Bangla Handwritten Digit Recognition and Generation

Md. Fahim Sikder

Abstract Handwritten digit or numeral recognition is one of the classical issues in the area of pattern recognition and has seen tremendous advancement because of the recent wide availability of computing resources. Plentiful works have already done on English, Arabic, Chinese, Japanese handwritten script. Some work on Bangla also have been done but there is space for development. From that angle, in this paper, an architecture has been implemented which achieved the validation accuracy of 99.44% on BHAND dataset and outperforms Alexnet and Inception V3 architecture. Beside digit recognition, digit generation is another field which has recently caught the attention of the researchers though not many works have been done in this field especially on Bangla. In this paper, a Semi-Supervised Generative Adversarial Network or SGAN has been applied to generate Bangla handwritten numerals and it successfully generated Bangla digits.

1 Introduction

Recognizing handwritten numerals is one of the emerging problems in the sector of computer vision. Automation of the banking system, postal services, form processing are the practical example of handwritten character recognition [24, 25, 33, 10, 21, 29, 5]. A lot of work already has been done with great accuracy in the recognition of English handwritten digits [3, 19]. Researchers used support vector machine, histogram of gradient oriented, neural network etc algorithm to solve these problems. Recently, a lot of focus has been drawn to the neural network architecture due to the wide availability of high-performance computing systems [1]. ANNs are computing system which is influenced by the organic neural network. Convolutional Neural Network

Md. Fahim Sikder
Jahangirnagar University, Savar, Bangladesh, e-mail: fahimsikder01@gmail.com

is one of the architectures of neural network which makes it easy to recognize image with great accuracy. Besides English, a lot of work also done in Arabic, Chinese, Japanese and Roman scripts [8, 14, 12, 20, 31, 30, 15, 9, 7]. But in the case of Bangla, not many works have been done and there is a chance for improvement.

On the other hand, generating images is another outstanding image processing field recently caught the attention of researchers. Image generation can be used in art creation, fraudulent detection also can be applied in law enforcement. Generative Adversarial Network or GAN, another architecture of neural network is been used to generate the image. Researchers also applied GAN to generate MNIST dataset but not much work has been done in other datasets. To mend this research gap on Bangla, we have implemented an architecture which recognizes Bangla handwritten digits at 99.44% accuracy using BHAND dataset which contains 70000 images of Bangla handwritten digits which are collected from 1750 persons. At the same time, we have implemented a semi-supervised generative adversarial network or SGAN to generate Bangla digits. The paper is arranged as follows: Section 2 reviews the relevant works, Section 3 describes the proposed solution, Section 4 describes the result and lastly, Section 5 concludes the paper.

2 Related Works

A lot of research works have been done on Bangla handwritten digit recognition using SVM [6], HOG [4] etc. Recently loads of attention is being given on deep learning because of easy access to GPU (graphics processing unit). Using multilayer convolutional layer, pooling layer increases the performance of accuracy. Some of the legendary deep learning based architecture such as Alexnet [16], LeNet [18], Inception V3 [32] took the accuracy of image recognition to the next level. MNIST recognition [17], CIFAR-10 database recognition [16] are some example of that architecture. For Bangla handwritten recognition numerous work has been done. But initially, it was troublesome for the researcher because of the limitation of a dataset [2]. But now some great datasets are available for Bangla digit recognition. A deep belief network is being introduced where the author first used unsupervised feature learning then it's followed by a supervised fine-tuning [27]. In [11], the author removed overfitting problem and has an error rate of 1.22%.

Besides digit recognition, few works have been done on digit generation. Researchers used different kinds of generative adversarial networks (GAN) to generate digits or characters. Auxiliary Classifier GAN [23], Bidirectional GAN [13], Deep Convolutional GAN [26], Semi-Supervised GAN [22] were used on MNIST dataset to generate digits.

3 Proposed Work

In this work, we have proposed a architecture for digit recognition which outperforms Alexnet [16] and Inception V3 [32] model at validation accuracy and error on the BHAND [11] dataset. Also, we have implemented Semi-Supervised Generative Adversarial Network (SGAN) for digit generation for the same dataset.

3.1 Dataset Description

For recognition and generation, BHAND dataset has been used which contains 70000 handwritten Bangla digits. This is one of the biggest datasets of handwritten Bangla digits. This dataset is divided into three sets: Training set (50000), Testing set (10000) and Validation set (10000). These 70000 data is collected from 1750 persons. The images are gray-scale and the dimension is $32 * 32$.

3.2 Network Architecture

For recognizing handwritten digit, we have proposed an architecture which consists several convolutional layers, pooling layers, normalization layers, and dense or fully connected layers. In the first convolutional layer, we took the $32 * 32$ images as input from the dataset. As mentioned earlier the images are grayscale, so it has 1 channel. In this layer, we have taken 32 filter which has the filter size of $2 * 2$. The output of this layer then goes into a second convolutional layer which also has 32 filters and the size of those filters is $2 * 2$. Then the outcome of the second convolutional layer feed into max pooling layer which has the filter size of $2 * 2$ and the stride size is 2. This outcome then goes into a normalization layer. These convolutional layers, pooling layer and normalization layer, together we named it *block*. In a single block the number of these layers could vary. The second block is composed of three convolutional layers, one max pooling layer, and another normalization layer. The amount of filters in the second block's convolutional layers are 64 and the filter size is $3 * 3$. This max pooling layer has also $2 * 2$ filter size and stride of 2. Then the third to sixth block consists of two convolutional layers, one pooling layer and one normalization layer. Third block's convolutional layer has 128 filters and the size of the filters is $5 * 5$, fourth block's convolutional layer has 256 filters which has $5 * 5$ filter size, fifth block's convolutional layer has 384 filters, sixth block's convolutional layer has 512 filters and their filter size is $5 * 5$. And all the blocks have the same pooling layer architecture. It

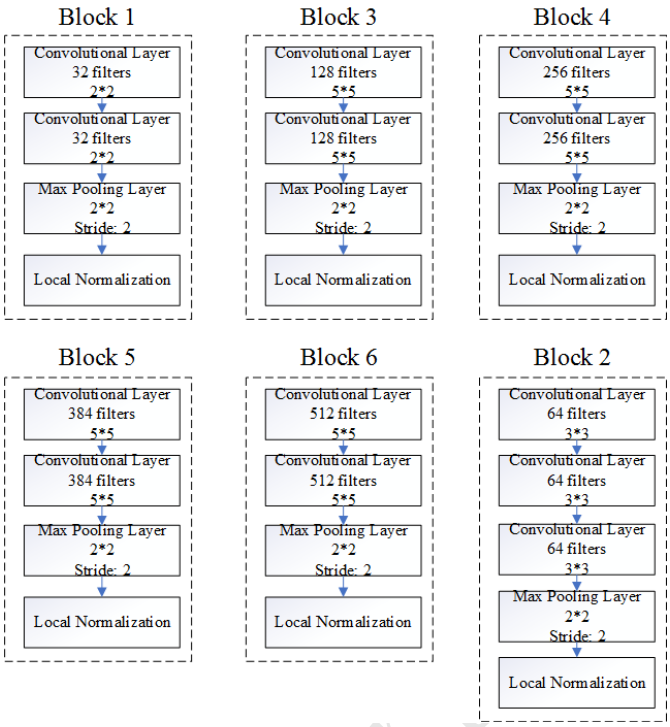

Fig. 1 Blocks of the architecture

has $2*2$ filter size and stride size 2. Figure 1 shows the blocks used in this architecture.

The outcome of the sixth block then feed into a fully connected layer which has 1024 units then we drop the 50% of the neuron for avoiding overfitting then the output is fed on the second fully connected layer which has 5120 units. Here we also drop the 50% of the neuron. Till now every layer used *relu* activation function. The following equation [28] is how *relu* works.

$$R(z) = max(0, z)$$

Now the output is then fed into the last fully connected layer which has 10 units because we have 10 class as output and here we have used $softmax$ activation function. The following equation [28] is how $softmax$ works.

$$s(z)_j = \frac{e^{z_j}}{\sum_{k=1}^{K} e^{z_k}}$$

The complete architecture of the recognizing part is shown in figure 2.

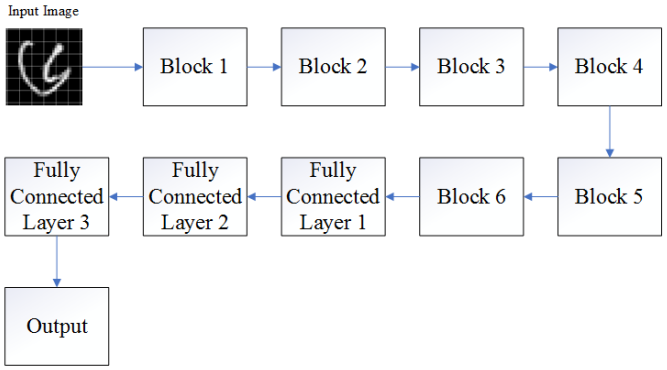

Fig. 2 Our architecture for digit recognition

Now for the digit generation part, here Semi-Supervised Generative Adversarial Network (SGAN) [22] is used for this task. Here we have a generator and discriminator. We took random noise as input, then the noise goes to the generator, at the same time we took a sample from training dataset. The generator attempts to forge the sample from training dataset and both the real and fake data goes to the discriminator then the discriminator attempts to distinguish between the genuine and the fabricated one. Usually, in GAN we train generator and discriminator concurrently and after training, we could discard discriminator because its only used for training the generator. In SGAN we alter the discriminator into a classifier and we discard the generator after the training. Here generator is used to aid the discriminator during training. Figure 3 shows the complete architecture of the SGAN.

In the generator, first, we took a random vector as input then we reshape it and then batch normalize it. Then we *upsample* the output. After that, we took a convolutional layer and pass the output through it. The convolutional layer has 128 filters and the filter size is $3*3$ also we used the *same* padding. We again use batch normalize and upsample in it. After that, we use another convolutional layer which has the same filter size and padding but it has only 64 filters and we again batch normalize it. The last two convolutional layers used *relu* activation function. Now the output is passed through the last convolutional layer which has one filter and the filter size and padding are like the same as others and it used *tanh* activation function. Now for the discriminator part, it is a multiclass classifier. We have used four convolutional layers. First convolutional layer takes $32*32$ images and it has 32 filters which have the size of $3*3$ also the strides of 2 to reduce the dimension of the feature vectors. Here we have used $LeakyRectifiedLinearUnit$ activation functions. Then we drop 25% of neurons for avoiding overfitting. Then the output goes to the next convolutional layers which have 64 filters and the size and strides are same as the last one. Then again, we drop 25% of neuron and use batch normalization. In the third and fourth convolutional layer, the

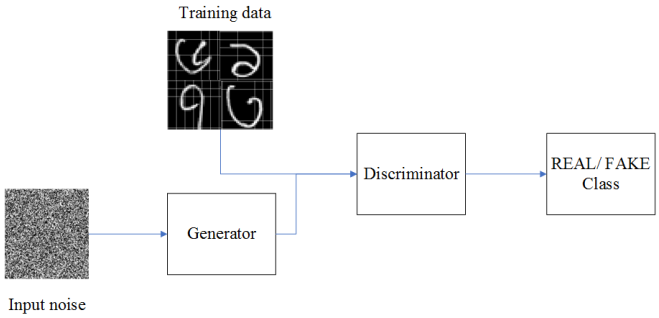

Fig. 3 Architecture of SGAN

filter size is the same but has 128 and 256 filters respectively. Then we flatten the output. In the end, we used two dense or fully connected layers. The last layer takes $N+1$ units because discriminator could generate $N+1$ outputs because of the fake label. Here is N is the number of total class and we used $softmax$ activation function. We used $binary-crossentropy$ loss function and *Adam* optimizer.

4 Experimental Analysis & Result

We have implemented our architecture using BHAND dataset which has 50000 training image, 10000 testing image and 10000 validating images of handwritten Bangla numerals. It has $32*32$ image dimension and the number of the channel was 1. For recognizing the digit, we have also applied this dataset in popular alexnet and inception v3 model. We have run a total of 19550 steps in the training and achieved 99.44% validation accuracy. We have used $rmsprop$ optimizer and $categorical-crossentropy$ as loss function. The learning rate in our architecture was 0.001. A detailed analysis of our experiments is shown in table 1.

Table 1 Comparison of our model with others for recognizing digit

Model Name	Steps	Validation Accuracy	Validation Error
Alexnet	19550	97.74%	0.1032
Inception V3	19550	98.13%	0.07203
Our Model	19550	99.44%	0.04524

The validation accuracy and the validation error of our model is shown respectively in figure 4 and 5.

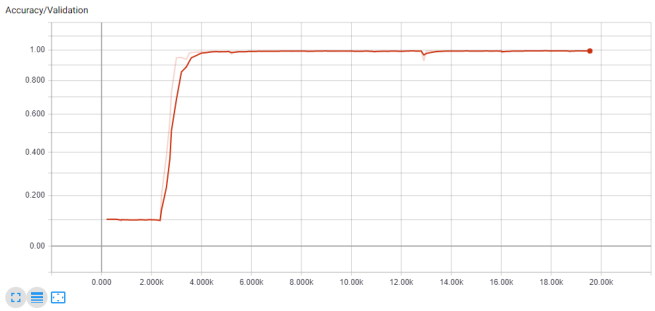

Fig. 4 Validation Accuracy of our model for recognizing digit

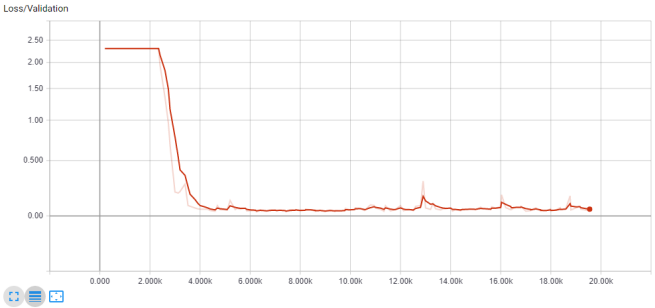

Fig. 5 Validation Error of our model for recognizing digit

For generating the Bangla handwritten image we also used the same dataset. For generating an image, we have used the Semi-Supervised Generative Adversarial Network (SGAN). Here we have built our model using generator and discriminator. Generator took a random vector as input. On the other hand from the real train dataset, an image goes to the discriminator. Generator tries to fool the discriminator by mimicking the real image. Then the discriminator discriminates the real and forges image. For our generator, we have used a combination of a fully connected layer, convolutional layer. Also, we need to normalize and upsample our data. For the discriminator, it also has a series of a convolutional layer and fully connected layer. Discriminator took the image as input to the input dimension is $32 * 32$. It used two loss function: $binary - crossentropy$ and $categorical - crossentropy$ whereas generator used $binary - crossentropy$. Here we have used Adam optimizer where the learning rate is 0.002. We have also reshaped our data to -1 to 1 because of the usage of *sigmoid* and *tanh* activation function. After 300000 steps of training, we have got 0.368 loss of discriminator and 0.694 generator loss. From figure 6 we can see the output of our SGAN. The first image (a) is from 0 step, the second image (b) is after 100000 and (c) and (d) image are after respectively 200000 and 300000 steps. The training loss is shown in figure 7.

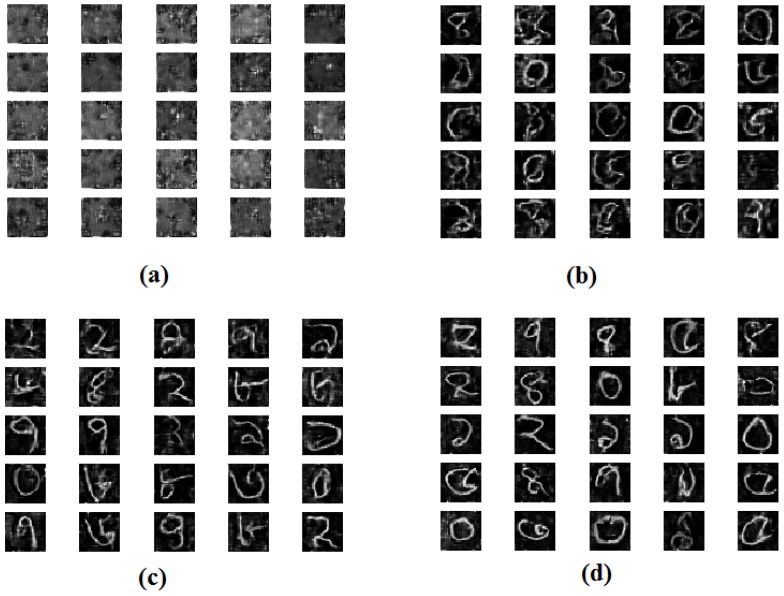

Fig. 6 Output of our generation model at step 0, 100000, 200000 and 300000

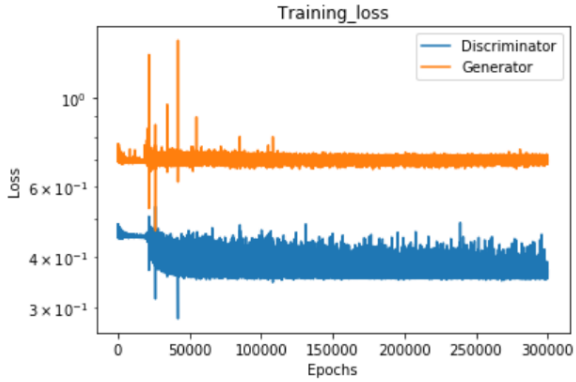

Fig. 7 Training Loss of our model for digit generation

5 Conclusion

Loads of work have been done in the area of handwritten numeral recognition but still, there is an opportunity to improve and only a few works has been done in the area of digit generation. From that motivation, in this paper, we have proposed a architecture for recognizing Bangla handwritten digits which outperforms popular alexnet and inception v3 architecture using BHAND

dataset. By adding a more convolutional layer and hyperparameter tuning could result in a better performance. Also, we have implemented the Semi-Supervised Generative Adversarial Network (SGAN) using the same dataset and successfully generate Bangla digits. In the future, we will try to reduce the discriminator's training loss on SGAN.

Acknowledgment

The author is grateful to the anonymous reviewers for their comments that improved the quality of this paper, also thankful to Md. Rokonuzzaman Sir from ISTT and Umme Habiba Islam for their support and help.

References

[1] Abir B, Mahal SN, Islam MS, Chakrabarty A (2019) Bangla handwritten character recognition with multilayer convolutional neural network. In: Advances in Data and Information Sciences, Springer, pp 155–165
[2] Akhand M, Rahman MM, Shill P, Islam S, Rahman MH (2015) Bangla handwritten numeral recognition using convolutional neural network. In: Electrical Engineering and Information Communication Technology (ICEEICT), 2015 International Conference on, IEEE, pp 1–5
[3] Bengio Y, Lamblin P, Popovici D, Larochelle H (2007) Greedy layer-wise training of deep networks. In: Advances in neural information processing systems, pp 153–160
[4] Bhattacharya U, Chaudhuri BB (2009) Handwritten numeral databases of indian scripts and multistage recognition of mixed numerals. IEEE transactions on pattern analysis and machine intelligence 31(3):444–457
[5] Bhowmik S, Malakar S, Sarkar R, Basu S, Kundu M, Nasipuri M (2018) Off-line bangla handwritten word recognition: a holistic approach. Neural Computing and Applications pp 1–16
[6] Bhowmik TK, Ghanty P, Roy A, Parui SK (2009) Svm-based hierarchical architectures for handwritten bangla character recognition. International Journal on Document Analysis and Recognition (IJDAR) 12(2):97–108
[7] Bozinovic RM, Srihari SN (1989) Off-line cursive script word recognition. IEEE Transactions on pattern analysis and machine intelligence 11(1):68–83
[8] Broumandnia A, Shanbehzadeh J, Varnoosfaderani MR (2008) Persian/arabic handwritten word recognition using m-band packet wavelet transform. Image and Vision Computing 26(6):829–842

[9] Bunke H (2003) Recognition of cursive roman handwriting: past, present and future. In: Document Analysis and Recognition, 2003. Proceedings. Seventh International Conference on, IEEE, pp 448–459
[10] Bunke H, Bengio S, Vinciarelli A (2004) Offline recognition of unconstrained handwritten texts using hmms and statistical language models. IEEE transactions on Pattern analysis and Machine intelligence 26(6):709–720
[11] Chowdhury AMS, Rahman MS (December 2016) Towards optimal convolutional neural network parameters for bengali handwritten numerals recognition. In: Proceedings of 19th International Conference on Computer and Information Technology (ICCIT), Dhaka, pp 431–436
[12] Dehghan M, Faez K, Ahmadi M, Shridhar M (2001) Handwritten farsi (arabic) word recognition: a holistic approach using discrete hmm. Pattern Recognition 34(5):1057–1065
[13] Donahue J, Krähenbühl P, Darrell T (2016) Adversarial feature learning. arXiv preprint arXiv:160509782
[14] El Qacimy B, Kerroum MA, Hammouch A (2015) Word-based arabic handwritten recognition using svm classifier with a reject option. In: Intelligent Systems Design and Applications (ISDA), 2015 15th International Conference on, IEEE, pp 64–68
[15] Koerich AL, Sabourin R, Suen CY (2005) Recognition and verification of unconstrained handwritten words. IEEE Transactions on Pattern Analysis and Machine Intelligence 27(10):1509–1522
[16] Krizhevsky A, Sutskever I, Hinton GE (2012) Imagenet classification with deep convolutional neural networks. In: Advances in neural information processing systems, pp 1097–1105
[17] LeCun Y, Boser B, Denker JS, Henderson D, Howard RE, Hubbard W, Jackel LD (1989) Backpropagation applied to handwritten zip code recognition. Neural computation 1(4):541–551
[18] LeCun Y, Boser BE, Denker JS, Henderson D, Howard RE, Hubbard WE, Jackel LD (1990) Handwritten digit recognition with a backpropagation network. In: Advances in neural information processing systems, pp 396–404
[19] LeCun Y, Jackel L, Bottou L, Brunot A, Cortes C, Denker J, Drucker H, Guyon I, Muller U, Sackinger E, et al (1995) Comparison of learning algorithms for handwritten digit recognition. In: International conference on artificial neural networks, Perth, Australia, vol 60, pp 53–60
[20] Liu CL, Koga M, Fujisawa H (2002) Lexicon-driven segmentation and recognition of handwritten character strings for japanese address reading. IEEE Transactions on Pattern Analysis & Machine Intelligence (11):1425–1437
[21] Madhvanath S, Govindaraju V, Ramanaprasad V, Lee DS, Srihari SN (1995) Reading handwritten us census forms. In: Document Analysis and Recognition, 1995., Proceedings of the Third International Conference on, IEEE, vol 1, pp 82–85

[22] Odena A (2016) Semi-supervised learning with generative adversarial networks. arXiv preprint arXiv:160601583
[23] Odena A, Olah C, Shlens J (2016) Conditional image synthesis with auxiliary classifier gans. arXiv preprint arXiv:161009585
[24] Pal U, Roy K, Kimura F (2009) A lexicon-driven handwritten city-name recognition scheme for indian postal automation. IEICE transactions on information and systems 92(5):1146–1158
[25] Pal U, Roy RK, Kimura F (2012) Multi-lingual city name recognition for indian postal automation. In: 2012 International Conference on Frontiers in Handwriting Recognition (ICFHR 2012), IEEE, pp 169–173
[26] Radford A, Metz L, Chintala S (2015) Unsupervised representation learning with deep convolutional generative adversarial networks. arXiv preprint arXiv:151106434
[27] Sazal MMR, Biswas SK, Amin MF, Murase K (2014) Bangla handwritten character recognition using deep belief network. In: Electrical Information and Communication Technology (EICT), 2013 International Conference on, IEEE, pp 1–5
[28] Sharma S (2018) Activation functions: Neural networks. URL https://towardsdatascience.com/activation-functions-neural-networks-1cbd9f8d91d6
[29] Srihari SN, Shin YC, Ramanaprasad V, Lee DS (1995) Name and address block reader system for tax form processing. In: Document Analysis and Recognition, 1995., Proceedings of the Third International Conference on, IEEE, vol 1, pp 5–10
[30] Srihari SN, Yang X, Ball GR (2007) Offline chinese handwriting recognition: an assessment of current technology. Frontiers of Computer Science in China 1(2):137–155
[31] Su T (2013) Chinese handwriting recognition: an algorithmic perspective. Springer Science & Business Media
[32] Szegedy C, Liu W, Jia Y, Sermanet P, Reed S, Anguelov D, Erhan D, Vanhoucke V, Rabinovich A (2015) Going deeper with convolutions. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 1–9
[33] Yacoubi AE (2001) Handwritten month word recognition on brazilian bank checks. In: Proceedings of the Sixth International Conference on Document Analysis and Recognition, IEEE Computer Society, p 972