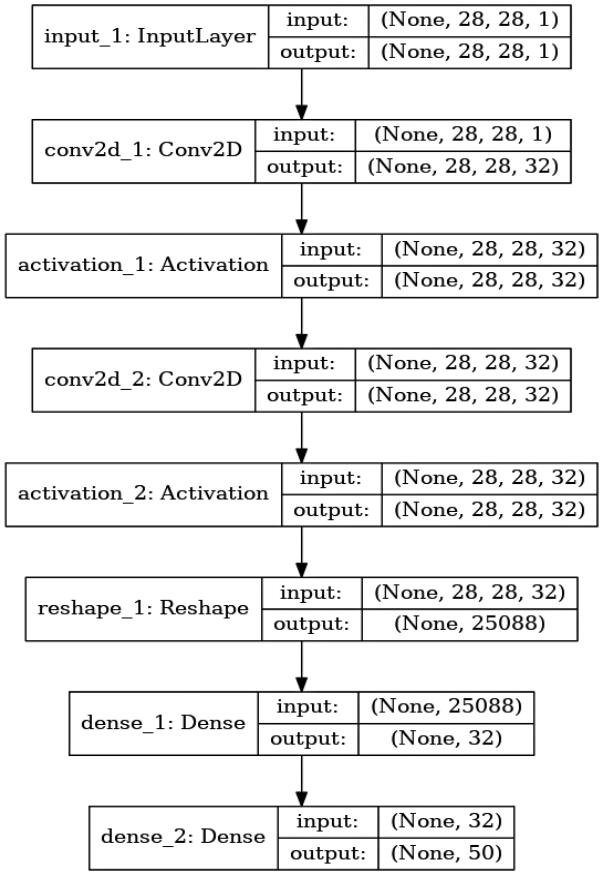

Fig-2: Shallow Convnet Architecture.

To achieve the goals, some experiments are run by changing some parameters of the datasets.

2.2 Deeper Architectures

After doing initial experiment to find out the nature of the dataset and find the nominal parameter for final dataset, some more components are added on existing shallow conv to achieve the optimal depth.

2.2.1 *Intermediate Convonet*

To achieve the desired goal, a Convolutional neural network is designed using 2 convolutional layers and 2 fully connected

layers. The kernel size of the 2 convolutional layers are 3x3 and 5x5. After every convolutional layer, max pooling is also used. Then 2 fully connected layer is used where depth of the

first layer is 64 and depth of the second layer is 32. A dropout layer is also used with a dropping probability of 50% is introduced. Also the learning rate of the model is exponentially reduced with a starting learning rate of 0.2.

2.2.2 *Deeper Convonet With Inception Module*

The module consists of 1x1 convolutions that reduce the dimension of the input followed by separate , parallel 3x3 and 5x5 convolutions. The input is also max-pooled with kernel size 3 and stride 1 followed by another 1x1 convolution. The 1x1 convolutions have significant impact, most of it they are utilized mainly as dimension reduction modules to dispel computational bottlenecks that would somehow limit the size of the networks. This takes into account expanding the depth, as well as the width of the networks without a noteworthy execution penalty. ReLU's are added after each of these paths. The output of all the filters are concatenated into one large output block and passed on as input to the next layer.

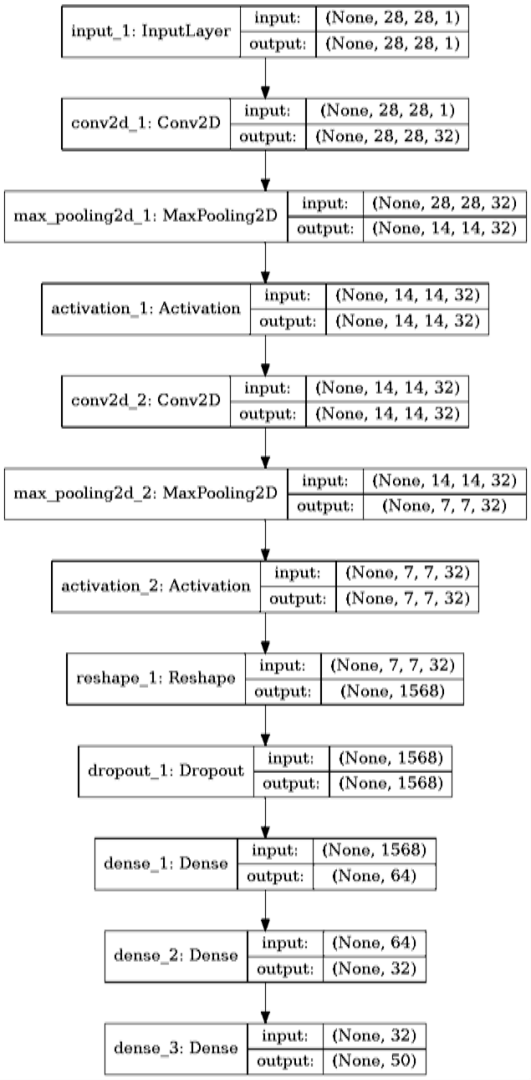

Fig-3: Intermediate Convnet Architecture.

This deeper model is trained with the final dataset. After introducing three(3) inception modules to the Intermediate Convnet architecture before the dense fully connected layers. 3 revision of the mentioned architecture by adding 1 Inception module each per revision thus it end up with 3 Models : Inception_1 Convnet, Fig-4, Inception_2 Convnet, Fig-5, Inception_3 Convnet, Fig-6. The reason for not make the model deeper because of increase in training time and decrease in accuracy.

3. HANDWRITTEN BANGLA CHARACTER DATABASSE

In this research benchmark dataset is used. This database along with other databases are freely available in this website [19]. The database contains 15000 images of 50 handwritten Bengali basic characters. For training it contains 12000 images of 50 characters, having 240 images per character and for testing it contains 3000 images of 50 characters, having 60 images per character. Being a benchmark dataset, this contains little to no salt and paper noise but has some wanted noise in form of blurring and missing pixel value

Fig-7: Examples of CMATERdb 3.1.2 dataset .

3.1 Dataset Preprocessing

Fig-8: Same image in different size& style.

In deep learning not only a huge number of data is needed but pre-processing data is needed for improving the accuracy of the model. Dataset contains wide variation of distinct characters because of different peoples' writing styles, Fig-8. All the images in the base dataset are in varying shapes. As convolutional neural network can not take input data that are varying in shape, all the images is scaled to a fixed dimension to make them eligible to feed into the network. Many of the images had gray background Fig-9(a). Inverting these images would not create a pure black and white image. So, all the images is taken by the folder, searching for pixel value of "127" i.e. gray color and replacing that with "255" i.e. white color. Lastly, all the images are converted from RGB to GRAYSCALE and inverted the colors Fig-9(b) to reduce computational cost & it would reduce use of memory while

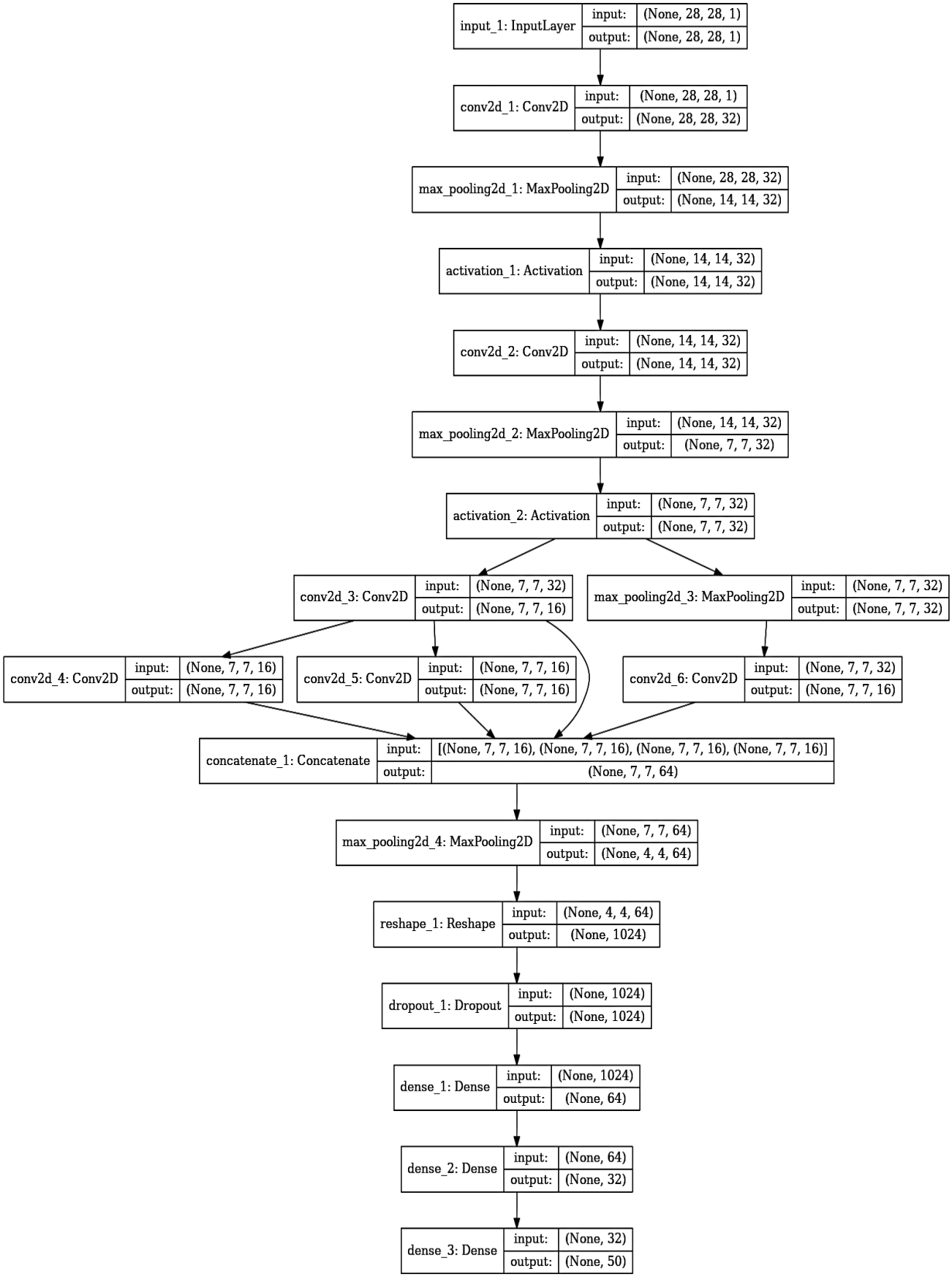

Fig-4: Architecture of Inception1 convnet.

Fig 5: Architecture of Inception2 convnet.

Fig 6: Architecture of Inception3 convnet.

training, inversion of the background into black and the actual character into white.

(a) **(b)**

Fig-9: (a)raw images (b)corresponding resized and gray scale images.

For overcoming the problem of data scarcity, the dataset is augmented to increase the number of input images which is ideal for the training a neural network. First of all different size of data is used where each pixel in the image has different value After selecting the size the image has been rotated in 20-25 degree. It changes the pixel value of that image. So, it becomes the new image for machine.

3.2 Data Normalization

Being a benchmark dataset, this contains little to no salt and paper noise but has some wanted noise in form of blurring and missing pixel value. To normalize the dataset for some noisy data as hand picking all of them would be a huge task. So, normalize the data to have approximately zero mean Fig-10 and standard deviation Fig-11 of 0.5 to make training the model easier.

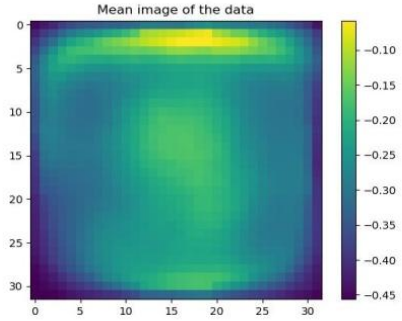

Fig-10: Mean of the dataset (28x28image)

Initially primary training dataset had 50,000 images while validation dataset 5,000 and 3,000 images respectively.

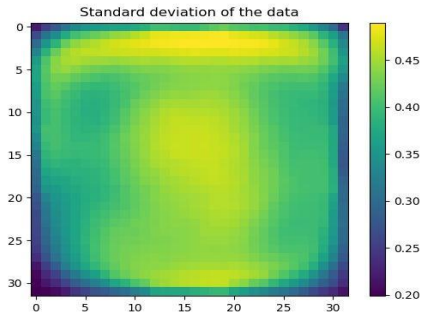

Fig-11: Standard deviation of the data (28x28image).

3.3 Data Randomization

In the mini-batch training of a neural network, it is important to shuffle the training data because it helps the training converge fast and prevents any bias during the training and prevents the model from learning the order of the training. A random sample of training images are taken at the time of training Fig-12.

So after performing multiple experiments by varying the size of each dataset in total the final dataset contains 100000 images of dimension 28x28. 85000 images for training, 12000 for validation and 3000 for testing is used. The reason behind using less testing images is that given 3000 images in the benchmark dataset without augmenting it to preserve the integrity of the performance of the proposed architecture.

Fig-12: Randomly taken training Images during training.

4. IMPLEMENTATION AND ANALYSIS

Image size for training a convolutional neural network in vital because while in some cases, higher resolution images helps to achieve higher accuracy while in others , lower resolution is desired. So it's important to determine the optimal image size for training. Different image size starting from 24x24 pixel size image to 52x52 pixel image size with an increment of 4 pixel . In total 8 datasets has been created containing 50000 training image ,5000 validation image and 3000 training images. It has been trained on shallow convonet. The result is shown in Table-1.

How much the result gets affected by changing number of inputs but creating 6 datasets for each input dimensions, training them, analyzing and representing the data would be complicated. Instead, 32x32 of input dimension selected a constant and created 5 more datasets alongside of the primary dataset of the above-mentioned dimension training data increasing 10000 training images per dataset up to 1000000 training images for 6th dataset. The training time increased with the number of training image and also accuracy increasing proportionally, Table-2

Table-2: Results of varying Train data in datasets on shallow convnet.

SL No	Image Size	No. of Training Images	Training Time (sec)	Test Accuracy (in %)
1	32	50,000	101.98	87.17
2		60,000	125.92	87.14
3		70,000	142.77	88.26
4		80,000	164.44	88.63
5		90,000	187.88	89.02
6		1,00,000	219.15	88.95

Table-1: Results of varying image size datasets on shallow convnet.

SL No	Image size (pixel)	Training Time (sec)	Testing time (sec)	Total time (sec)	Validation accuracy (in %)	Test accuracy (in %)
1	24	63.93	0.26	64.19	93.66	86.86
2	28	82.08	0.35	82.44	95.58	87.37
3	32	101.98	0.45	102.44	94.82	87.17
4	36	127.59	0.57	128.16	92.52	86.52
5	40	156.48	0.78	157.25	93.82	86.44
6	44	184.10	0.66	184.77	93.42	86.03
7	48	220.99	0.78	221.77	90.94	84.68
8	52	424.55	0.98	425.53	91.18	84.43

Then 32x32 size image is selected again and created 5 more datasets of that image size with training data increasing testing images per dataset up to 15000 training images for 6th dataset. After training the shallow convnet with these dataset by seeing in the below Table-3 testing time increase with the number of testing image and but the accuracy is decreasing. This is nothing out of the ordinary due to the nature of convolutional layer. On its own, a convolutional layer is not rotation invariant that means the model is not good at classifying rotating testing image accurately. Naturally all convnets have 1 or more pooling layers to achieve rotation invariance. In this convonet, does not have any pooling layer and the test images is augmented by use of rotation. So, this result is expected as well.

Table-3: Results of varying Test data in datasets on shallow convnet.

SL No	Image Size	No. of Testing Images	Testing Time (sec)	Test Accuracy (in %)
1	32	3,000	0.46	87.17
2		5,000	0.45	86.58
3		6,000	0.71	86.45
4		9,000	1.08	85.10
5		10,000	1.20	85.07
6		12,000	1.43	85.15

4.1 Training the Model For Increasing Epochs

Then for achieving the highest accuracy have to find out the optimal number of epoch. This does vary for model due to their depth and capacity. For this experiment the primary 32x32 dimension image with 50000 training ,5000 validation and 3000 testing data has been taken. The calculation of epoch is using this formula[20]:

1 Epoch = No of training image/Batch size

The model is trained from 10 epochs to 50 epochs with a 5

5 epochs increment. The results are shown in Table-4

Table-4: Results of varying training epochs on shallow convnet.

SL No	Image Size	No. of Epochs Trained	Training Time (sec)	Test Accuracy (in %)
1	32	10	101.978	87.17
2		15	148.18	87.76
3		20	218.17	88.33
4		25	247.18	88.8
5		30	316.77	89.07
6		35	346.92	88.92
7		40	416.72	88.86
8		45	447.07	88.76
9		50	520.78	88.72

4.2 Training With Deeper Models

After deciding on the dataset parameters and creation of the dataset, then deeper models has trained with said dataset. These models have trained for 80 epochs and this hyperparameter is used for all later trainings. The learning rate is reduced per 2000 global step each. Dropout layer is also used for each of the model. Dropout effectively allows training and sampling from a probability distribution of network architectures. According to Baldi et al [21], Dropping a neuron with 0.5 probability gets the highest variance for this distribution or results in the maximum amount of regularization.

4.2.1 *Training with intermediate model*

At first the intermediate model is trained with final dataset. This model seemed like the next step to build upon the existing model. This model yielded a Validation accuracy of 94.9% and a test accuracy of 93.23% and took 43.25min to train. Loss and accuracy of this model is shown below in Fig-13.

4.2.2 *Training with Inception Convonet*

The inclusion of pooling layer and a fully connected layer in the end does improve the performance of the model. After training with said model, an inception module is added to it and trained it with the dataset. That gave a validation accuracy of 97.89% and a test accuracy of 96.23% and took 116.585 min to train. Loss and accuracy for 80 epochs are given below in Fig-14. As the inclusion of inception module increased the accuracy significantly, another inception module is added on to the existing inception_1 model to create inception_2 model for better performance . This model produced a validation accuracy of 97.11% and a testing accuracy of 96.7% and took 145.19 min to train. As The performance improved but not as drastically as the previous model. Loss and accuracy for 80 epochs for this model are given below in Fig-15. After 2nd inception module improved the performance, another inception module is also added to existing inception_2 model and create this model . This model yielded a validation accuracy of 96.82% and a test accuracy of 95.3%and took 151.45 min to train. But this time this model does not perform better than the previous one. Loss and accuracy for 80 epochs for this model are given below in Fig-16.

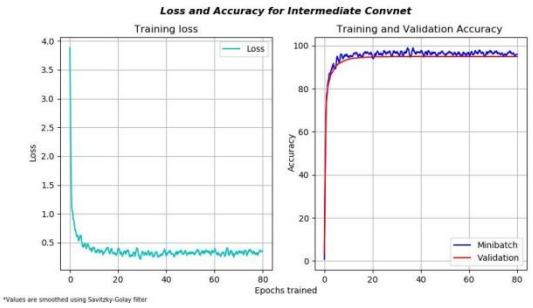

Fig-13: Loss and Accuracy for Intermediate Convnet.

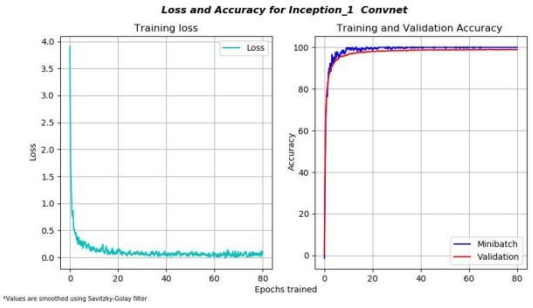

Fig-14: Loss and Accuracy for Inception_1 Convnet.

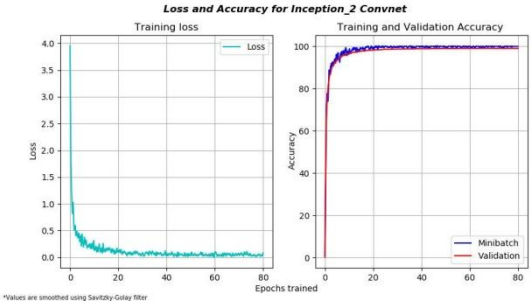

Fig-15: Loss and Accuracy for Inception_2 Convnet.

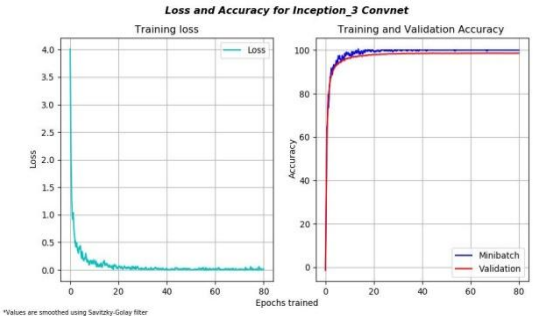

Fig-16: Loss and Accuracy for Inception_3 Convnet.

5. AGGLOMERATION OF THE RESULTS AND COMPARISION

After training the dataset with 5 different models , all the data is collected and analyzed the capability of completion of task of the said models

For the sake of uniformity ,the final dataset is trained by shallow convnet and achieved a validation accuracy of 95.23% and test accuracy of 89.02% . By looking at the results in the given Table-5, a significant increase has been seen in performance gain with each model up until Inception_2.

By plotting the data in a bar graph , the results are quite similar for all the inception models, Fig-17 :

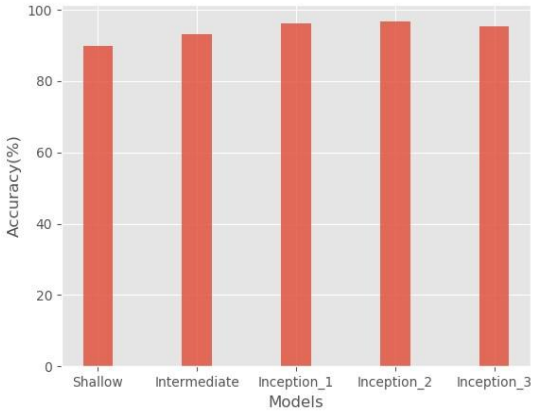

Fig-17: Accuracy of the different models.

Table-5: Comparison of the performance.

SL no	Model Name	Validation Accuracy (%)	Test Accuracy (%)	Precision	Sensitivity	F1-Measure	Specificity
1	Shallow Convnet	95.23	89.02	0.8962	0.8923	0.8942	0.99780
2	Intermediate Convnet	94.9	93.23	0.9341	0.9333	0.9337	0.99863
3	Inception_1 Convnet	97.89	96.23	0.9601	0.9593	0.9597	0.99917
4	Inception_2 Convnet	97.11	96.7	0.9601	0.9596	0.9599	0.99917
5	Inception_3 Convnet	96.82	95.3	0.9509	0.951	0.9512	0.999

To look at the time required for training each model , a pattern can detect here:

Table-6: Training time for all the models.

SL no	Model Name	Epochs Trained	Training time (Minutes)
1	Shallow Convnet	80	30.544
2	Intermediate Convnet		43.254
3	Inception_1 Convnet		116.585
4	Inception_2 Convnet		145.17
5	Inception_3 Convnet		151.45

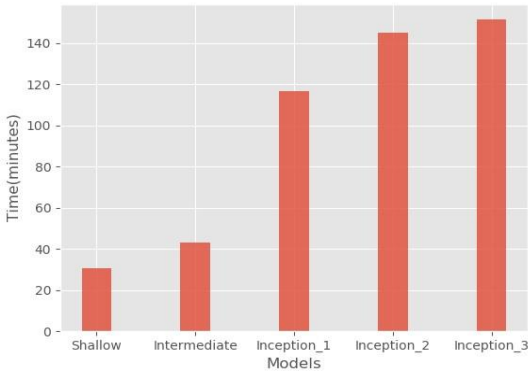

Fig-18: Bar plot of Training time for all the models.

5.1 Analysis of the Result

As evident in the results, the accuracy for dataset increases with depth but the increases diminish up to certain depth and then decreases. This occurs due to the reduction in learnable parameter in the network from shallow Convnet with each model and then increase in parameter and complexity with 3 inception modules in Inception_3 Convnet . The increase in time can be justified by the increasing complexity of the models thus requiring more time to be trained with each model.

5.2 Comparison With Existing Work

By comparing the performance of this models with the existing works Table-7, this models perform better than most of them notably than proposed model by Rahman et al [6] which got 85.96% accuracy and slightly better than proposed method by Bhattacharya et al[10] which got 95.84% accuracy. Whereas proposed model have achieved 96.7% accuracy.

5.3 Visualization of Result

This is the confusion matrix for Inception_2 Convnet Trained with dataset at Fig-19. This shows how this model is doing. Some test images has taken randomly at the end of training to see how the model is doing. By comparing the prediction of those images at Fig-20 and Fig-21 an incorrect prediction can see here. Because there are similarities to some degree to the incorrect predicted characters and true characters of that class. So only the incorrect predictions are plotted Fig-21 to see if there is any ongoing pattern.

In the figure the model is predicting the characters having similar structure incorrectly with 7th example being the exception of said pattern.

Table-7: Comparison with Existing methods of Handwritten Bangla Character.

Reference	Total class	Classification	Accuracy(%)
Das et al [1]	256	SVM ,MLP, KNN	80.58
Bhowmick et al. [9]	45	MLP	84.33
Rahman et al[3]	50	CNN	85.96
Bhattacharya et al	50	MQDF, MLP	95.84
Proposed Inception_2 Convonet	50	CNN with inception module	96.7

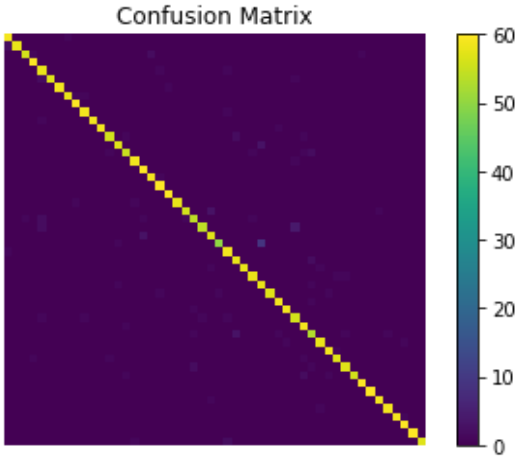

Fig-19: Confusion matrix of Inception_2 Convnet.

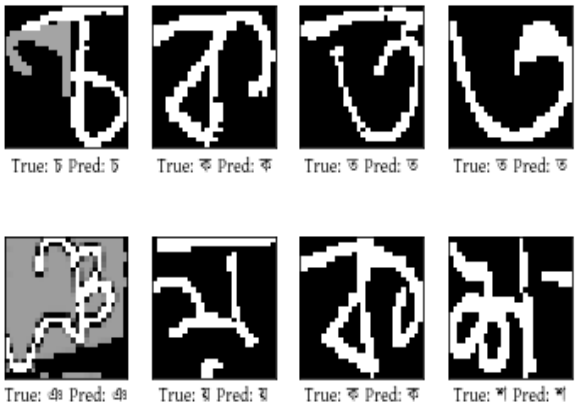

Fig-20: Randomly taken test images and prediction of that image.

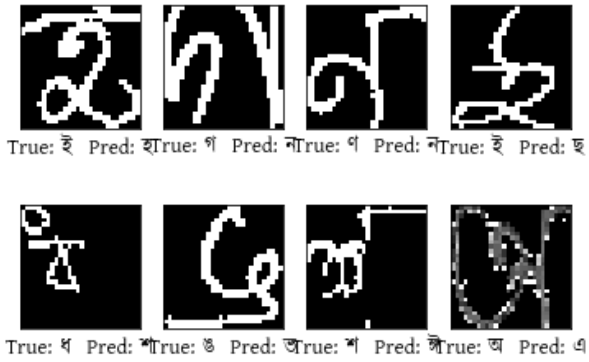

Fig-21: Test images of incorrect predictions.

6. CONCLUSION

Convolutional Neural Network (CNN) is found efficient for Handwritten Bangla Character Recognition now a days. It does not employ any feature extraction method like other related works. To achieve highest possible accuracy, a benchmark dataset is selected, processed and augmented the training images to create a general format for the dataset. A large number of dataset is tested by CNN with inception module. Using two inception conv the better accuracy have been achieved than using three inception conv. The proposed method is shown competitive performance with the exiting methods on the basis of test set accuracy for the dataset. The accuracy of this work is 96.7%. Based on the result it indicates that there might be still some improvement in CNN & inception training to get better performance.

All the process of this work still does not produce errorless prediction. So, in order to achieve that the way is to improve the architecture by adding right amount layers that can distinguish between characters that have the same shape at the same time make the model avert overfitting and use different arrangement of inception module like arranging them in parallel to see if there is an improvement in capturing highly detailed features. state of the art CNN architectures could be trained like Inception-ResNet, DenseNet and Capsule net which are deemed to perform near flawlessly in case of image recognition with these dataset. Aside from Bengali alphabets, working on the recognition of Bengali handwritten digits and compound character as well will be the contribution of next research. It can be said that the work in this research is a step in the right direction and can be useful for Bengali character and word recognition and OCR application.

7. REFERENCES

[1] Williams, Kyle, Hussein Suleman, and Jorgina K. do R Paihama. "A comparison of machine learning techniques for handwritten." *Proceedings of the South African Institute for Computer Scientists and Information Technologists Conference*.ACM, 2013.

[2] Vijayaraghavan, Prashanth, and Misha Sra. "Handwritten Tamil Recognition using a Convolutional Neural Network",2015.

[3] Shanthi, N. and Duraiswamy, K., 2010. A novel SVM-based handwritten Tamil character recognition system. *Pattern Analysis and Applications*, *13*(2), pp.173180.

[4] Kumar, Parveen, Nitin Sharma, and ArunRana. "Handwritten Character Recognition using Different Kernel based SVM Classifier and MLP Neural Network (A COMPARISON)." *International Journal of Computer Applications*53.11 (2012).

[5] Das, Nibaran, et al. "Handwritten Bangla character recognition using a soft computing paradigm embedded in two pass approach." *Pattern Recognition*48.6 (2015): 2054-2071.

[6] Rahman, Md Mahbubar, et al. "Bangla handwritten character recognition using convolutional neural network." *International Journal of Image, Graphics and Signal Processing (IJIGSP)* 7.8 (2015): 42.

[7] Bhowmik, Tapan Kumar, et al. "SVM-based hierarchical architectures for handwritten Bangla character recognition." *International Journal on Document Analysis and Recognition (IJDAR)* 12.2 (2009): 97-108.

[8] Rahman, Ahmad Fuad Rezaur, R. Rahman, and Michael C. Fairhurst. "Recognition of handwritten Bengali characters: a novel multistage approach." *Pattern Recognition* 35, no. 5 (2002): 997-1006.

[9] Khan, Haider Adnan, Abdullah Al Helal, and Khawza I. Ahmed. "Handwritten bangla digit recognition using sparse representation classifier." *Informatics, Electronics & Vision (ICIEV), 2014 International Conference on*. IEEE, 2014.

[10] Bhowmik, T.K., Bhattacharya, U. and Parui, S.K., 2004, November. Recognition of Bangla handwritten characters using an MLP classifier based on stroke features. In *International Conference on Neural Information Processing*(pp. 814-819).Springer, Berlin, Heidelberg.

[11] Alom, MdZahangir, et al. "Handwritten Bangla Digit Recognition Using Deep Learning." *arXiv preprint arXiv:1705.02680* (2017).

[12] Szegedy, Christian, et al. "Going deeper with convolutions." Proceedings of the IEEE conference on computer vision and pattern recognition. 2015.

[13] Scherer, Dominik, Andreas Müller, and Sven Behnke. "Evaluation of pooling operations in convolutional architectures for object recognition." *Artificial Neural Networks–ICANN 2010* (2010): 92-101.

[14] Discussion about Convolutional Neural Network http://ufldl.stanford.edu/tutorial/supervised/ConvolutionalNeuralNetwork (LAT*-16 july,2017,11:06pm) .

[15] Stanford cs231n lectures on CNN http://cs231n.github.io /convolutional-networks (LAT*-15 april,2018,12:35am).

[16] A Beginner's Guide To Understanding Convolutional Neural Network https://adeshpande3.github.io/ adeshpande3.github.io/A-Beginner%27s-Guide-ToUnderstanding-Convolutional-Neural-Networks (LAT*-20 june,2017,9:21pm).

[17] Discussion about input dimension of CNNs https://www.quora.com/Does-input-dimension-resolution-affect-the-performance-of-Convolutional-Neural-Networks. (LAT*-30 april,2018,12:00am).

[18] Coursera DNN specialization course lectures https://www.coursera.org/learn/neural-networks-deeplearning/ lecture/A0tBd/ gradient-descent(LAT*-28 January,2018,10:02am).

[19] Selected dataset repository on google code http://code.google.com/p/cmaterdb(LAT*-18 june,2017, 6:42 pm).

[20] Discussion the characterization of epoch and iteration https://stackoverflow.com/questions/4752626/epoch-vs-iteration-when-training neural-networks(LAT*-28 November,2017,10:s02am).

[21] Baldi, P. and Sadowski, P.J., 2013. Understanding dropout. In *Advances in neural information processing systems* (pp. 2814-2822).

[22] Kanan, Christopher, and Garrison W. Cottrell. "Color-to-grayscale: does the method matter in image recognition?." *PloS one* 7.1 (2012): e29740.

[23] Article explaining CNNs on a deeper level https://ujjwalkarn.me/2016/08/11/intuitive-explanation-convnets(LAT*-23 july, 2017,8:50pm) .

[24] Article explaining inception modules https://hacktilldawn.com/2016/09/25/inception-modules-explained-and-implemented(LAT*-18april,2018,10:15 pm).

[25] Discussion on the process of calculating the number of parameters of a convolutional neural network https://stackoverflow.com/questions/42786717/how-to-calculate-the-number-of-parameters-for-convolutiona-neural-neural-network/42787467(LAT*-25april,2018, 6:45pm).

[26] Coursera DNN lecture on activation functions https://www.coursera.org/learn/neural-networks-deep-learning/lecture/ OASKH/ why-do-you-need-non-linear-activationfunctions(LAT*-28january,2018,10:03am).