2022 4th International Conference on Sustainable Technologies for Industry 4.0 (STI), 17-18 December, Dhaka

An Effective Approach for Bengali Handwritten Punctuation Recognition by Using a Low-cost Convolutional Neural Network

Azmain Yakin Srizon*, Md. Ali Hossain†, Abu Sayeed‡ and Md. Mehedi Hasan§
Department of Computer Science & Engineering
Rajshahi University of Engineering & Technology, Rajshahi, Bangladesh
Emails: *azmainsrizon@gmail.com, †ali.ruet@gmail.com, ‡abusayeed.cse@gmail.com, §mmehedihasann@gmail.com

***Abstract*—Bengali handwritten character recognition is considered difficult due to the complex curvatures and diverse variations of Bengali characters. Being the seventh most-spoken language of the world, Bengali has received some worth-mentioning contributions in this domain with the creation of different datasets such as NumtaDB, BanglaLekha, CMATERdb, and so on. However, none of the previous studies solely focused on recognizing punctuation marks in Bengali literature. Punctuations are important in Bengali language as without them the meaning of the whole sentence can be changed drastically. Another challenge in handwritten character recognition is the lack of low-cost convolutional neural networks (CNN). Recent studies suggested remarkable performance utilizing transfer learning approaches. However, due to the high computation of such methods, they are not suitable for recognizing in real-time via mobile devices. To solve this dilemma, in this study, we have proposed a low-cost CNN for the recognition of handwritten punctuations. Experimental results showed that our proposed architecture achieved an overall accuracy of 96.25% and outperformed popular transfer learning approaches such as DenseNet-201, InceptionV3, Xception, and EfficientNet-B6 by a significant margin with the least recognition time and the least number of parameters. Moreover, to solve the lack of dataset problem in this domain, we have collected 50 handwritten samples for each of the 28 Bengali punctuation marks in this study and formed a new dataset called 'BiramDB' (In Bengali, 'Biram' signs means punctuations). This research will be beneficial for diverse applications such as converting handwritten poems, stories, historical documents, and other writings into machine-encoded texts.**

***Index Terms*—BiramDB, Low-cost Convolutional Neural Network, Handwritten Character Recognition, DenseNet-201, InceptionV3, Xception, EfficientNet-B6**

I. Introduction

Optical character recognition (OCR) has been an active area of research for decades that focuses on converting handwritten or printed texts into machine-encoded texts from scanned or image documents [1]. In simpler terms, handwritten character recognition or cursive OCR is a sub-domain of OCR that converts handwritten characters into machine-encoded characters in real-time and is considered more difficult than recognizing printed characters due to complex curvatures and multiple writing fashions. Handwritten character recognition can be utilized for various applications such as recognizing texts on handwritten letters, envelopes, bank checks, historical newspapers and documents, historical manuscripts and books, signboards, business cards, and so on. Over the past decades, researchers have proposed diverse approaches for accurate character recognition. However, the performance of these approaches varies with language as each language has its own set of characters.

Among the 7151 languages of the world [2], Bengali is ranked seventh among the most-spoken languages and fifth as the most-spoken native language of the world with approximately 300 million native and almost 37 million second-language speakers [3], [4]. There exists almost 400 characters in Bengali literature that are currently in practice [5]. Bengali not only has numerals and basic characters (vowels and consonants) but also has compound characters, punctuations, and special characters. The complex curvature, diverse writing fashions, and similarity among Bengali characters have made the handwritten Bengali character recognition more difficult.

In recent years, much emphasis has been provided on the accurate recognition of Bengali numerals, basic letters, and compound characters [6], [7], [8], [9]. However, not much emphasis has been given to recognizing the punctuations of Bengali literature. In fact, a balanced and complete dataset of Bengali punctuation still does not exist in this domain of research. Like other popular languages, punctuation in Bengali literature is essential for expressing the emotions, attitudes, and moods of the speaker. Numerous Bengali writers and poets have been using punctuations in their writings and poems for centuries to strengthen arguments, express emotions, and communicate with the readers [10], [11]. Moreover, punctuations are important for natural language processing since the entire meaning of a sentence can be changed if the punctuations are wrongly used. Furthermore, with the increasing demand for text-to-speech conversion with natural voices, it's essential to recognize the punctuations to distinguish the pauses and emphasize specific portions of the texts. Solving this issue will enable us to lead the fourth industrial revolution regarding the involvement of the Bengali language in all aspects.

In this study, we have considered all 28 punctuations of Bengali literature that are currently in practice. We have collected 50 handwritten samples of each of the considered punctuations. In Bengali, punctuations are called 'Biram' signs or 'Joti' signs. That's why we have named the dataset BiramDB (Biram Database). To recognize the handwritten Bengali punctuations, we have proposed a low-cost deep convolution neural

978-1-6654-9045-0/22$31.00 © 2022 IEEE

TABLE I: Punctuation marks of Bengali literature along with their signs and pause times

Serial	Punctuation Name	Symbol / Sign	Pause Time
1	Comma	,	Time needed to say '1'
2	Semicolon	;	Double the time needed to say '1'
3	Full Stop	।	1 second
4	Note of Interrogation	?	1 second
5	Note of Exclamation	!	1 second
6	Colon	:	1 second
7	Colon Dash	:–	1 second
8	Dash	–	1 second
9	Hyphen	-	No need to pause
10	Inverted Commas (2 classes)	" "	Time needed to say '1'
11	Quotation Mark (2 classes)	' '	1 second
12	First Brackets (2 classes)	( )	No need to pause
13	Second Brackets (2 classes)	{ }	No need to pause
14	Third Bracket (2 classes)	[ ]	No need to pause
15	Asterix	***	No need to pause
16	Dot	.	No need to pause
17	Three Dots	...	No need to pause
18	Slash	/	No need to pause
19	Equals	=	No need to pause
20	Greater Than	>	No need to pause
21	Less Than	<	No need to pause
22	Taka (Bangladeshi Currency)	Taka Sign	No need to pause
23	Root	√	No need to pause

network (CNN) that has only 349,628 (approximately 350K) parameters. Moreover, we have also used different and popular transfer learning approaches and showed that our proposed architecture utilizes the least number of parameters, requires the least recognition time, and outperforms popular transfer learning approaches in terms of overall accuracy, precision, recall, and f1-score. Finally, after presenting and discussing the results, we have commented on the causes of misclassifications and possible future works.

II. Literature Review

In this study, the focus was on solving two major problems related to Bengali handwritten character recognition. The first problem to address is the lack of a Bengali handwritten punctuations dataset. Throughout decades, Bengali literature has received many contributions in terms of handwritten characters datasets. NumtaDB focused on handwritten Bengali numerals recognition [12]. There are approximately 85,000 images in the NumtaDB dataset. BanglaLekha-Isolated dataset introduced 84 different Bengali characters including 50 basic characters, 10 numerals, and 24 selected compound characters [13]. CMATERdb, the largest Bengali handwritten dataset consists of 238 different classes including 50 basic letters, 10 numerals, 7 modifiers, and 171 compound characters [5], [14]. Despite these significant contributions, Bengali punctuations have not been addressed while designing these datasets. In one research, the authors have focused on recognizing only the 'slash' character to extract the dates in texts [15]. In another research, the authors have considered three punctuations (full stop, comma, and hyphen) while recognizing other characters [16]. Although many researchers acknowledged that punctuation recognition is necessary for accurate recognition of Bengali handwritten texts [17], [18], a complete and balanced punctuations dataset is still not available in the literature.

The other problem that we focused on is designing a low-cost convolutional neural network. Machine learning approaches have been employed for the last decades for handwritten character recognition [l4]. However, recent advances in deep learning showed that a better performance can be achieved by applying convolutional neural networks [19], [20]. In recent times, transfer learning has also shown remarkable results for computer vision applications including handwritten character recognition [6], [7], [8]. However, with the increase in performance, the cost of computation has increased too. Applications like handwritten character recognition are carried out by mobile devices in real-time and therefore, the recognition needs to be lightweight and fast. Very few researchers have focused on designing a low-cost convolutional neural network for handwritten character recognition [21], [9]. However, all of them were utilized for recognizing basic letters, numerals, and compound characters. A low-cost CNN capable of recognizing Bengali handwritten punctuation marks is still not present in literature.

Keeping the two above-mentioned problems in mind, we have proposed two solutions in this paper. Firstly, we have created a dataset consisting of 28 different punctuation marks, each having 50 samples. Secondly, we have proposed a low-cost convolution neural network suitable for character recognition in real-time via mobile devices. More on these two aspects have been discussed in the next section.

Fig. 1: Sample images of each punctuation in the BiramDB dataset.

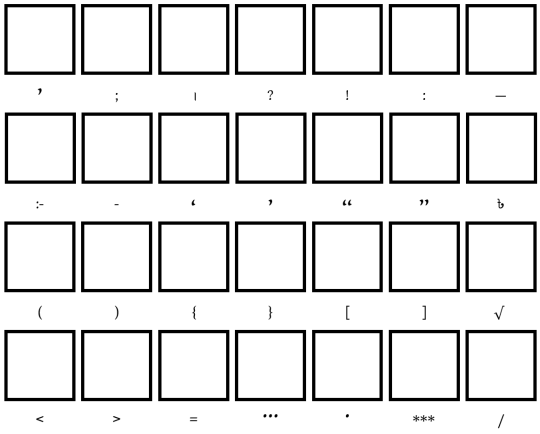

Fig. 2: Form for data collection.

III. Materials and Methods

In this section, first, the data collection has been described. Next, data augmentation has been described. After that, details on the proposed convolutional neural network have been presented. Finally, transfer learning has been briefly explained.

A. Data Collection and Processing

As mentioned earlier, previously, many works have focused on the recognition of numerals, basic letters, and compound characters. In very few recent works, the authors have considered one to three punctuation marks. However, a complete and balanced database of punctuations was not present in the literature. To overcome this lack of database, we have collected 50 samples for each of the 28 considered punctuations and created BiramDB (Biram Database) consisting of 1400 samples in total. Sample images of each punctuation in the BiramDB dataset are illustrated in Fig. 1. The punctuation marks considered for this study along with the pause time have been shown in Table I. It can be noticed that serial 10 to 14 of Table I need 2 classes each as the opening and closing signs are different. Furthermore, we have ignored the apostrophe sign as it's similar to a comma with the only difference in the position of the punctuation mark.

To collect the samples, we first created a form with 28 boxes in which the participants wrote the punctuation marks as illustrated in Fig. 2. After that, the whole written form was scanned and the handwritten punctuation marks were extracted from the scanned images. Next, all the extracted images were converted to binary images by using Otsu's method [22]. Then, the dataset was labeled and checked by multiple professionals to ensure that the labeling is correct. Finally, the dataset was split into train set and test set where the train set had 30 samples per class and the test set had 20 samples per class i.e., the total size of the train and test were 840 and 560 respectively.

B. Data Augmentation

It is a well-known phenomenon in deep learning that the performance of a deep convolutional neural network can be enhanced with the increase of data. Data augmentation is a process of augmenting or increasing the data by not removing the essential characteristics of the data. Rather it helps the model with additional learning. For example, let's consider a cat picture. If the picture is rotated 10 degrees to the left or right, it's still a cat picture. If the picture is zoomed in or out, it's still a cat picture. However, now with the new augmented cat pictures, the model will learn that if a rotated or scaled cat picture is provided, it's still a cat picture which implies that the learning has gotten better.

In our study, we have applied augmentation to the training dataset only. No augmentation was applied to the test dataset. The Augmentor library was utilized for applying augmentation [23]. Three types of augmentation techniques were applied i.e., rotation, zooming, and random distortion. While applying the rotation function, max left and right rotation were set to 10 with 50% probability. While applying the zoom function, maximum and minimum factors were set to 1.0 and 1.1 respectively with 50% probability. While applying the random distortion function, grid width and height were set to 4 with a magnitude of 8 and a 50% probability. After applying the augmentation process, 200 images were generated per class, a total of 5600 samples. Finally, the training dataset was split into the train set with 160 samples per class and the validation set with 40 samples per class. Note that, in this research validation set and test set are two different sets where the test set was prepared before applying augmentation and can be considered as the independent test set.

C. Proposed Convolutional Neural Network

The motivation behind convolutional neural networks (CNNs) was the interconnection of neurons that can be discovered in the animal vortex. Therefore, CNNs have been employed for the applications of computer vision and pattern recognition for the past decade [24]. It is a well-known phenomenon that with a larger and more complex CNN, a better result can be achieved. However, large and complex CNNs are not best suited for mobile devices due to high computation. In our case, Bengali handwritten character recognition will take place in real-time using mobile devices. To facilitate the users, the CNN used for our application has to be lightweight with low computation. Keeping the application in mind, we are proposing a low-cost CNN illustrated in Fig. 3.

The architecture takes a grayscale image as input with the height and width of 28 and 28 respectively which is much smaller than the conventional size of $224 \times 224 \times 3$ used by popular transfer learning approaches. Next, we have added three convolution blocks. In the first convolution block, there are two convolution layers of size 32 each followed by a

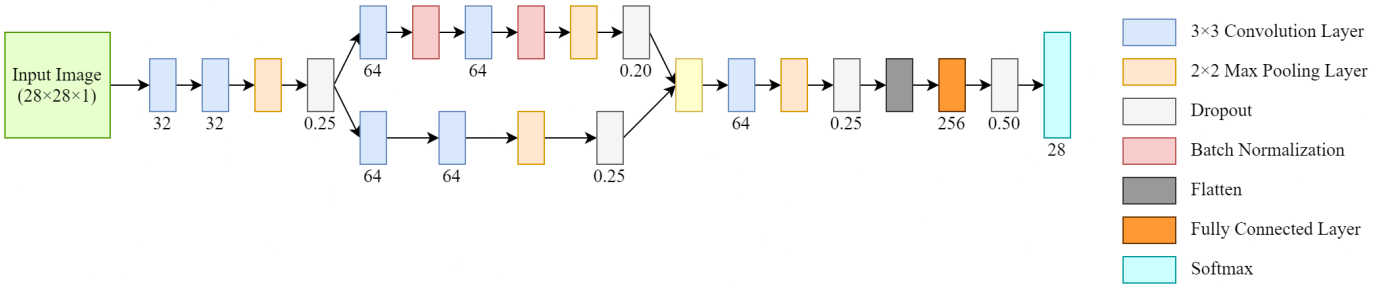

Fig. 3: Proposed low-cost convolutional neural network

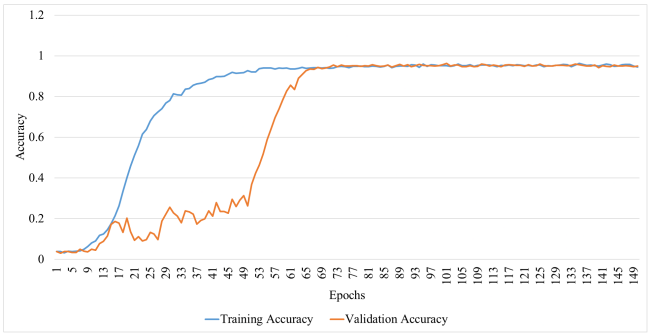

Fig. 4: Training accuracy vs. validation accuracy for 150 epochs while applying the proposed architecture

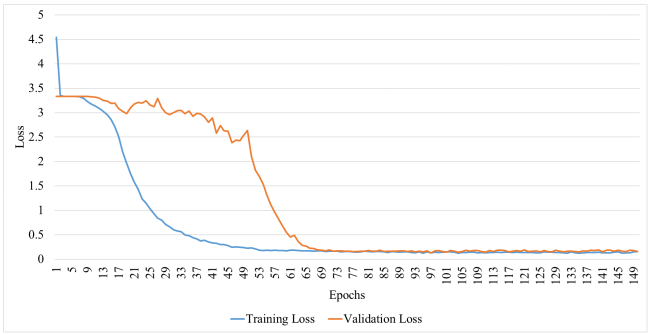

Fig. 5: Training loss vs. validation loss for 150 epochs while applying the proposed architecture

pooling layer and a dropout of 25%. The size of all filters used for convolution layers is 3×3 whereas the pool size is fixed to 2×2. Max pooling has been utilized here. The dropout technique has been used here to overcome the overfitting problem.

In the second convolution block, the network has been divided into two branches. In the first branch, two convolutional layers of size 64 followed by two batch normalization layers each have been added. Next, a max-pooling layer followed by a dropout of 20% is added. In the second branch, two convolutional layers of size 64 followed by a max-pooling layer and a dropout of 25% are added. After that, the outputs of both branches are concatenated. Here, batch normalization has been introduced to overcome the gradient vanishing problem that was occurring due to using consecutive convolution layers

TABLE II: Class-wise accuracy precision, recall and f1-score for the proposed CNN. Here, support=20 for all classes.

Classes	Accuracy	Precision	Recall	F1-Score
Comma	0.90	0.90	0.90	0.90
Semicolon	1.00	1.00	1.00	1.00
Full Stop	1.00	0.91	1.00	0.95
Note of Interrogation	1.00	1.00	1.00	1.00
Note of Exclamation	0.95	1.00	0.95	0.97
Colon	1.00	0.95	1.00	0.98
Colon Dash	0.95	1.00	0.95	0.97
Root	0.95	1.00	0.95	0.97
Hyphen	0.70	1.00	0.70	0.82
Inverted Commas Start	1.00	0.91	1.00	0.95
Inverted Commas End	0.85	1.00	0.85	0.92
Quotation Mark Start	0.95	1.00	0.95	0.97
Quotation Mark End	0.90	0.90	0.90	0.90
Dash	1.00	0.77	1.00	0.87
First Bracket Start	1.00	0.95	1.00	0.98
First Bracket End	0.95	0.95	0.95	0.95
Second Bracket Start	0.95	1.00	0.95	0.97
Second Bracket End	0.95	0.95	0.95	0.95
Third Bracket Start	1.00	1.00	1.00	1.00
Third Bracket End	0.95	1.00	0.95	0.97
Asterix	1.00	1.00	1.00	1.00
Three Dots	1.00	1.00	1.00	1.00
Dot	1.00	0.95	1.00	0.98
Slash	1.00	1.00	1.00	1.00
Equals	1.00	0.95	1.00	0.98
Greater Than	1.00	1.00	1.00	1.00
Less Than	1.00	0.95	1.00	0.98
Taka	1.00	1.00	1.00	1.00

for a small input image. Batch normalization converts the input into a range-based output (typically within 0 and 1) and therefore, prevents gradient vanishing problems. However, for some classes, the batch normalization technique was not producing a better result. For those classes, normal convolution layers were achieving a better result. That's why both techniques were adopted and branched. In this way, none of the valuable information gets vanished.

In the third convolution block, a convolution layer of size 64 followed by a max-pooling layer and a dropout of 25% is added. Next, after flattening the output a fully connected layer of size 256 followed by a dropout of 50% is added. Finally, an output layer of size 28 (number of classes) with 'softmax'

TABLE III: Comparison with popular transfer learning approaches in terms of overall accuracy, number of parameters and required time for prediction

Approaches	Accuracy	Parameters	Prediction Time
DenseNet-201	93.93%	18,375,772	8.9 milliseconds
Xception	91.61%	20,918,852	5.3 milliseconds
InceptionV3	93.21%	21,860,156	5.4 milliseconds
EfficientNet-B6	93.39%	41,024,683	10.7 milliseconds
Proposed CNN	**96.25%**	**349,628**	**2.7 milliseconds**

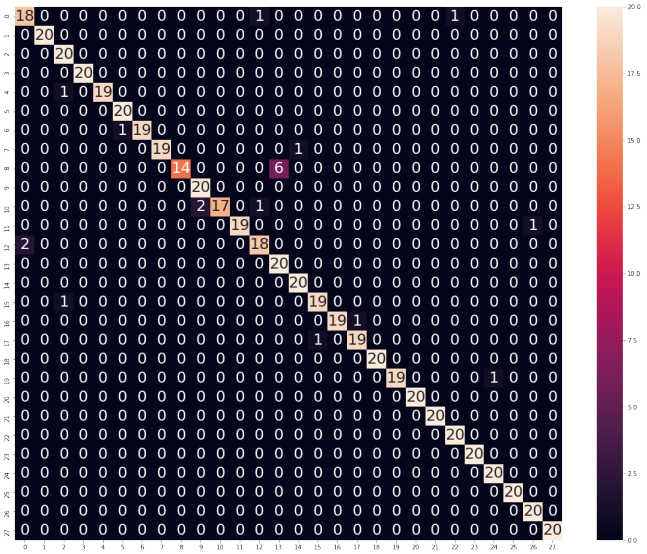

Fig. 6: Confusion matrix for the proposed convolutional neural architecture

activation is added for the final prediction. Apart from the output layer, all the convolution layers and dense layers of the architecture utilizes the 'ReLU' activation function. Due to the minimalistic design of the architecture, it needs only 349,628 parameters among which the number of trainable parameters is only 349,372.

D. Transfer Learning

In transfer learning, previous knowledge of recognition can be utilized for a similar recognition problem. For example, suppose, the previous problem was to recognize if a given picture is a truck or not and the new problem is if a given picture is a car or not. Both cars and trucks have some similar characteristics such as both being vehicles, both having four wheels, and so on. That means the knowledge retrieved while recognizing trucks can be reused while recognizing cars with slight modification. On a computation basis, for the new problem, no long-term training is necessary, and therefore, the recognition time can be reduced by a significant margin.

With the invention of the 'ImageNet' dataset, many works have been utilizing the ImageNet weights for different transfer learning methods. In this literature, we have adopted 4 popular transfer learning approaches. The first model is DenseNet-201 [25] which has a small number of parameters than the other transfer-learned methods but produces a better result due to the dense connectivity among neurons. The second architecture is InceptionV3 [26] which was introduced by Google and performs faster computation than others. The third model is Xception [27] or Extreme Inception which is an update of InceptionV3. The last architecture under consideration is EfficientNet-B6 [28] which performs well but has high computation.

IV. Experimental Analysis

A. Experimental Setup

While applying the proposed convolutional neural network, the batch size was set to 512. 'Adam' optimizer was utilized for optimization. The categorical cross-entropy function was used for calculating the loss. The proposed architecture was run for 150 epochs. The learning rate was kept at 0.001 for the first 50 epochs. For the next 40, 30, 20, and 10 epochs the learning rate was kept at 0.0001, 0.00001, 0.000005, and 0.000001 respectively. The dropout technique was used for minimizing overfitting.

For the transfer learning approaches (DenseNet-201, Xception, InceptionV3, and EfficientNetB6), the batch size was kept at 24 due to the machine constraints. The learning rate was 0.00001 and the number of epochs was fixed to 25. The values of other hyperparameters were the same as the values of hyperparameters used for the proposed architecture. The values of the hyperparameters were decided based on the grid search technique. All the experiments were run in a Kaggle GPU environment with 12 GB VRAM.

B. Experimental Results

We applied the proposed architecture to the BiramDB dataset and it achieved 96.25% overall accuracy. Fig. 4 illustrates the training accuracy vs. validation accuracy curve whereas Fig. 5 illustrates the training loss vs. validation loss curve. The confusion matrix has been presented in Fig. 6. Table II shows the class-wise accuracy, precision, recall and f1-score.

The results of transfer learning approaches along with the number of parameters and time required for prediction have been shown in Table III. It can be observed that DenseNet-201 architecture has achieved the highest overall accuracy of 93.93% among all the four transfer learning approaches in consideration. However, still it couldn't outperform the performance achieved by the proposed architecture. From Table III, it can be noticed that our proposed architecture needs the least number of parameters and least time for recognition with the highest overall accuracy.

C. Comments on Misclassifications

In our dataset, there are some classes that are extremely similar. For example, hyphens and dashes are very similar to one another. Commas and ending quotation marks are also very similar. If we look closely at the misclassifications, it can be seen that the 'Hyphen' class has the lowest accuracy of 70% only. However, if we remove some of the similar classes,

the performance improves. For example, if we consider 26 classes by removing the 'hyphen' and the 'ending quotation mark' classes, the overall accuracy jumps to 99.42% which proves that the model itself is very capable of distinguishing among the classes but due to the similarity of some classes, the misclassifications are happening.

Furthermore, if we calculate top-3 accuracy for 28 classes, the performance jumps to 100%. This denotes that our proposed architecture is capable of providing correct predictions within three tries.

V. Conclusion

In this paper, we focused on the two major problems of Bengali handwritten punctuation recognition. The first one is the lack of a complete and balanced dataset for Bengali handwritten punctuations and the second one is the lack of a low-cost convolutional neural network for handwritten character recognition. To solve the first problem, we collected 1400 samples for 28 Bengali punctuation marks and created a new balanced and complete dataset called BiramDB. Next, we have proposed a low-cost convolutional neural network that utilizes approximately 350K parameters which is less than the popular deep learning and transfer learning approaches used in previous works. The experimental results showed that our proposed architecture needs the least prediction time, and the least number of parameters and achieves a better accuracy of 96.25% than popular and mainstream deep models. Further investigation on misclassifications revealed that most of the misclassifications were happening due to the similarity among characters which also proved that our model is capable of distinguishing between the punctuations accurately. In the future, we will focus on collecting samples for all 400 characters in Bengali literature and proposing a low-cost CNN for a complete Bengali handwritten character recognition.

References

[1] S. Mori, H. Nishida, and H. Yamada, *Optical character recognition*. John Wiley & Sons, Inc., 1999.

[2] D. Berhard, G. Simons, and C. Fennig, "Ethnologue: Languages of the world. twenty," 2020.

[3] C. I. Agency, *The world factbook 2009*. Government Printing Office, 2009.

[4] D. Eberhard, G. Simons, and C. Fennig, "Ethnologue: languages of the world, 22nd edn dallas," *TX: SIL International*, 2019.

[5] N. Das, K. Acharya, R. Sarkar, S. Basu, M. Kundu, and M. Nasipuri, "A benchmark image database of isolated bangla handwritten compound characters," *International Journal on Document Analysis and Recognition (IJDAR)*, vol. 17, no. 4, pp. 413–431, 2014.

[6] M. M. Hasan, A. Y. Srizon, A. Sayeed, and M. A. M. Hasan, "Handwritten numerals recognition by employing a transfer learned deep convolution neural network for diverse literature," in *2020 11th International Conference on Electrical and Computer Engineering (ICECE)*. IEEE, 2020, pp. 431–434.

[7] M. Hasan, A. Y. Srizon, A. Sayeed, and M. A. M. Hasan, "Bengali handwritten compound characters recognition by utilizing transfer learning," in *2020 11th International Conference on Electrical and Computer Engineering (ICECE)*. IEEE, 2020, pp. 218–221.

[8] M. M. Hasan, A. Y. Srizon, A. Sayeed, and M. A. M. Hasan, "Bengali handwritten isolated compound characters recognition by applying transfer learned deep convolutional neural network," in *2020 23rd International Conference on Computer and Information Technology (ICCIT)*. IEEE, 2020, pp. 1–6.

[9] A. Sayeed, J. Shin, M. Hasan, A. Mehedi, A. Y. Srizon *et al.*, "Bengalinet: A low-cost novel convolutional neural network for bengali handwritten characters recognition," *Applied Sciences*, vol. 11, no. 15, p. 6845, 2021.

[10] M. Alvi, "A warm modernist: Naked lonely hand: Selected poems," *PN Review*, vol. 30, no. 3, 2004.

[11] P. P. Ray *et al.*, "Publication of tagore's song offerings, the gitanjali: A study," *Annals of Library and Information Studies (ALIS)*, vol. 60, no. 1, pp. 15–21, 2013.

[12] S. Alam, T. Reasat, R. M. Doha, and A. I. Humayun, "Numtadb-assembled bengali handwritten digits," *arXiv preprint arXiv:1806.02452*, 2018.

[13] M. Biswas, R. Islam, G. K. Shom, M. Shopon, N. Mohammed, S. Momen, and A. Abedin, "Banglalekha-isolated: A multi-purpose comprehensive dataset of handwritten bangla isolated characters," *Data in brief*, vol. 12, pp. 103–107, 2017.

[14] S. Roy, N. Das, M. Kundu, and M. Nasipuri, "Handwritten isolated bangla compound character recognition: A new benchmark using a novel deep learning approach," *Pattern Recognition Letters*, vol. 90, pp. 15–21, 2017.

[15] R. Mandal, P. P. Roy, and U. Pal, "Bangla date field extraction in offline handwritten documents," in *Proceeding of the workshop on Document Analysis and Recognition*, 2012, pp. 37–41.

[16] N. Majid and E. H. B. Smith, "Segmentation-free bangla offline handwriting recognition using sequential detection of characters and diacritics with a faster r-cnn," in *2019 International Conference on Document Analysis and Recognition (ICDAR)*. IEEE, 2019, pp. 228–233.

[17] F. Sadaf, S. T. U. Raju, and A. Muntakim, "Offline bangla handwritten text recognition: A comprehensive study of various deep learning approaches," in *2021 3rd International Conference on Electrical & Electronic Engineering (ICEEE)*. IEEE, 2021, pp. 153–156.

[18] S. Chakraborty and S. Paul, "Bengali handwritten character transformation: basic to compound and compound to basic using convolutional neural network," in *2021 2nd international conference on robotics, electrical and signal processing techniques (ICREST)*. IEEE, 2021, pp. 142–146.

[19] M. M. Islam, A. Das, I. Kowsar, A. S. A. Rabby, N. Hasan, and F. Rahman, "Towards building a bangla text recognition solution with a multi-headed cnn architecture," in *2021 IEEE International Conference on Big Data (Big Data)*. IEEE, 2021, pp. 1061–1067.

[20] R. Bhattacharya, S. Malakar, S. Ghosh, S. Bhowmik, and R. Sarkar, "Understanding contents of filled-in bangla form images," *Multimedia Tools and Applications*, vol. 80, no. 3, pp. 3529–3570, 2021.

[21] A. S. A. Rabby, S. Haque, S. Abujar, and S. A. Hossain, "Ekushnet: using convolutional neural network for bangla handwritten recognition," *Procedia computer science*, vol. 143, pp. 603–610, 2018.

[22] M. Sezgin and B. Sankur, "Survey over image thresholding techniques and quantitative performance evaluation," *Journal of Electronic imaging*, vol. 13, no. 1, pp. 146–165, 2004.

[23] M. D. Bloice, C. Stocker, and A. Holzinger, "Augmentor: An image augmentation library for machine learning," *arXiv e-prints*, pp. arXiv–1708, 2017.

[24] M. V. Valueva, N. Nagornov, P. A. Lyakhov, G. V. Valuev, and N. I. Chervyakov, "Application of the residue number system to reduce hardware costs of the convolutional neural network implementation," *Mathematics and Computers in Simulation*, vol. 177, pp. 232–243, 2020.

[25] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 4700–4708.

[26] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 2818–2826.

[27] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 1251–1258.

[28] M. Tan and Q. Le, "Efficientnet: Rethinking model scaling for convolutional neural networks," in *International conference on machine learning*. PMLR, 2019, pp. 6105–6114.