Image Based Handwritten Bangla Word Recognition

1st Mehedi Hasan Raj
Department Of Electrical and Computer Engineering
North South University
Plot 15, Block-B, Bashundhara, Dhaka - 1229, Bangladesh.
mehedihasanraj007@gmail.com

2nd ANM Asifur Rahman
Department Of Electrical and Computer Engineering
North South University
Plot 15, Block-B, Bashundhara, Dhaka - 1229, Bangladesh.
anm.rahman@northsouth.edu

3rd Umma Habiba Akter
Department Of Electrical and Computer Engineering
North South University
Plot 15, Block-B, Bashundhara, Dhaka - 1229, Bangladesh.
habibaumme13@gmail.com

4th Khayrun Nahar
Department Of Electrical and Computer Engineering
North South University
Plot 15, Block-B, Bashundhara, Dhaka - 1229, Bangladesh.
khayrun.nahar@northsouth.edudek

***Abstract*—Handwritten word recognition is always a challenging thing in the field of Artificial Intelligence(AI). This recognition is more complicated when it is the matter of cursive characters, and Bangla has mixed cursive characters. It is the second most popular language in the Indian subcontinent and at 7th rank among 100 most spoken languages in the world. There are in total fifty(50) basic characters in Bangla, including, eleven(11) vowels and thirty nine(39) consonants. There are also 10 digits for bangla. Some other languages have excellent performance to recognize handwritten for their simplicity of the character, but there is no such existing method that has a satisfactory performance of Bangla handwritten recognition. In the existing methods, they have used some primitive Artificial Neural Network(ANN) models. These models are totally worthless for Bangla handwritten word recognition because of its nature. In this paper, we present a hybrid architecture approach for recognizing the bangla handwritten word. By our architecture, we can recognize not only the alphabets in a word, but also the combinations of digits as well as special characters. We use some most popular ANN models, such as Residual Neural Network(ResNet) and Bidirectional Long Short-Term Memory (BLSTM). By combining the both models as our architecture, we find an incredible performance for the recognition of the Bangla handwritten text.**

***Index Terms*—ResNet, BLSTM, OCR, Image Based Text Recognition**

I. Introduction

The world today is influenced greatly by computers, and virtually every important thing is stored electronically. Data between persons and machines must also be distributed easily and rapidly. Character recognition is a insistent subject of study since the 1960s. Since the problem is difficult, this remains an important area of research. Optical Character Recognition (OCR) has become a important and widely used technology. The various practical uses include scanners used in store check-out desks, currency changers, office scanners and attempted simplification of the delivery system. In industry and science, the technologies for Optical Character Recognition (OCR) and text recognition are now commonly used. The significant gain is effort and time that can be minimized through implementing this type of program. A paper-based text, for example, will take about 5 minutes to re-write. However, doing the same job at an accuracy rate of up to 95 percent will take about a hundred milliseconds. There are a number of games, software libraries, and commentary programs for OCR in international languages , including English, Chinese, and other similar languages.

The OCR methodology relies primarily on characteristic extraction and classification (based on patterns) of these characteristics. Handwritten OCR has been gaining growing interest as a subfield of OCR. On the basis of the input data, it is also classified as an offline and an online scheme. The offline approach is a static method in which input data is in the form of scanned images, while the online system feedback is more dynamic and is based on the movement of the pen tip with certain speed, angle, position and locus. A more advanced and developed online framework is thus named, since it fixes the overlapping issue of input data in the offline framework. When we hear about Optical character Recognition(OCR) first time that comes in our mind is Bangla hand written recognition .B.B Chowdhuri et al. [1] present for the first time, an OCR device to translate the written Bangla script is proposed. It is believed that there is Linotype font text in the paper that is valid for most Bangla documents. For text written on main paper, they obtained a high recognition rate. Device output is being examined on papers with a differing degree of noise. They also introduce very useful flowchart which is given in figure 1. Another researcher M.Alam et al. [2] proposed an approach of Bangla Scripts Recognition.

In our paper, we concern on Handwriten Bangla Recognition. we use Resnet based network for recognition which has 288 features. This paper is structured as follows: We define past work for OCR in Section 2. Section 3 describes the built experimental framework, and finally, in Section 4, we present the result and discussion.

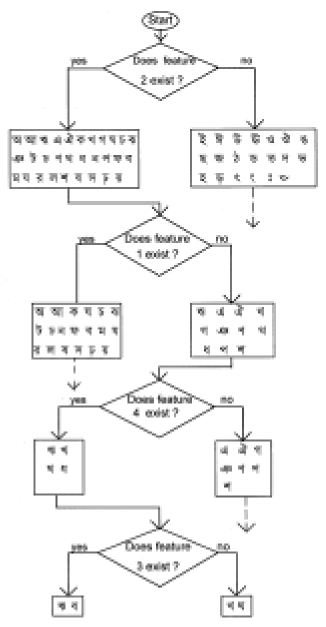

Fig. 1. Flow chart of Character Recognition .

II. Related Work

OCR is a very versatile method in different applications, such as natural language processing, automated data generation, automatic text processing, character readings etc. Characters in Bangla are nuanced, very similar in shape, and some of them can be written in different ways.In this paper [3],at first, they separated the Bangla document then process it by correction of skew, zoning line, segmentation of character and word. For the conversion of the digitized images which are gray tones into two-tone images, they using a histogram-based thresholding technique with digitized and prepared the data. A complete skew correction was got by them using Hough transformation. The upper, middle, lower zones and the middle or lower zones were demarcated by using the histogram and difficult. After that, for segmentation of words and characters line of the script was vertically scanned and plotted horizontally. To identify the simple characters, a feature-based tree classifier was used. When there are compound characters, they separated them into small blocks by using a tree classifier based on the features. Next, to distinguish individual characters, an effective schematic matching method is used. Shaded portions in t e symbol reflect the element. They analyzed the data with the corpus of 2.5 million words in Bangla. The rules of judgment are usually binary, say, presence/absence of the feature. From the available features, a semi-optimal tree classification was created. A criterion for recognition as a known character should be surpassed by the best matching score for the nominee. Otherwise, the claimant is ignored as unrecognized. They found different types of error and approach technique to detect and correct the errors. Their approach to error detection and correction is based on a dictionary search instead of n-gram statistics.They followed the same procedures for Devnagari.
Due to a strong distinction with the spellings, converting current structures of languages from Latin to alpha syllabary languages is difficult. Because of a cursive, the segmentation of graphical constituents corresponding to characters also becomes considerably difficult. Writing method and regular use of diacritics in languages of the alpha-syllabary family. In this paper [4], they suggest a graphemes-based marking scheme, linguistic word forming segments that linearizes segmentation within alpha-syllabary words. It introduces the first dataset of handwritten Bengali graphemes which is widely used in a daily context as part of the Bengali. AI Handwriting Grapheme Classification Challenge on Kaggle to benchmark vision algorithms for classification of multi-label graphemes. From competition procedures, we see that even though they are absent during preparation, deep learning strategies may generalize to a wide range of uncommon graphemes.
A two-stage recognition method was proposed by Bhattacharya et al. In this work [5], consisting of 50. Groups of specific Bangla characters, in which a regularly spaced rectangular grid consists of on the character bounding box and function vector, horizontal and vertical lines are overlaid for. The first classifier is determined to evaluate the response of the first classifier to Identify the confusion between a pair of characters of distinct shapes. The misunderstanding is overcome by a second level of grouping, and another rectangular grid is overlaid over the character bounding box to measure the function vector, except this time the rectangular grid is irregularly spaced over the character bounding box with horizontal and vertical lines. The Modified Quadratic Discriminant Function (MQDF) classifier and MLP were used as classifiers in both levels.
In this paper [6], author recommended separate 12 architectures, which takes a hierarchical approach to the classification of section characters from words and MLP. They reduced total feature by selecting some patterns into 36 classes in the stage of segmentation. Then, the work procedure followed through combining identical characters into a single class and using three different extraction features for the processes.

III. Methodology

In this section, we try to explain our working methodology for our experiment.Our work process has been categorized into several following phases.

- Preprocessing
- Model Built
- Training The Model
- Performance Measurement Using Various Metrics

A. Image Preprocessing

It is very important to preprocess the image as the model learns based on images. Having lots of noises makes the model difficult to learn and helps to perform the worst. In our dataset, the number of images is almost 8M where each image contains a unique world and corresponding their name. In the beginning, we convert each image to a 32x128 dimension.

Also converted this image to a grayscale image so that we do not need to do the channel normalization. Then, As each image name contains 3 components including index, label name, and type of the image, by using the splitting method, We process the label from there. After that to make the model's prediction easy, we extract and filter the grapheme from the images and a variable that contains garbage Unicode so that it becomes easy to predict for the model. Moreover, we do resize the input for the network 32x128x1 where the last number donates the channel dimension. Finally, we normalize our images and reshape the size into 1x32x128 so that we can perform various operations into the images by using a PyTorch popular deep learning framework.

B. Model Built

Model building is a fundamental part of the deep learning task. The performance of the model depends on how the model is defined. If the model is well defined, then it performs quite well. Our model is composed of Convolution Neural Network(CNN), Recurrent Neural Network(RNN), and Transcription Layer.

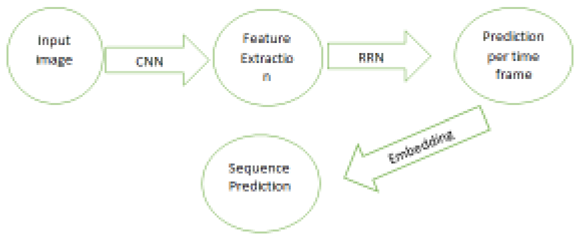

Fig. 2. Model Work flows .

1) Convulotion Neural Network(CNN): Features extraction is an important task for a successful machine learning project, and the convolution Neural Network (CNN) helps to extract the features. Without extracting the features we can not get our desire goal. In the CNN step, we try to follow the Resnet Architecture. In the Resnet Architecture, we try to do some operations [table 1] like convolution, batch normalization, Relu, Max pooling, and Resnet. We use convolution to decrease the image size, Relu as an activate function, and max pooling for decreasing the channel dimension.

2) Recurrent Neural Network(RNN) : Recurrent Neural Network performs better of the sequence if having an arbitrary length. A cell of RRN is composed of hidden layers with the input and output terminal. For each time frame, it gets a sequence, and after mathematical operations, it gives the output. There is a problem with traditional RRN. Having a vanishing gradient problem makes it hard to solve the sequencing problem and suffers a lot. We use a new technique Long Short Term Memory(LSTM) [8,7] that overcomes this problem and performs better than traditional RRN. We use the Bidirectional LSTM in our work and do some basic configuration[Table 2]. We use two bidirectional LSTM in our model. Then finally, we do the Transcription process in our model. Transcription is nothing but converting the prediction which made by RRN.

TABLE I

Operations	maps	Karnel size	Padding	Stride
MaxPool	-	(2,2)	(0,0)	(2,1)
Batch Normalization + Relu	-	-	-	-
Convolution 8	512	3x3	(1,1)	(1,1)
MaxPool	-	(2,2)	(0,0)	(2,1)
Batch Normalization + Relu	-	-	-	-
Convolution 7	512	3x3	(1,1)	(1,1)
Batch Normalization + Relu	-	-	-	-
Convolution 6	512	3x3	(1,1)	(1,1)
Batch Normalization + Relu	-	-	-	-
ResNet2	512	(3,3)	(1,1)	(1,1)
MaxPool		(2,2)	(1,1)	(2,1)
Batch Normalization + Relu	-	-	-	-
Convolution 5	512	3x3	(0,1)	(1,1)
MaxPool	-	2x2	(0,0)	(2,2)
Batch Normalization + Relu	-	-	-	-
Convolution 4	256	3x3	(1,1)	(1,1)
Batch Normalization + Relu	-	-	-	-
Convolution 3	128	3x3	(1,1)	(1,1)
Batch Normalization + Relu	-	-	-	-
ResNet1	128	(3, 3)	(1, 1)	(1, 1)
MaxPool	-	(2,2)	(0,0)	(2,2)
Batch Normalization + Relu	-	-	-	-
Convolution 2	128	3x3	(1,1)	(1,1)
Batch Normalization+Relu	-	-	-	-
Convolution 1	64	3x3	(1,1)	(1,1)
Input image	32x128	-	-	-

TABLE II

Operations	# units
Embedding	288
Bidirectional LSTM 2	256
Bidirectional LSTM 1	256

If we talk it mathematically, then it just finds the sequence of the label which has the highest probability. Finally, we do embedding with 288 output features.

C. Training the Model

After successfully built the model, we work on our training phase. In the training phase, we try the different number of epoch sizes but finally, we fixed it by the value 20 because of the better performance. Although we are having numerous optimizers, we prefer Adam as our optimizer because of its fast learning. By using it, the model can learn comparatively in less time and reduce our time cost.For our model we fixed the learning rate depending on epoch number.For epoch number less than 10, learning rate 0.001.If epoch number is less than 15, then learning rate 0.0003, otherwise 1e-5.For getting optimum global point, we use decreasing method of learning rate. We use our dataset into two following parts training set and validation set according to 80% and 20%. For every epoch, we make a prediction using the training set and update the gradient. Also, we define different types of batch sizes depending on the GPU limit of the google Colab. After that, we check the model performance by using the validation set. In both cases, we calculate our training and validation loss. After some epochs, we stop our training because we got our desire result. If we do not do this, there can be a possibility of overfitting.

D. Performance Measurement Using Various Metrics

A model can be good or bad is checked by using various metrics. Similarly, for validation of our model, we use different types of metrics like training loss, validation loss, training accuracy, f1 score, f2 score, and so on[Fig. 2][Fig. 3]. We get satisfied results of the model for every metric without test accuracy.

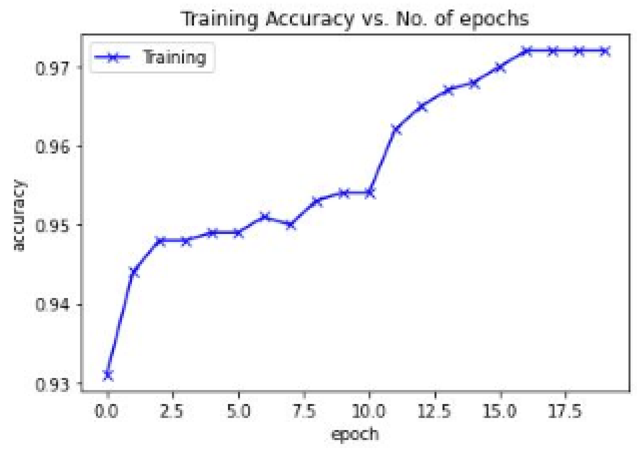

Fig. 3. Training Accuracy vs No. Of epoch.

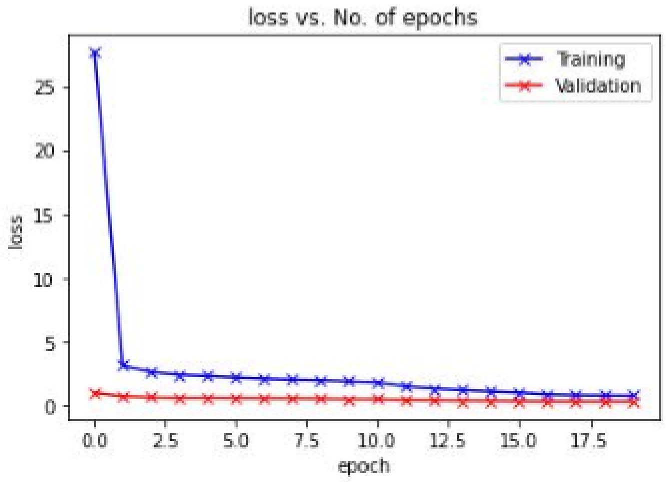

Fig. 4. Loss vs No. Of epoch.

IV. Result and Discussion

We also try to implement ResNet architecture in our model as it makes the model performs robust. We did not use data augmentation in our model as our data already cropped and small in size. We use various hyperparameter operations like different batch sizes, various numbers of epoch sizes, different types of hidden layers for the architecture, and also try different types of learning rates for the model. In our model, we get 97% accuracy in the validation phase, but unfortunately, we get the test accuracy 4%. Our implementation is good for the training set and validation set.

TABLE III

Epoch	accuracy	train_loss	valid_loss	total_wer	f1_micro
0	0.931	27.675	0.974	0.301	0.931
1	0.944	3.128	0.703	0.238	0.944
2	0.948	2.593	0.615	0.219	0.949
3	0.948	2.381	0.571	0.212	0.949
4	0.949	2.289	0.580	0.211	0.949
5	0.949	2.175	0.558	0.208	0.950
6	0.951	2.082	0.532	0.202	0.952
7	0.950	2.011	0.529	0.198	0.951
8	0.953	1.961	0.498	0.190	0.953
9	0.954	1.879	0.468	0.184	0.954
10	0.954	1.792	0.473	0.180	0.955
11	0.962	1.467	0.380	0.153	0.962
12	0.965	1.293	0.353	0.143	0.966
13	0.967	1.176	0.331	0.132	0.967
14	0.968	1.075	0.317	0.129	0.969
15	0.970	0.993	0.297	0.123	0.971
16	0.972	0.844	0.281	0.118	0.972
17	0.972	0.797	0.272	0.118	0.972
18	0.972	0.773	0.275	0.117	0.973
19	0.972	0.756	0.274	0.118	0.973

V. Conclusion

In this research, we have investigated that handwritten bangla words can be recognized with higher accuracy than earlier proposed architecture. we can recognize any Bangla word with the combinations of alphabets, digits and even special characters. The experimental results indicated that the combination of Residual Neural Network (ResNet) and Bidirectional Long Short-Term Memory(BLSTM) performs best. We achieved 97% accuracy in the validation step. For the testing step, we find 4% accuracy for Bangla handwritten word recognition. To the best of information, on this dataset, though the accuracy is too low, but this is the best recognition performance. In future, we will implement another Deep learning model such as the attention model on it, which will be more efficient for Bangla handwritten word detection.

References

[1] B. B. Chaudhuri and U. Pal, “A complete printed Bangla OCR system,” Pattern Recognit., vol. 31, no. 5, pp. 531–549, 1998, doi: 10.1016/S0031-3203(97)00078-2.

[2] M. Alam and M. A. Kashem, “A Complete Bangla OCR System for Printed Chracters,” vol. 01, no. 01, pp. 30–35, 2010.

[3] B. B. Chaudhuri and U. Pal, ”An OCR system to read two Indian language scripts: Bangla and Devnagari (Hindi),” Proceedings of the Fourth International Conference on Document Analysis and Recognition, Ulm, Germany, 1997, pp. 1011-1015 vol.2, doi: 10.1109/ICDAR.1997.620662.

[4] Alam, Samiul & Tahsin, Reasat & Sushmit, Asif & Siddiquee, Sadi & Rahman, Fuad & Hasan, Mahady & Humayun, Ahmed. (2020). Multi-label Classification of Common Bengali Handwritten Graphemes: Dataset and Challenge.

[5] Bhattacharya, Ujjwal & Shridhar, M. & Parui, Swapan & Sen, P. & Chaudhuri, Bidyut. (2012). Offline recognition of handwritten Bangla characters: An efficient two-stage approach. Pattern Analysis and Applications. 15. 10.1007/s10044-012-0278-6.

[6] Subhadip Basu, Nibaran Das, Ram Sarkar, Mahantapas Kundu, Mita Nasipuri, Dipak Kumar Basu, A hierarchical approach to recognition of handwritten Bangla characters, Pattern Recognition,

[7] F. A. Gers, N. N. Schraudolph, and J. Schmidhuber. Learning precise timing with LSTM recurrent networks. JMLR, 3:115–143, 2002.

[8] S. Hochreiter and J. Schmidhuber. Long short-term memory. Neural Computation, 9(8):1735–1780, 1997.
Volume 42, Issue 7,2009, Pages 1467-1484, ISSN 0031-3203,