第 48 卷 第 12 期
2021 年 12 月

湖南大学学报（自然科学版）
Journal of Hunan University (Natural Sciences)

Vol. 48 No. 12
December 2021

A Deep Convolution Neural Network-Based SE-ResNext Model for Bangla Handwritten Basic to Compound Character Recognition

Mohammad Meraj Khan[1], Mohammad Shorif Uddin[2], Mohammad Zavid Parvez[1], Lutfur Nahar[2], Jia Uddin[3]

[1] Department of Computer Science and Engineering, BRAC University, Dhaka, Bangladesh

[2] Department of Computer Science and Engineering, Jahangirnagar University, Dhaka, Bangladesh

[3] AI and Big Data Department, Endicott College, Woosung University, Daejeon, South Korea

Abstract: With the recent advancement in artificial intelligence, the demand for handwritten character recognition increases day by day due to its widespread applications in diverse real-life situations. As Bangla is the world's 7th most spoken language, hence the Bangla handwritten character recognition is demanding. In Bangla, there are basic characters, numerals, and compound characters. Character identicalness, curviness, size and writing pattern variations, lots of angles, and diversity make the Bangla handwritten character recognition task challenging. Recently, few papers have been published which study Bangla numeral, basic, and handwritten compound characters and the accuracy level in all three areas. The main objective of this paper is to propose a novel model that performs equally outstanding in all three different character types and to increase the efficiency of building a real-world Bangla handwritten character recognition system. This work describes a novel method for recognizing Bangla basic to compound character using a very special deep convolutional neural network model known as Squeeze-and-Excitation ResNext. The architectural novelty of our model is in introducing the Squeeze and Excitation (SE) Block, a very simple mathematical block with simple computation but very effective in finding complex features. We obtained 99.80% accuracy from a benchmark dataset of Bangla handwritten basic, numerals, and compound characters containing 160,000 samples. Additionally, our model demonstrates outperforming results compared to other state-of-the-art models.

Keywords: Bangla handwritten-character recognition, Deep Convolutional Neural Network, Squeeze and Excitation ResNext, global average pooling.

一种基于深度卷积神经网络的东南-水库下一个 模型，用于孟加拉语手写基础到复合字符识别

摘要：随着最近人工智能的进步，由于手写字符识别在各种现实生活中的广泛应用，对手写字符识别的需求与日俱增。由于孟加拉语是世界第七大语言，因此孟加拉语手写字符识别要求很高。在孟加拉语中，有基本字符、数字和复合字符。字符的一致性、曲线、大小和书写模式的变化、大量的角度和多样性使孟加拉手写字符识别任务具有挑战性。最近，很少有论文研究孟加拉数字、基本和手写复合字符以及这三个领域的准确度水平。本文的主要目的是提出一种新颖的模型，该模型在所有三种不同的字符类型中表现同样出色，并提高构建真实孟加拉手写字符识别系统的效率。这项工作描述了一种使用称为 挤压和激发 下一个。的非常特殊的深度卷积神经网络模型识别 孟加拉语基本到复合字符的新方法。我们模型的架构新颖之处在于引入了挤压和激发 (东南) 模块，这是一个非常简单的数学模块，计算简单，

Received: August 11, 2021 / Revised: October 7, 2021 / Accepted: November 8, 2021 / Published: December 30, 2021

About the authors: Mohammad Meraj Khan, Department of Computer Science and Engineering, BRAC University, Dhaka, Bangladesh; Mohammad Shorif Uddin, Department of Computer Science and Engineering, Jahangirnagar University, Dhaka, Bangladesh; Mohammad Zavid Parvez, Department of Computer Science and Engineering, BRAC University, Dhaka, Bangladesh; Lutfur Nahar, Department of Computer Science and Engineering, Jahangirnagar University, Dhaka, Bangladesh; Jia Uddin, AI and Big Data Department, Endicott College, Woosung University, Daejeon, South Korea

但在查找复杂特征方面非常有效。我们从包含 160,000 个样本的孟加拉手写基本、数字和复合字符的基准数据集中获得了 99.80% 的准确率。此外，与其他最先进的模型相比，我们的模型表现出更好的结果。

关键词: 孟加拉语手写字符识别、深度卷积神经网络、挤压和激发 下一个、全局平均池化.

1. Introduction

Bangla is one of the most spoken languages, with approximately 228 million native speakers and about 37 million second-language speakers. It is the fifth most-spoken native language and the seventh most spoken language by a total number of speakers in the world. Bangla is the official language and mother tongue in Bangladesh and the West Bengal State of India. It has a very rich history of more than a thousand years. There are 50 basic characters, 11 numerals, and around 300 compound characters in the Bangla language [1]. The basic character set contains 11 vowels and 39 consonants. Table 1 represents the printed version of basic characters, and Table 2 represents the corresponding handwritten version. Two or more characters together form a compound character. The printed version of some frequently used compound characters and their corresponding handwritten version is shown in Table 3 and Table 4, respectively.

In most cases, the compound character does not retain the actual shape of the basic characters from which it is made of. A few examples of different compound character formations are depicted in Table 5. In this table, we observe that the position of the basic character which took part to form the compound character also changes the shape. In some cases, the compound character has no similarities with the basic characters. The fourth row in Table 5 shows such a compound character "ঙ্গ". If we look closely into the second and third-row, where there are two compound characters, "ষ্প" and "ষ্ণ" formatted from two basic characters, the shape of the basic character is partially retained. One important thing to notice here is that even though the same basic character "ষ" is common for both compound characters, its shape is different in the resulting compound character. The last row shows a compound character "ন্ত্র", its formation is much complex than the rest characters in Table 5, where none of the three basic character shapes is retained. In this table, only the compound character retains the basic character shape in the first row. Character identicalness, variations, and writing patterns sometimes make different characters similar looking. Table 6 shows a few similar-looking different characters.

Table 1 Bangla numerals and basic characters (1st-row numerals – 2nd-row vowels, 3rd-6th rows consonants)

০	১	২	৩	৪	৫	৬	৭	৮	৯	
অ	আ	ই	ঈ	উ	ঊ	ঋ	এ	ঐ	ও	ঔ
ক	খ	গ	ঘ	ঙ	চ	ছ	জ	ঝ	ঞ	ট
ঠ	ড	ঢ	ণ	ত	থ	দ	ধ	ন	প	ফ
ব	ভ	ম	য	র	ল	শ	ষ	স	হ	ড়
ঢ়	য়	ৎ	ং	ঃ	ঁ					

Table 2 Handwritten Bangla numerals and basic characters are shown in Table 1 (1st-row numerals, 2nd-row vowels, 3rd-6th rows consonants)

Table 3 A printed version of the considered Bangla compound characters

ক্ষ	ব্দ	ঙ্গ	হ্ম	স্ফ	স্থ	চ্ছ	ক্ত
ম্ন	ষ্ণ	ষ্প	প্ত	ম্ব	ণ্ড	ড্ড	ক্ষ্ম
হ্ন	ষ্ঠ	ল্ল	ষ্প	ন্দ	ন্ধ	ম্ম	ণ্ঠ

Table 4 Bangla compound characters' handwritten version shown in Table 3 (24 classes)

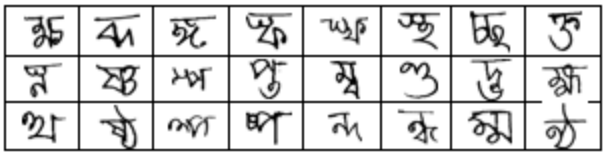
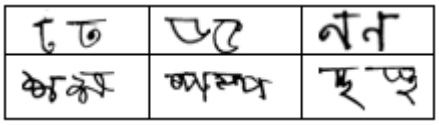


Table 5 Some examples to show the development of compound characters

Combinations		Compound Character
চ + ছ	=	চ্ছ
ষ + প	=	ষ্প
ষ + ণ	=	ষ্ণ
ঙ + গ	=	ঙ্গ
স + ত + র	=	স্ত্র

Table 6 Some similar-looking Bangla handwritten characters



Handwritten character recognition finds widespread applications in diverse fields, such as automated survey form data entry, vehicle number plate recognition, various documentation digitalization, bank cheque processing, OCR applications, etc.

There are many reasons behind the difficulty of the Bangla handwritten characters, such as similarities between characters, writing pattern variations, changed shape in a compound character, too much curviness, variations in sizes and shapes. The length of aces, degree of angles, the sizes of turns also make it difficult. The presence of 'Matra' (a horizontal line at the top) even makes it harder. Even a single dot makes a character looks like a different character. Recognition of all (combined numerals, basic and compound) characters is much difficult than only numeric characters, or only basic characters, or only compound characters due to an increase in the number of classes.

There are very few publications on combined Bangla handwritten characters compared to only basic or only numerals or only compound [1-10] handwritten characters recognition.

The most notable techniques used in these publications are supported vector machine (SVM) [1], multilayer perceptron (MLP) [2], convolutional neural network (CNN) [3, 4-10]. The accuracy level and the precision achieved by these models are not up to the mark to build a standard Bangla OCR. In a recent study in object recognition, a state-of-the-art model shows superior performance, impressed by the result, we have decided to explore, and for further improvement, we have decided to work on the variety of squeeze and excitation ResNext deep convolutional neural network models. This research is also a continuation of our previous studies [11], where we only recognized Bangla handwritten compound characters. These models are pretty accurate in determining the inter-channel feature dependencies, which in turn makes it efficient to learn very complex features. The main contribution of this work is to develop an effective deep learning technique to recognize a full set of Bangla characters (basic characters, numerals, and compound characters).

We have organized the paper in the following manner: Section 2 describes the research background, Section 3 discusses the proposed models' architecture, Section 4 analyzes the experiment and results, Section 5 draws conclusions.

2. Related Works

As English is the international language, so it is well studied, and techniques have already been developed for the English language. However, our main focus is on the Bangla language in this paper. Our research identified numerous publications conducted in various languages other than Bangla. The most mentionable studies covered Chinese, Arabic, English, Romanian, Japanese, Devanagari, Urdu, Pasto, Gujarati [12-21]. Bangla was also studied; the most notable results are discussed in [1-10]. In most of the early works, researchers depend on the process of hand-engineered feature extraction. However, these processes are erroneous, difficult to extract complex features and take time and effort. Also, the outcome is not satisfactory to recognize the characters efficiently.

MLP-based models [2] are inefficient and redundant. In MLP, each neuron from a layer is connected with every other neuron of the next layer, resulting in too many weights and overfitting. In an SVM Based model [1], if the dataset is complex and huge, handling correlation becomes difficult and cannot use spatial information in the recognition and detection tasks.

These models find it difficult to recognize similar shaped basic to compound characters. Hence, recognizing image-based characters is inefficient in a huge dataset environment. The accuracy level achieved so far by these methods is not satisfactory to build an OCR to use in real-life situations. As convolutional neural network (CNN) performs much better than these models in image recognition, many researchers explored CNN [3, 8-10] to recognize Bangla handwritten characters. Even though the simple CNN performs better than the above-mentioned models, the accuracy level is not satisfactory. Researchers have explored deep CNN models [4-7, 11, 22-25] in recent times. Deep CNN models proved their superiority over other models, and this has motivated us to investigate a model which outperforms all other architectures and overcome the limitations.

3. Our Proposed Model

The architecture of our proposed method is explained in detail in this section. Fig. 1 shows the simplified block diagram of our method.

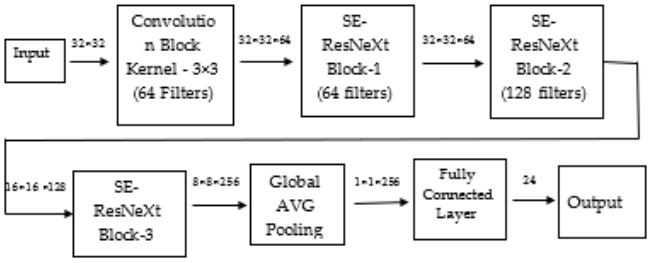


Fig. 1 The block diagram of the proposed model

The proposed model takes a 32×32 size image as input, and the final fully connected layer outputs the probabilities of 84 possible classes of 50 basic characters, ten numerals, and 24 compound characters. The first layer is a convolutional block followed by the first SE-ResNext Block, - the second and the third SE-ResNext Block, then a global average polling layer followed by the final fully connected layer. The detailed construction of the SE-ResNext block is described in detail below. Each of the SE-ResNext blocks is a combination of 3 stacked SE-ResNext layers.

3.1. The Block Construction of Squeeze and Excitation

The construction of squeeze and excitation block [26] can be achieved by simply stacking a squeeze operation and an excitation operation. It is a computationally less intensive unit, and it can be expressed by the transformation:

$$F_{tr} : \mathrm{X} \rightarrow \mathrm{U}, \mathrm{X} \in R^{H \times W \times C}$$

where F_{tr} denotes a convolutional operator performed on the input X that produces the output U, *H* and *W* are the input image height and width, respectively, and *C* represents the channel.

Let V = [$v_1, v_2, \ldots, v_c$] denotes the learned set of filter kernels, where refers to the parameters of the *c*-th filter. We can then write the outputs of as U = [$u_1, u_2, \ldots, u_c$], the output of all channels can be written as U = [$u_1, u_2, \ldots, u_c$], for the transformation F_{tr}, where

$$u_c = v_c * X = \sum_{s=1}^{c'} v_c^s * x^s \qquad (1)$$

Here v_c is the filter kernel, *X* is the input image, and * denotes the convolution operation. The term bias is not included in the equation for the sake of simplicity.

X could be $X = [x^1, x^2, \ldots, x^{c'}]$, and v_c could be $v_c = [v_c^1, v_c^2, \ldots, v_c^{c'}]$.

The output is produced by summing up all the operations in all channels, where v_c^s is a spatial kernel, which operates with the input *X* for that channel. The channel dependencies are embedded in v_c through spatial correlation. The transformation operation which retrieves the informative feature of inter-channel dependencies empowers the network. Fig. 2 shows the block diagram of the SE building block, which will be added with the ResNext model.

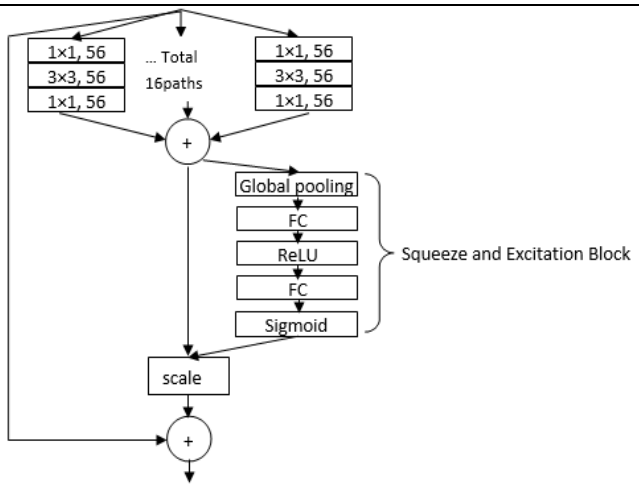


Fig. 2 The building block of SE-ResNext

3.2. Global Information Embedding: Squeeze Operation

The deep neural network faces an exploitation problem of inter-channel dependencies in the later layers, mainly where the receptive field size is comparatively smaller. It cannot share the information or the computational output produced by applying a filter to a local receptive field with other connective channels. We calculate a summary for a channel, which can also be viewed as a statistical representation of a channel to address the barrier of information sharing among channels. Global average pooling is the technique that makes this possible. This operation describes a channel by retrieving the features which carry most of the information. A channel's descriptor $z \in R^c$, with spatial dimensions H × W can be calculated by Equation (2) through the statistic of a channel z of the c-th element can be calculated:

$$z_c = F_{sq}(u_c) = \frac{1}{H \times W} \sum_{i=1}^{H} \sum_{j=1}^{W} u_c(i,j) \qquad (2)$$

The convolutional operation between the local receptive field of the input image X and the 2D filter produces the transformation output u_c (image intensity). Here (i, j) denotes the image's coordinate, 'i' could be any value from 0 to *H*-1, and 'j' could be any value from 0 to *W*-1. The whole image information can be retrieved from the transformation output U, which can be seen as the collection of the major information container of the image. This type of technique is widely used in feature engineering.

3.3. Mutation Rectification: Excitation Operation

The main objective of the operation in the excitation layer is to capture the inter-channel dependencies fully. To achieve this goal, we can use the channel statistics achieved from the previous layer squeeze operation. Two essential criteria must be fulfilled by a very simple gated layer to get the desired result: (i) the channel-to-channel nonlinearity may be capturable, and (ii) as multiple channels are emphasized as opposed to one

hot activation, it must be able to find the channel wise non-mutually-exclusive relationship. In Equation (3), this gated function is formulated as

$$s = F_{ex}(z, W) = \sigma\big(g(z, W)\big) = \sigma\big(W_2\delta(W_1 z)\big) \quad (3)$$

The activation function ReLU [27] is denoted by δ, used in the excitation operation of the above equation. A dimensionality-reduction layer is used to improve the model's generalization capability and decrease the complexity. Two fully connected layers parameterized with a gating function around the nonlinearity layer are used. Here a reduction ratio r =16 is applied with the parameter W_1, a ReLU activation function is used for nonlinearity, the parameter W_2 is used for dimensionality increase. Finally, a rescaling operation is performed to get the final output.

$$\tilde{x}_c = F_{scale}(u_c, s_c) = s_c u_c \quad (4)$$

where,

$\tilde{X} = [\tilde{x}_1 \tilde{x}_2, \ldots, \tilde{x}_C]$ and $F_{scale}(u_c, s_c)$ indicate the channel-wise influence of the feature map $u_c \in R^{H \times W}$ and the scalar s_c.

3.4. Applying Regularization: Dropout

Dropout is a regularization technique; the term denotes dropout randomly selects neurons from both hidden and visible layers in a neural network. This is one of the early techniques to overcome the overfitting problem. In each iteration, as some neurons were dropped out in forwarding pass, then in the backpropagation, these neurons do not exist. For the input layer, 0.1 is the recommended dropout rate, and for internal layers, this value could be anything between 0.5 to 0.8. Despite some advantages of using dropout, it is needed to be cautious using dropout in some scenarios like a) regularization is unnecessary when the model is small relative to the volume of the training dataset; B) when we have a limited training time; c) should not use it to the layer before the final classification layer (i.e., last layer), as the model cannot "correct" errors induced by dropout before the classification happens. We have successfully used the dropout technique to address the overfitting problem and make our model more general. Fig. 3 shows a sample network where dropout is applied.

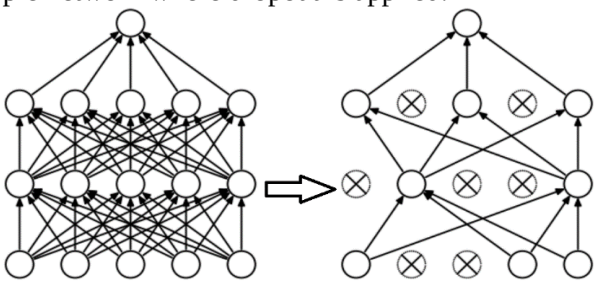


Fig. 3 Transformation of neural network where dropout [28] in place

3.5. Applying Regularization: Dropout

Stacking layers of SE-ResNext Blocks constructs the model. A slight modification has been performed on the original ResNext model. Here, a simple version of the squeeze and excitation computational layer is added with the existing architecture of ResNext.

Table 7 Filter and parameter settings - construction of the SE-ResNeXt model

Layer	Kernel Size	Filter	Activation	Cardinality	Block	Output
Convolution	3×3	64	RELU	-	-	32×32×64
Convolution	1×1	64	RELU	16		
Convolution	3×3	64	RELU			
Convolution	1×1	64	-		3	32×32×64
Global avg. pooling	-	-	-	-		
Convolution	-	-	RELU			
Convolution	-	-	Sigmoid			
Convolution	1×1	64	RELU	16		
Convolution	3×3	64	RELU			
Convolution	1×1	64	-		3	16×16×128
Global avg. pooling	-	-	-	-		
Fully-Connected	-	-	RELU			
Fully-Connected	-	-	Sigmoid			
Convolution	1×1	64	RELU	16		
Convolution	3×3	64	RELU			
Convolution	1×1	64	-		3	8×8×256
Global avg. pooling	-	-	-	-		
Fully-Connected	-	-	RELU			
Fully-Connected	-	-	Sigmoid			
Global avg. pooling	-	-	-	-	-	256
Fully-Connected	-	-	-	-	-	84

Three consecutive convolutional layers followed by a ReLU nonlinearity for the first two Conv layers, global average pooling layer, and two fully connected layers followed by nonlinearity make the SE block. Table 7 shows the detailed architecture of the model. The cardinality column shows the splits of the ResNext blocks, and the block column shows the number of SE blocks to construct a SE layer. The model takes the input image of 32×32 and outputs a 1×84 single dimensional array of probabilities.

4. Experiment and Result Analysis

The standard Mendeley BanglaLekha-Isolated 2 dataset [28] was used to train and validate our model. The dataset is well-curated, taken samples from people of different ages, groups, and sex, containing handwritten Bangla numerals, basic and compound characters.

Table 8 Dataset comparative study - MNIST vs. MENDELEY BanglaLekha-Isolated 2

Attributes	MNIST	Mendeley BanglaLekha-Isolated 2
Position	Centered	Non-Centered
Formation	Uniform	Non-Uniform

Classes	10 (only ten digits)	84 (50 basic characters, 24 compound characters, ten digits)
Total samples	70000	166105

The dataset contains 10 Bangla numerals, 50 basic characters, 24 different frequently used compound characters. For a single character, around 2000 different handwritten samples were collected and performed pre-processing tasks. The total number of samples is 166106 for 84 different classes in place of 168000, 2000 each because some erroneous samples are discarded. We have taken 165000 samples from this dataset for the training and testing, 132000 for training and 33000 for testing. Mendeley BanglaLekha-Isolated 2 is comparatively complex than the MNIST dataset [29] from scaling, location, and the number of classes' points of view. The differences between these two datasets are shown in Table 8.

The standard equation of Accuracy, Precession, Recall, and F1-score are shown below. We can get the accuracy by dividing correctly predicted samples by total samples, get the precision by dividing true positively predicted samples by total predicted positive samples. The recall can be obtained by dividing correctly positive predicted samples by all positive samples. F1-score considers both precision and recall, the weighted average of these two values. The performance results shown in Table 3 are really good performances.

$$Accuracy\ (\%) = \frac{TP + TN}{TP + TN + FP + FN} \times 100 \quad (5)$$

$$Precision\ (\%) = \frac{TP}{TP + FP} \times 100 \quad (6)$$

$$Recall\ (\%) = \frac{TP}{TP + FN} \times 100 \quad (7)$$

$$F1 - score\ (\%) = 2 \times \frac{Precision \times Recall}{Precision + Recal} \times 100 \quad (8)$$

Here, TP denotes the correctly predicted correct value. TN denotes a correctly predicted wrong value, FP denotes a predicted wrong value as a correct one. FN denotes the predicted correct value as wrong. These measurements can be calculated by Eq. (9) to Eq. (12) [29].

$$TP_i = c_{ii} \quad (9)$$

$$TN_i = \sum_{k=1,k\neq i}^{n} \sum_{j=1,j\neq i}^{n} c_{jk} \quad (10)$$

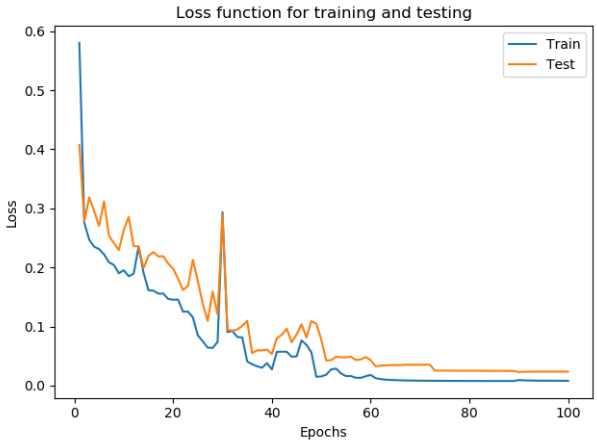
$$FP_i = \sum_{j=1,j\neq i}^{n} c_{ji} \quad (11)$$

$$FN_i = \sum_{j=1,j\neq i}^{n} c_{ij} \quad (12)$$

Here '*i*' denotes the current category, and *n* denotes character categories ($n = 84$).

For training and validation of our model, a computer with 9th generation processor Intel 9700K core-i7, 3.6 GHz with 16 GB RAM is used. An NVIDIA GPU RTX-2080Ti with 11GB memory of GDDR6 was used to get the CUDA accelerated parallel computing.

We need to pre-process the dataset images before feeding them to our model. All the image sizes are between 155×155 to 185×185. These images needed to be converted to grayscale with a size of 32×32 pixels. We choose 64 as the training batch size and 10 as an iteration for the testing phase to get the average result. Nesterov Momentum was employed as the optimization function. We performed batch normalization [30] after a convolutional operation and before feeding the result to the next computational layer, and it helps greatly to quick convergence and faster training process. The demonstrated result in the training and testing phase indicates that the model overcomes the overfitting [27] problem and performs well.



(a)

(b)

Fig. 4 A comparative study on model performance in training and testing phase

To overcome the issue of overfitting, we used a generalization technique called dropout, which makes the model simpler and more generalized. Fig. 4(a) clarifies that the training and testing lines are very close to each other and almost approach zero. It

expresses two things: first, the model is generalized well, training loss is not too low than the validation loss (no overfitting), training loss is not too high (no underfitting); second, as both of the lines close to zero indicates a well-trained stable model. Fig. 4(b) shows that both the training and testing accuracy become stable after some iteration, indicating the model is doing well for unseen data. Moreover, both lines are almost close to 1, indicating the model is properly trained. We used another technique called early stopping, which refers to a specific point of training iteration where the training accuracy reaches its peak. After that point, the model would not learn much but loosen its ability to generalize and overfit the training data. Therefore, the training process should be stopped for early stopping before passing that point.

Compared to the traditional deep learning model, which requires many hyperparameters, our proposed model requires to set the cardinality as the only one hyperparameter. We have used RELU and skip connections to address the issue of vanishing gradient. RELU helps efficiently gradient flow where skip connection makes the gradient flow without any loss. Before the final fully connected layer, we have used a more native CNN environment layer with the global average pooling, which is very efficient in finding the similarities between the output classes and the input feature maps.

Our experiment has demonstrated that the true positive rate is 100% for the majority of the classes, except for two classes "৫" and "৬", where the true positive rate is 99% for both classes, and the false positive rate is 1%, because of few samples where these two characters look almost similar. When looking closely at these examples, we can see that ৫ if the upper portion arc is open, the character "" becomes identical to the character "৬".

Some basic characters whose formation and pattern are almost similar to some compound characters and numerals. It is harder to distinguish them for obtaining a good result in these cases. For example, sample image data of classes "ঠ", "ভ" and "ঙ" looks like images of numeric classes "৮", "৫" and compound class "ল্ড" consecutively. In some compound characters, the model faces this problem also, easily noticeable characters of similar patterns are স্ফ, ক্ষ, ষ্প, ষ্প, ক্ষ্ম, ক্ষ. Our model also finds it difficult to distinguish between the character "স্ফ" which formation is "স+ফ", and the character "ষ্প", which formation is "ষ+প", but the turns, angels, and the writing pattern make these two characters almost similar to the human eye. The same conclusion can be drawn for the character "স্ফ", as it is similar to the character "ক্ষ" and character "ক্ষ্ম" as it is similar to the character "ক্ষ".

We have seen these similarities in some basic characters. If we closely look into the character pairs ("খ", "ঘ"), ("ভ","ড"), ("ঙ", "ল্ড") and ("র","ব"), we can find the similarity problem for the first pair, "খ" and "ঘ", even though the computer typed character looks different. However, the handwritten format of these two looks similar. Samples from some individuals show that if they do not put the 'matra' (the top horizontal line), "ঘ" looks like "খ". For characters "ঙ" and "ল্ড", if someone puts a 'matra' on top of "ঙ", it looks the same as "ল্ড", which is a compound character. The same can be said for the characters "ভ" and "ড". If we look at the characters "র" and "ব", where the dot at the character bottom "র" is too tiny to identify, making it similar to the character "ব".

Table 9 Class-wise performance matrix

Class	Accuracy	Precision	Recall	F1-score
০	1.00	1.00	1	1.00
১	0.999	1.00	0.990099	1.00
২	1.00	1.00	1	1.00
৩	1.00	1.00	1	1.00
৪	1.00	1.00	1	1.00
৫	0.998	0.99	0.99	0.99
৬	0.998	0.99	0.99	0.99
৭	1.00	1.00	1	1.00
৮	1.00	1.00	1	1.00
৯	0.999	0.99	1	0.99
অ	1.00	1.00	1.00	1.00
আ	1.00	1.00	1.00	1.00
ই	1.00	1.00	1.00	1.00
ঈ	1.00	1.00	1.00	1.00
উ	1.00	0.99	0.87	0.93
ঊ	1.00	0.98	0.99	0.98
ঋ	0.99	0.88	0.71	0.79
এ	1.00	1.00	1.00	1.00
ঐ	1.00	1.00	1.00	1.00
ও	1.00	1.00	1.00	1.00
ঔ	1.00	1.00	1.00	1.00
ক	1.00	1.00	1.00	1.00
খ	1.00	0.95	1.00	0.98
গ	1.00	1.00	1.00	1.00
ঘ	1.00	0.95	1.00	0.98
ঙ	1.00	1.00	1.00	1.00
চ	0.99	0.89	0.71	0.79
ছ	1.00	1.00	1.00	1.00
জ	1.00	1.00	1.00	1.00
ঝ	1.00	0.82	0.77	0.79
ঞ	1.00	1.00	1.00	1.00
ট	1.00	1.00	1.00	1.00
ঠ	1.00	1.00	1.00	1.00
ড	1.00	1.00	1.00	1.00
ঢ	1.00	1.00	1.00	1.00
ণ	1.00	0.98	0.98	0.98
ত	1.00	1.00	1.00	1.00
থ	0.99	0.91	0.70	0.79
দ	1.00	1.00	1.00	1.00
ধ	1.00	0.91	0.90	0.90
ন	0.99	0.95	0.98	0.97
প	1.00	1.00	1.00	1.00
ফ	1.00	1.00	1.00	1.00
ব	1.00	1.00	1.00	1.00

ভ	1.00	1.00	1.00	1.00
ম	0.99	0.91	0.70	0.79
য	0.98	0.95	0.93	0.94
র	0.99	0.98	0.77	0.86
ল	1.00	1.00	1.00	1.00
শ	1.00	1.00	1.00	1.00
ষ	1.00	1.00	1.00	1.00
স	1.00	1.00	1.00	1.00
হ	1.00	1.00	1.00	1.00
ড়	1.00	1.00	1.00	1.00
Continuation of Table 9				
ঢ়	1.00	1.00	1.00	1.00
য়	0.98	0.92	0.98	0.95
ৎ	1.00	1.00	1.00	1.00
ং	1.00	1.00	1.00	1.00
ঃ	1.00	1.00	1.00	1.00
ঁ	1.00	1.00	1.00	1.00
ক্ষ	0.99	0.84	0.95	0.89
ঝ	1.00	1.00	1.00	1.00
স্ফ	1.00	0.99	0.91	0.95
স্থ	1.00	1.00	1.00	1.00
চ্ছ	1.00	0.99	0.99	0.99
ক্ত	1.00	1.00	1.00	1.00
ম্ন	1.00	0.99	1.00	1.00
ঞ্চ	1.00	0.99	1.00	1.00
ল্প	0.99	0.96	0.85	0.90
প্ত	1.00	1.00	1.00	1.00
ষ্ব	0.997	1.00	0.99	0.99
ণ্ড	1.00	1.00	1.00	1.00
ড্ড	1.00	1.00	1.00	1.00
হ্ম	1.00	1.00	0.99	0.99
র্ত	1.00	0.98	1.00	0.99
ল্ল	1.00	1.00	1.00	1.00
ল্প	0.99	0.88	0.98	0.92
ন্দ	1.00	1.00	1.00	1.00
ন্ধ	1.00	1.00	0.99	0.99
ষ্ম	0.998	0.99	0.98	0.99
র্ক	1.00	1.00	0.98	0.99
ঙ্গ	1.00	1.00	1.00	1.00
ক্ষ	1.00	0.91	0.98	0.94
ক্ষ্ম	0.99	0.93	0.84	0.88
Average	99.82%	98.08%	96.90%	97.40%

From our experimental results, we have drawn an 84×84 normalized confusion matrix. Using this confusion matrix and the formula from Equations (5) to (9), we have calculated the accuracy, precession, and other measurements of our proposed model. The testing phase revealed that the accuracy level for most classes reached almost 100, except for some characters, a few of which we have mentioned earlier. To recognize a character, the average time taken by our model is 12ms, which denotes that the model can be applied to any real-world scenario.

Table 10 Comparative study of our proposed method with similar recent state-of-the-art methods

Proposed By	**Used Methodology**	**Accuracy**
Kibria *et al.* [1]	Support Vector Machine (SVM)	88.73%
Pramanik *et al.* [2]	Shape decomposition + MLP	88.74%
Ashiquzzaman *et al.* [23]	Deep CNN	93.68%
Fardous *et al.* [22]	DCNN with ReLU and Dropout	95.50%
Alif *et al.* [10]	ResNet-18	95.99%
Chatterjee *et al.* [24]	ResNet-50	96.12%
Saha and Saha. [25]	DCNN + Divide and Merge Mapping + Optimal Path Finder	97.12%
Alom *et al.* [4]	ResNet + DenseNet	98.31%
Hasan *et al.* [7]	DCNN + BiLSTM	98.50%
Our Method	SE-ResNeXt	99.82%

Table 10 clearly shows that the accuracy of our model is outperforming other similar models from recent times. The computational complexity and the architecture of our model are also much simpler than the rest of similar models. The computational power and the memory requirement to train the model are minimal for our model.

Table 10 shows a list of different methods used for Bangla character recognition. It is seen that the performance dramatically improved with CNN or deep CNN-based models. The computational power has indeed increased with time, but it mainly accelerates the training process and allows models to train on more data.

From the table, we can see that the performance of the SVM-based [1] model is poor. SVM relies on spectral information only; it cannot use rich spatial information. Moreover, if the data size is large and formation is complex, SVM failed to recognize the underlying correlation. The MLP performance (multilayer perceptron) [2] and shape decomposition-based model are also poor. In MLP, every neuron of a layer is connected with every other neuron of another layer, ending up with too many weights, becomes unmanageable, overfitting is a common problem, so the models become less generalized. The shape decomposition model requires hand engineering for feature extraction, which is an erroneous and time-inefficient technique, requires much expertise.

The deep CNN model [22, 23] performs comparatively better than SVM or MLP based model. Nevertheless, the accuracy is not up to the mark for real-life usages. The model proposed in [23] is not properly generalized and fine-tuned. Compared to this, the model proposed in [22] performs better but is not an industry standard.

The performance of both ResNet [10] and DenseNet [4] based models is much better than the models discussed above. Both of these networks require too much memory. Even DenseNet requires a bit more than ResNet as it requires performing concatenation operation on tensors from different layers, but its accuracy is much better. If we look into two ResNet models [10] and [30], going deep from 18 to 50 layers does not ensure a significant improvement in accuracy; rather, it makes the model complex and increases the training time.

DCNN + LSTM model [7] performs well. As deep CNN goes deeper by adding layer after layer, it means

more weights and biases needed to be trained; it requires many training data to get a good result, the training process is extremely expensive. For LSTM, overfitting is a very common problem as the dropout technique is a bit difficult to apply in LSTM. LSTM based model has many parameters that require many memories to train. Most importantly, LSTM is good for time series or sequential data, but the image is not sequential.

In our proposed SE-Resnext model, rather than going deep by adding convolutional blocks one after another, we added SE blocks, making the model wider with fewer parameters. It is very efficient in learning complex features by fusing channel-wise feature dependencies. Only one hyperparameter called split or cardinality is necessary. This hyperparameter denotes how many parallel paths are required to operate on the input image or the output of the previous block. Since it requires only one hyperparameter to be tuned up, the model is easily manageable. Another noticeable change is using skip connections, which allows the gradient to flow naturally without any loss, allows maximum information flow, and helps the earlier neuron train faster. It also addresses the vanishing gradient problem along with the ReLU nonlinearity.

Despite the outperforming result, we have noticed some misclassification in the training phase. We have analyzed and recorded these misclassification samples. Most of these samples were very tough to recognize, even for the human eye, due to the similarities, writing patterns, or erroneous data. The fewer data, lack of pattern variations, and formations could have resulted in this misclassification. Modification or further improvements of models, training on more versatile data could bring further good results. In Fig. 13, a few misclassification samples are listed.

Table 11 Some misclassification cases

Test Data	True Class	Predicted Class
	ঠ	৮
	ভ	৫
	ক্ষ্ম	ক্ষ
	ষ্প	স্প

According to Table 11, the test character from the first row is predicted as "৮" which is a numeral character, but its true class is "ঠ", which is a basic character. If we look at some samples from our dataset, "  ", "  " and "  " of class "ঠ", then it becomes clear that these samples are almost identical with the writing samples "  ", "  " and "  " of class "৮". The same can be seen for the rest of the test character data of the table.

We have compared the misclassification rate and the level of our model with other related models. Some of these misclassified results by other methods are shown in Table 12.

Table 12 Some misclassification cases for other related methods

Method	Test Data	True Class	Predicted Class
Rabby *et al.* [3]		খ	ঘ
		ঙ	ভ
		য়	র
Alif *et al.* [10]		ও	ল
		ভ	ড
		ড	ভ
Pramanik *et al.* [2]		গ্ন	গ্ন
		ঙ্ক	ঙ্ক
		হ্ন	হ্ন
Ashiquzzaman *et al.* [23]		স্থ	চ্ছ
		প্ন	প্র
		প্র	ত্র
Hasan *et al.* [7]		প্র	ম্র
		ণ্ট	ন্ট
		ল্ড	ল্ত

If we look at the misclassification result from Table 12, the model proposed in [3] shows poor performance. The sample image data shows that images clearly identify their true class easily. For the first two rows of sample data, the model fails to recognize a very simple but essential feature, the 'matra' (the top horizontal line). If we closely look into these two sample characters, we can see the first one has half matra and the second one has no 'matra', but the model's predicted class for these two-sample data shows full matra. The same scenario happened for the model proposed *in* [10]. The models proposed in [2] and [23] perform better, and these two models failed to recognize complex images due to formation and cursive-ness. The performance of the model proposed in [7] is even better than all the above. Here, the sample images are complex in terms of formation, curviness, and erroneous. However, the last and middle sample images are easily identifiable, which the method failed to identify.

5. Discussion

5.1. Main Findings of the Present Study

A single model can perform equally well in all different types of characters, which helps us to reduce

the dependencies of using separate models for each type of character. Rather than going deep in layers and complex architecture, we can stack a simple mathematical block, SE-Block, which is very efficient in terms of complex feature identification and computational cost. Quick convergence, fewer chances of overfitting, and simplicity make our model a good choice to use in modern applications.

5.2. Comparison with Other Studies

Our study shows that the SVM or MLP based models are not ideal for large volume and complex datasets. Overfitting is a common problem for these types of models. Memory consumption is also high for MLP base models. Shape decomposition models require hand engineering for feature extraction, which is a time-consuming and complex task. The DCNN, Resnet, or Dense models require too much memory, Model's architecture is very complex. These models require longer training times. Overfitting is very common in LSTM based models.

5.3. Implication and Explanation of Findings

Identifying the complex feature from inter-channel feature dependencies helps our model perform better. The mathematical operation behind the SE block is very simple, enabling the model to consume less computational power. Besides, the models do not go deep, so they do not require training many parameters, fewer chances of overfitting, and minimal training time.

5.4. Strengths and Limitations

Quick convergence, high accuracy in all three different types of characters, and the requirement of fewer training data sets are major strengths of our proposed model. Architectural simplicity and less computation cost also make our model suitable for industry standards. There are few symbols and special characters in Bangla language scripts, and those are not in scope to our model. Also, some similar shaped characters' identification could be better. Our model also trained only a single dataset. A trained model volume could have smaller to use on a smart or IoT device.

6. Conclusion

In our paper, we proposed a deep convolutional neural network-based model named SE-ResNeXt to recognize the full set of handwritten Bangla basic, numerals, and compound characters. SE layer is very simple to design and with almost no computational cost. It empowers the model to quick convergence while training and learning channel-wise feature inter-dependencies that are effective in learning complex features. A standard benchmark dataset is utilized to validate the performance of the proposed model. The experimental result demonstrates an average accuracy of 99.82%, a precision of 97.75%, a recall of 97.63%, and an F1-score of 97.62% using the Mendeley BanglaLekha-Isolated 2 dataset. In addition, the proposed model shows comparatively better results than the existing model with higher accuracy. We will apply the model to other datasets for further improvement in the future.

References

[1] KIBRIA M R, AHMED A, FIRDAWSI Z, et al. Bangla Compound Character Recognition using Support Vector Machine (SVM) on Advanced Feature Sets [C]. *IEEE Region 10 Symposium (TENSYMP), Dhaka, Bangladesh, 2020.*

[2] PRAMANIK R, & BAG S. Shape decomposition-based handwritten compound character recognition for Bangla OCR [J]. *Journal of Visual Communication and Image Representation*, 2018 (50):123-134.

[3] RABBY A S A, HAQUE S, ISLAM M S, et al. BornoNet: Bangla Handwritten Characters Recognition Using Convolutional Neural Network [C]. *Proceedings of the International Conference on Advances in Computing and Communication (ICACC-2018), Kochi, India*, 2018(9).

[4] ALOM M Z, SIDIKE P, HASAN M, et al. Handwritten Bangla Character Recognition Using the State-of-the-Art Deep Convolutional Neural Networks [J]. *Computational Intelligence and Neuroscience, Hindawi*. Article ID 6747098, 2018, (2018): 13.

[5] NASIB A U, KABIR H, AHMED R, et al. A Real Time Speech to Text Conversion Technique for Bengali Language [C]. *2018 Proceedings of the International Conference on Computer, Communication, Chemical, Material and Electronic Engineering (IC4ME2), Rajshahi, Bangladesh*, 2018.

[6] CHOWDHURY A R, BISWAS A, HASAN S M F, et al. Bengali Sign language to text conversion using artificial neural network and support vector machine [C]. *2017 3rd Proceedings of the International Conference on Electrical Information and Communication Technology (EICT), Khulna, Bangladesh*, 2017.

[7] HASAN M J, WAHID M F, & ALOM M S. Bangla Compound Character Recognition by Combining Deep Convolutional Neural Network with Bidirectional Long Short-Term Memory [C]. *Proceedings of the International Conference on Electrical Information and Communication Technology (EICT), Khulna, Bangladesh*, 2019.

[8] REZA S, AMIN O B, & HASHEM M M A. Basic to Compound: A Novel Transfer Learning Approach for Bengali Handwritten Character Recognition [C]. *Proceedings of the International Conference on Bangla Speech and Language Processing (ICBSLP), Sylhet, Bangladesh*, 2019.

[9] KESERWANI P, ALI T, & ROY P P. A two phase trained Convolutional Neural Network for Handwritten Bangla Compound Character Recognition [C]. *Proceedings of the 9th International Conference on Advances in Pattern Recognition (ICAPR), Bangalore, India*, 2017.

[10] ALIF M A R, AHMED S, & HASAN M A. Isolated Bangla Handwritten Character Recognition with Convolutional Neural Network [C]. *Proceedings of the*

International Conference of Computer and Information Technology (ICCIT), Dhaka, Bangladesh, 2017.
[11] KHAN M M, UDDIN M S, PARVEZ M Z, et al. A squeeze and excitation ResNeXt-based deep learning model for Bangla handwritten compound character recognition [J]. *Journal of King Saud University – Computer and Information Sciences*, 2021: 1-9.
[12] LI ZH, WU Q, XIAO Y, et al. Deep Matching Network for Handwritten Chinese Character Recognition [J]. *Pattern Recognition*, 2020 (107): 107471, [EB/OL]. https://doi.org/10.1016/j.patcog.2020.107471.
[13] GAN J, WANG W, & LU K. Compressing the CNN architecture for in-air handwritten Chinese character recognition [J]. *Pattern Recognition Letters*, 2020 (129): 190-197, [EB/OL]. https://doi.org/10.1016/j.patrec.2019.11.028
[14] ELTAY M, ZIDOURI A, & AHMAD I. Exploring Deep Learning Approaches to Recognize Handwritten Arabic Texts [J]. *IEEE Access*, 2020 (8): 89882 - 89898, [EB/OL]. https://ieeexplore.ieee.org/document/9091836
[15] BHAGYASREE P V, JAMES A, & SARAVANAN C. A Proposed Framework for Recognition of Handwritten Cursive English Characters using DAG-CNN [C]. *Proceedings of the International Conference on Innovations in Information and Communication Technology (ICIICT), CHENNAI, India*, 2019: 1-6.
[16] SAUFI M M, ZAMANHURI M A, MOHAMMAD N, et al. Deep Learning for Roman Handwritten Character Recognition [J]. *Indonesian Journal of Electrical Engineering and Computer Science*, 2018, 12(2): 455-460.
[17] CLANUWAT T, LAMB A, & KITAMOTO A. KuroNet: Pre-Modern Japanese Kuzushiji Character Recognition with Deep Learning [C]. *Proceedings of the International Conference on Document Analysis and Recognition (ICDAR2019), Sydney, Australia*, 2019: 607-614.
[18] JANGID M, & SRIVASTAVA S. Handwritten Devanagari Character Recognition Using Layer-Wise Training of Deep Convolutional Neural Networks and Adaptive Gradient Methods [J]. *Journal of Imaging, MDPI*, 2018, 4(2): 41.
[19] HUSNAIN M, MISSEN M M S, Mumtaz S, et al. Recognition of Urdu Handwritten Characters Using Convolutional Neural Network [J]. *Applied Sciences, MDPI*, 2019, 9, (13): 2758.
[20] AMIN M S, YASIR S M, & AHN H. Recognition of Pashto Handwritten Characters Based on Deep Learning [J]. *Sensors*, MDPI, 2020, 20 (20): 5884.
[21] PAREEKM J, SINGHANIA D, KUMARI R R, et al. Gujarati Handwritten Character Recognition from Text Images [J]. *Procedia Computer Science*, 2019, 171: 514-523.
[22] FARDOUS A, AFROGE S. Handwritten Isolated Bangla Compound Character Recognition [C]. *Proceedings of the International Conference on Electrical, Computer and Communication Engineering (ECCE), Cox's Bazar, Bangladesh*, 2019.
[23] ASHIQUZZAMAN A K M, TUSHAR A K, DUTTA S, et al. An Efficient Method for Improving Classification Accuracy of Handwritten Bangla Compound Characters using DCNN with Dropout and ELU [C]. *Proceedings of the International Conference on Research in Computational Intelligence and Communications Networks (ICRCICN), Dhaka, Bangladesh*, 2017.
[24] CHATTERJEE S, DUTTA R K, GANGULY D, et al. Bengali Handwritten Character Classification using Transfer Learning on Deep Convolutional Neural Network [C]. *Proceedings of the International Conference on Intelligent Human Computer Interaction, Allahabad, India*, 2019.
[25] SAHA S, & SAHA N. A Lightning fast approach to classify Bangla Handwritten Characters and Numerals using newly structured Deep Neural Network [C]. *Proceedings of the International Conference on Computational Intelligence and Data Science (ICCIDS 2018), Gurugram, India*, 2018.
[26] HU J, SHEN L, ALBANIE S, et al. Squeeze-and-Excitation Networks [J]. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020, 42(8): 2011-2023.
[27] CLEVERT D A, UNTERTHINER T, HOCHREITER S. Fast and accurate deep network learning by exponential linear units (ELUs) [c]. *2016 Proceedings of the International Conference on Learning Representations. May 2, 2016 - May 4, 2016, San Juan, Puerto Rico [EB/OL]*. arXiv:1511.07289, 2016.
[28] MOHAMMED N, MOMEN S, ABEDIN A, et al. BanglaLekha-Isolated [EB/OL]. https://data.mendeley.com/datasets/hf6sf8zrkc/2, 2017.
[29] LECUN Y, CORTES C, & BURGES C J. MNIST handwritten digit database. AT&T Labs. [EB/OL]. http://yann.lecun.com/exdb/mnist, 2010, 2.
[30] THAKKAR V, TEWARY S, & CHAKRABORTY C. Batch Normalization in Convolutional Neural Networks-A comparative study with CIFAR-10 data [C]. *Proceedings of the International Conference on Emerging Applications of Information Technology (EAIT), Kolkata, India*, 2018.

参考文:

[1] KIBRIA M R、AHMED A、FIRDAWSI Z 等。 在高级特征集上使用支持向量机 (支持向量机) 进行孟加拉语复合字符识别 [C]。IEEE 10 区区研讨会（张力），孟加拉国达卡，2020。
[2] PRAMANIK R, 和 BAG S. 基于形状分解的 孟加拉语光学字符识别手写复合字符识别[J]. 视觉传达与图像表示杂志，2018 (50):123-134。
[3] RABBY A S A、HAQUE S、ISLAM M S 等。 博尔诺网：使用卷积神经网络的孟加拉手写字符识别 [C]。计算和通信进展国际会议 (廉政公署-2018)，印度科钦，2018(9)。
[4] ALOM M Z、SIDIKE P、HASAN M 等。使用最先进的深度卷积神经网络的手写孟加拉字符识别[J]。计算智能和神经科学，欣达维。文章 ID 6747098, 2018, (2018): 13.
[5] NASIB A U、KABIR H、AHMED R 等。孟加拉语实时语音到文本转换技术[C]。2018 年计算机、通信、化学、材料和电子工程国际会议 (IC4ME2)，孟加拉拉杰沙希，2018。
[6] CHOWDHURY A R、BISWAS A、HASAN S M F 等人。使用人工神经网络和支持向量机的孟加拉语手语到文本转换 [C]. 2017 年第三届电气信息和通信技术 (信息通信技术) 国际会议，孟加拉国库尔纳，2017。
[7] HASAN M J, WAHID M F, ALOM M S. 结合深度卷积神经网络与双向长短期记忆的孟加拉语复合字符识别[C]

。电气信息和通信技术 (信息通信技术) 国际会议，孟加拉国库尔纳，2019。
[8] REZA S, AMIN O B, 和 HASHEM M M A. 从基础到复合：孟加拉手写字符识别的新型迁移学习方法 [C]。孟加拉语语音和语言处理国际会议 (ICBSLP)，锡尔赫特，孟加拉国，2019。
[9] KESERWANI P, ALI T, 和 ROY P P. 用于手写孟加拉语复合字符识别的两阶段训练卷积神经网络 [C]。第九届模式识别进展国际会议 (ICAPR)，印度班加罗尔，2017.
[10] ALIF M A R，AHMED S，和 HASAN M A. 使用卷积神经网络的孤立孟加拉手写字符识别 [C]。国际计算机和信息技术会议 (国际刑事法院)，达卡，孟加拉国，2017。
[11] KHAN M M、UDDIN M S、PARVEZ M Z 等。一种基于挤压激发水库下一个 的孟加拉语手写复合字符识别深度学习模型 [J]. 沙特国王大学学报 - 计算机与信息科学，2021：1-9。
[12] LI ZH, WU Q, XIAO Y，等。手写汉字识别的深度匹配网络[J]. 模式识别，2020（107）：107471，[EB/OL]。 https://doi.org/10.1016/j.patcog.2020.107471。
[13] GAN J, WANG W, 和 LU K. 压缩美国有线电视新闻网架构用于空中手写汉字识别[J]. 模式识别快报，2020 (129) : 190-197 , [EB/OL] 。 https://doi.org/10.1016/j.patrec.2019.11.028
[14] ELTAY M, ZIDOURI A, 和 AHMAD I. 探索识别手写阿拉伯文本的深度学习方法[J]. IEEE 访问，2020 (8) : 89882 - 89898 , [EB/OL] 。 https://ieeexplore.ieee.org/document/9091836
[15] BHAGYASREE P V, JAMES A, 和 SARAVANAN C. 使用有向无环图-美国有线电视新闻网[C] 识别手写草书英文字符的提议框架。信息和通信技术创新国际会议 (ICIICT)，印度钦奈，2019：1-6。
[16] SAUFI M M、ZAMANHURI M A、MOHAMMAD N 等。罗马手写字符识别的深度学习[J]. 印度尼西亚电气工程与计算机科学杂志, 2018, 12(2): 455-460.
[17] CLANUWAT T, LAMB A, 和 KITAMOTO A. 黑网：基于深度学习的前现代日本 九子寺 字符识别 [C]。文件分析与识别国际会议 (雷达 2019)，澳大利亚悉尼，2019：607-614。
[18] JANGID M, 和 SRIVASTAVA S. 使用深度卷积神经网络和自适应梯度方法的分层训练的手写梵文字符识别[J]。影像学杂志，MDPI，2018，4（2）：41。
[19] HUSNAIN M、MISSEN M M S、MUMTAZ S 等。基于卷积神经网络的乌尔都语手写字符识别[J]. 应用科学, MDPI, 2019, 9, (13): 2758。
[20] AMIN M S, YASIR S M, 和 AHN H. 基于深度学习的普什图手写字符识别[J].传感器，MDPI，2020，20 (20): 5884。
[21] PAREEKM J, SINGHANIA D, KUMRI R R, 等。基于文本图像的古吉拉特语手写字符识别[J]. 普罗西迪亚 计算机科学，2019，171：514-523。
[22] FARDOUS A, AFROGE S. 手写孤立孟加拉语复合字符识别 [C]。电气、计算机和通信工程 (幼儿教育委员会) 国际会议，孟加拉国考克斯巴扎尔，2019。
[23] ASHIQUZZAMAN A K M、TUSHAR A K、DUTTA S 等。一种使用带有辍学和 ELU 的美国有线电视新闻网提高手写孟加拉语复合字符分类精度的有效方法 [C]。计算智能和通信网络研究国际会议 (红十字国际委员会)，孟加拉国达卡，2017。
[24] CHATTERJEE S、DUTTA R K、GANGULY D 等。在深度卷积神经网络上使用迁移学习的孟加拉手写字符分类 [C]。智能人机交互国际会议，印度阿拉哈巴德，2019 年。
[25] SAHA S, 和 SAHA N. 使用新结构的深度神经网络 [C] 对孟加拉手写字符和数字进行分类的快速方法。计算智能和数据科学国际会议 (国际癌症中心 2018)，印度古尔格拉姆，2018。
[26] HU J, SHEN L, ALBANIE S，等。挤压和激发网络 [J]。 模式分析和机器智能汇刊, 2020, 42(8): 2011-2023。
[27] CLEVERT D A, UNTERTHINER T, 和 HOCHREITER S. 通过指数线性单元 (ELU) 进行快速准确的深度网络学习 [C]。 2016 年学习表征国际会议。 2016 年 5 月 2 日 - 2016 年 5 月 4 日，波多黎各圣胡安 [EB/OL]。arXiv：1511.07289，2016。
[28] MOHAMMED N、MOMEN S、ABEDIN A 等。 孟加拉莱卡 - 隔离 [EB/OL] 。 https://data.mendeley.com/datasets/hf6sf8zrkc/2, 2017。
[29] LECUN Y, CORTES C, 和 BURGES C J. MNIST 手写数字数据库。美国电话电报公司实验室。[EB/OL]。http://yann.lecun.com/exdb/mnist，2010，2。
[30] THAKKAR V, TEWARY S, 和 CHAKRABORTY C. 卷积神经网络中的批量归一化——与 CIFAR-10 数据的比较研究 [C]。信息技术新兴应用国际会议 (EAIT)，印度加尔各答，2018。