Egyptian Informatics Journal 24 (2023) 100413

Contents lists available at ScienceDirect

Egyptian Informatics Journal

journal homepage: www.sciencedirect.com

Full Length Article

Self-ChakmaNet: A deep learning framework for indigenous language learning using handwritten characters

Kanchon Kanti Podder [a], Ludmila Emdad Khan [b], Jyoti Chakma [c], Muhammad E.H. Chowdhury [a,*], Proma Dutta [d], Khan Md Anwarus Salam [b], Amith Khandakar [a], Mohamed Arselene Ayari [e], Bikash Kumar Bhawmick [f], S M Arafin Islam [a], Serkan Kiranyaz [a]

[a] Department of Electrical Engineering, Qatar University, Doha, 2713, Qatar
[b] Dream Door Soft Ltd, Dhaka, Bangladesh
[c] Department of Electrical and Electronic Engineering, University of Chittagong, Chittagong-4331, Bangladesh
[d] Department of Electrical & Electronic Engineering, Chittagong University of Engineering & Technology, Chittagong, 4349, Bangladesh
[e] College of Engineering, Qatar University, Doha, 2713, Qatar
[f] Bangladesh Reference Institute for Chemical Measurements (BRiCM), Dhaka, Bangladesh

ARTICLE INFO

Keywords:
Chakma language
Handwritten character recognition
Deep learning
Self-ONN

ABSTRACT

According to UNESCO's Atlas of the World's Languages in Danger, 40% of the languages today are counted as endangered in the future. Indigenous languages are endangered because of the less availability of interactive learning mediums for those languages. Thus this paper proposes an interactive deep learning method for Handwritten Character Recognition of the indigenous language "Chakma." The method comprises dataset creation using a mobile app named "EthnicData." It reports the first "Handwriting Character Dataset" of Chakma containing 47,000 images of 47 characters of Chakma language using the app. A novel SelfONN-based deep learning model, Self-ChakmaNet, is proposed in this research for Chakma Handwritten character recognition. The Self-ChakmaNet achieved 99.84% for overall accuracy, precision, recall, F1 score, and sensitivity. The proposed model with high accuracy can be implemented in mobile devices for handwritten character recognition as the model has less number of parameters and a faster processing speed.

1. Introduction

Using own mother tongue in daily life communication is a representation of freedom. However, freedom is often compromised if that individual or the individual's community resides in a country of people belonging to different linguistic communities. Several linguistic communities coexist inside a country as a result of migration, historical/political land division, and indigenous people that have lived there for millennia [1]. As a result, some communities have switched from their native languages to the dominant language [1]. Around 6500 languages are being spoken all over the world [16]. Within the next 40 years, at least one language will be lost per month [10]. To prevent the extinction of around or more than 1,500 languages by the end of the twenty-first century, urgent investments in language documentation, bilingual education, and other community-based programs are required [10].

Chakma language is spoken in Bangladesh and India. Currently, 320,000 people in southeast Bangladesh in the Chittagong Hill Tracts and another 230,000 in India speak the Chakma language. Chakma is written using the Chakma alphabet, also known as Ajhā pāṭhath, Ojhopath, or Aaojhapath. Fig. 1 illustrates the 8 consonants, 5 vowel and diacritics, and 10 numerals. The Bengali culture and language are having a significant impact on the Chakma population. As a result, people are increasingly converting to Bangla language, endangering the ethnic language. In this digital world indigenous languages need to be digitally usable and recognizable by digital systems. This paper proposes a method of using AI for Handwritten Character Recognition of indigenous language "Chakma". Owing to the unique patterns, strokes,

* Corresponding author.
E-mail addresses: kanchon.k.podder@bmpt.du.ac.bd (K.K. Podder), ludmila@dreamdoorsoft.com (L. Emdad Khan), jyotichakma101@gmail.com (J. Chakma), mchowdhury@qu.edu.qa (M.E.H. Chowdhury), promadutta.11@gmail.com (P. Dutta), anwar@dreamdoorsoft.com (K.M.A. Salam), amitk@qu.edu.qa (A. Khandakar), arslana@qu.edu.qa (M.A. Ayari), bikash@bricm.gov.bd (B.K. Bhawmick), arafinislam87@gmail.com (S.M. Islam), mkiranyaz@qu.edu.qa (S. Kiranyaz).

https://doi.org/10.1016/j.eij.2023.100413
Received 15 November 2022; Received in revised form 30 April 2023; Accepted 23 October 2023
Available online 22 November 2023

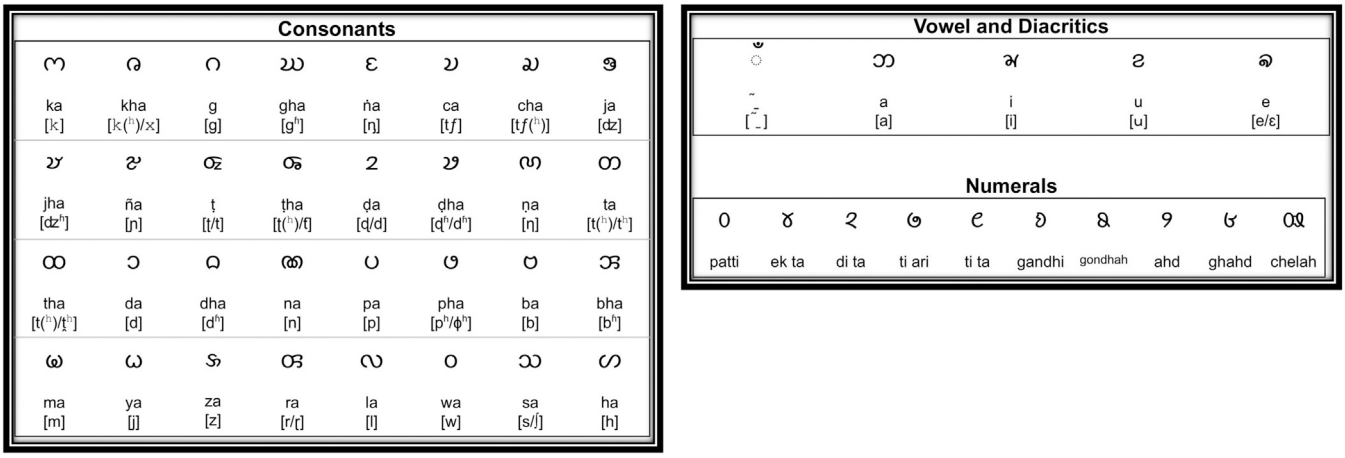


Fig. 1. Chakma language basic character set.

and number of characters in each script, the difficulty of handwritten character recognition varies. We proposed a novel architectural handwritten character recognition module for future use in learning Chakma characters' handwriting using a mobile app.

The paper proposes the use of Self-Organised operational Neural Network (SelfONN) for digitally recognizing the handwriting characters. SelfONN is conceptualized on Generalized Operational Perceptrons (GOPs) which mimics the functions of biological neuron [48]. Self-ONN proposed "generative neurons" to counteract the homogeneous network topology of Multi-Layer Perceptrons (MLPs) and its descendants, Convolutional Neural Networks (CNNs) [22]. Self-ONN models have recently been developed for severe image restoration [24], image denoising [25], image super-resolution applications [19], and image compression [51], surpassing CNN architectures. Given that previous study on the light weight Self-ONN model has shown that it can outmatch a deep CNN counterpart, the potential investigation must be explored for the handwritten character recognition.

The availability of Chakma Handwritten Character dataset is one of the key issues in Chakma Handwritten Character recognition. By addressing the issues of the Chakma Handwritten Character recognition, the contributions of this research are following:

- An image data collecting APP "EthnicData" is presented in order to collect data from various subjects in a simple and quick manner.
- The first "Handwriting Character Dataset" for the indigenous communities (Chakma) in Bangladesh is created using "EthnicData".
- A novel and lightweight SelfONN based handwritten character recognition module is proposed which performed similar to a state-of-the-art architecture MobileNet_V2.

2. Literature review

Mobile Assisted Language Learning (MALL) enables quick access for every learner regardless of location or time restrictions [14]. There are many existing AI based learning app, such as, Duolingo [27], Hello English [11], Babbel [13], Memrise [28], and Busuu [29]. These apps provide learning facilities on some popular languages such as English, French, Spanish, Estonian, German, and Russian. But there is no such app for indigenous languages such as Chakma, Marma, Saotal etc. in Bangladesh. A digital learning platform is needed to preserve this language. AI can be implemented for enhancing the reading, writing, speaking and listening skills.

Deep learning based solutions are gaining attention to solve various problems such as sign language recognition [32,33,31], COVID-19 detection systems [37,47], and Autonomous driving [9]. Automatic handwriting recognition is one of the most popular interest of academic and researchers. There are few handwritten character recognition in language such as Japanese [53], Chinese [50], and Greek [18]. The challenges in automatic handwriting recognition are variety of handwriting styles, different complex writing scripts consist of different form of writing words. These challenges are already mentioned by different research groups in the field of natural language processing [12,26,44]. Handwritten Bangla Character recognition has been done using different pre-trained convolutional neural networks such as VGG Net, ResNET, FractalNet, and DenseNet [2]. In [3], researchers applied deep learning method in 10 Bangla numeric digit recognition and achieved 98.8% accuracy. Another deep learning approach proposed by [34] for 80 handwritten characters. A combination of Multi Layer Perceptron and Adaboost is also used in offline handwritten numeral recognition [17]. In [7], researchers used such combination and delivered a comparison between Devanagari, Bangla and Oriya offline handwritten character recognition.

There is no such research on indigenous language learning by handwriting recognition. The handwriting recognition of indigenous languages in Bangladesh still needs to be explored. This is a big motivation for this research as well.

3. Proposed methodology

In this research, many steps were followed to build the overall system. The steps and components of the system are discussed as below:

3.1. Dataset

The study was approved by the local ethical committee of Qatar University. We have collected data from 50 subjects from the crowd, where the subjects shared their data using the mobile application and there was no identity related information was asked in the acquisition process. Each subject needs to read the consent form and sign the form and upload before proceeding to data recording. Each Chakma Handwritten character was saved as RGB image. This dataset contains 47 Chakma characters, including vowels, consonants, and numbers. All the dataset details are given in Table 1 and Fig. 2 represents the overview of the dataset.

The proposed architecture for Bangla Handwritten Character recognition was subjected to comparative analysis using five additional datasets, namely CMATERdb 3.1.1, BanglaLekha Numerals [8], ISI Numerals [6], CMATERdb 3.1.2, and Ekush [35]. Datasets comprising CMATERdb 3.1.1, BanglaLekha Numerals, and ISI Numerals were gathered to represent 10 distinct Bangla numeral characters. Additionally, CMATERdb 3.1.2 and Ekush datasets were collected to represent 50 distinct Bangla basic characters.

Table 1
Details of Indigenous Handwritten Character Dataset-Chakma.

Dataset information	Details
Image size	3×224×224
Total class	47
Images per class	1,000
Total number of images	47,000
Mean [R,G,B]	[0.9640, 0.9827, 0.9640]
Standard deviation [R,G,B]	[0.1380, 0.0627, 0.1379]

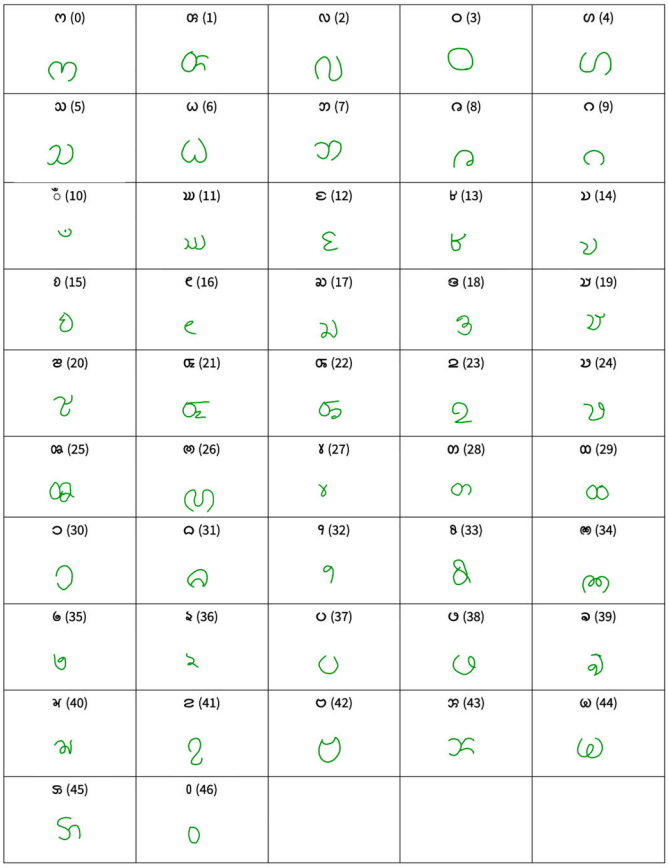


Fig. 2. Example of Indigenous Handwritten Character Dataset - Chakma.

3.2. Basic architecture of system

Language learning can be done by four techniques such as reading, writing, listening and speaking. Writing is a necessary strategy for language learning since it promotes the growth of critical thinking abilities, facilitates collaboration, and enables an individual to focus on and reevaluate his or her ideas later [23]. Currently, in this research we focused on indigenous language learning by achieving writing skill on "Chakma" language. Such system is illustrated in Fig. 3. Four main modules of such system are:

1. **Data Collection Module: *"EthnicData App"*:** Handwritten characters of Chakma language was collected from users using this module.
2. **Data Processing Module:** Collected data is labeled and validated in this module.
3. **"Handwritten Character Recognition" Training Module:** Using data from "EthnicData App", a model was trained and this trained model was used for handwritten character recognition.
4. **Ethnic Handwriting Learning Module: *"Swakkhor App"*:** User learn indigenous handwritten character by deep learning based character recognition process.

3.2.1. Data collection module: "EthnicData App"

This module was used for data collection for indigenous language learning. Fig. 3 block (a) represents the "EthnicData App" module. This module has two elements, including a handwriting canvas and a voice recording. The handwritten "Chakma" language characters were gathered using a handwriting canvas. There is a blank canvas in the Handwriting Canvas where users can draw any letter in accordance with the reference alphabet. When the user has completed sketching, the app saves the vector along with the strokes that have been analyzed and saves it to a Firebase database. Data from the Firebase database is gathered for the data processing module's use. The important parts of the "EthnicData" app are described below:

Canvas This canvas is intended for open and free sketching. Any alphabet may be drawn by the user freehand. In essence, it captures the stroke and makes it visible in a certain color. With the appropriate buttons, the user may manage the drawing further. The user may draw any letter freehand on the completely responsive canvas; but, to save the drawing to the database, they must hit the **"NEXT"** button once they have finished drawing.

Reference alphabet It is a guide that the user may refer to determine which character to draw on the canvas. It serves as a guide to assist the user in determining which alphabet to draw.

Buttons The canvas painting is controlled by the buttons. Five buttons on the handwritten canvas are designated for carrying out particular tasks. These are described below:

1. **"UNDO" Button:** The last drawing line on the canvas is undone using the undo button. If any lines are drawn incorrectly while drawing an alphabet on a canvas, the user can undo the preceding line to correct the error.
2. **"REDO" Button:** This button redo's the last drawing line. When users redo some lines or delete the previous line, user can choose to redo or execute the previous line to get it back.
3. **"CLEAR" Button:** It clears the canvas and make the canvas empty. If user click clear button, it clears the whole canvas to redraw the alphabet.
4. **"REPEAT" Button:** It repeats the current character for every time it is pressed. If there is a need to draw the same character for a specific time, the repeat button can make it happen.
5. **"NEXT" Button:** Basically it does 2 functions. (i) Save: It saves the complete alphabet's vector in Firebase Database, (ii) It clears the canvas and then loads the next alphabet reference to draw.

When a user clicks the "NEXT" button, the whole contents of the canvas are recorded in a vector format, including the coordinates of each stroke, in the firebase database. Each distinct character drawing was saved in a corresponding folder. Though the voice recording component was not used in this research but it collects the corresponding voice recording of alphabets which can be later used for speech recognition in speaking skill development for an ethnic language. Fig. 4 represents the details and layout of "EthnicData" APP.

3.2.2. Data processing module

The purpose of this module is to label, clean, validate the stored dataset. Fig. 3 block B represents the data processing module. Because all data is gathered by crowd sourcing, there is a risk of junk and erroneous data being entered into the database. Validation necessitates retaining valid data while discarding erroneous data. The database was evaluated in order to create a reliable source of dataset for training a Handwriting Training Module.

3.2.3. "Handwritten character recognition" module

This research's goal in establishing written skills on "Indigenous Language Learning" is to assist users in learning "Chakma" alphabets

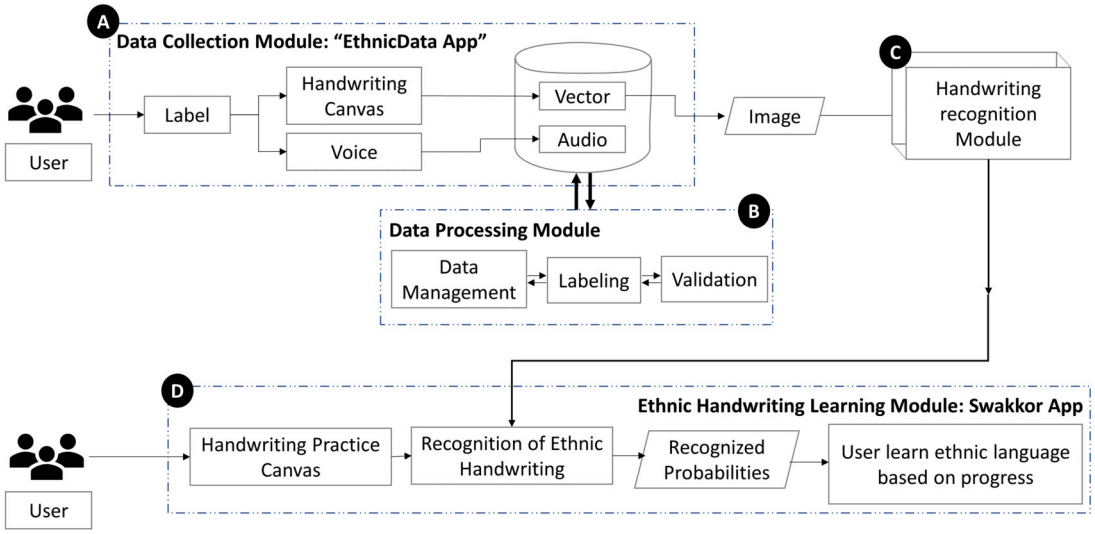


Fig. 3. System overview of "Ethnic Language Learning: Handwriting recognition" system.

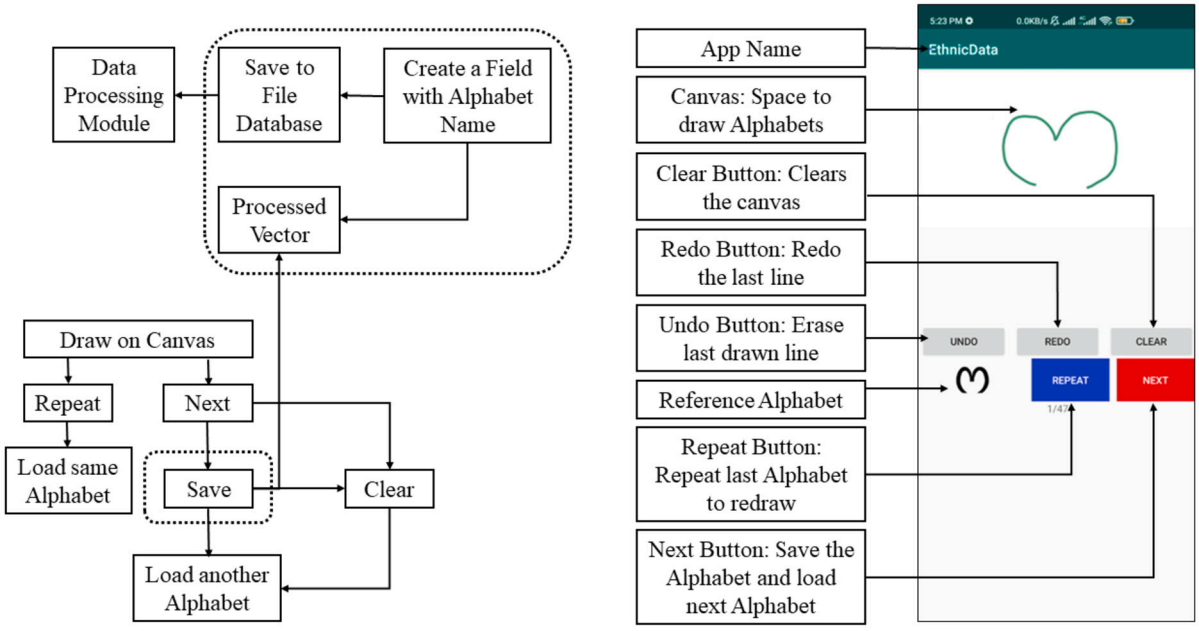


Fig. 4. Layout and details of EthnicData App.

by providing details on how accurate they are writing these alphabets. For that a handwritten character recognition module is needed. The existence of handwritten character recognition module can be found in Fig. 3 block B. Fig. 6 illustrated the flowchart of this module. Data collected from "EthnicData App" database is in image format. To avoid overfitting problem of deep learning models, on-the-fly image augmentation was used, where augmented images were fed into the network at each epoch rather than utilizing the same images in training at each epoch. Image augmentation techniques employed in this study included image resizing, rotation, and perspective. Fig. 5 describes the two out of three on-the-fly augmentation techniques used in this study.

According to Fig. 5, the image was rotated in the *theta* angle range of positive 20 degrees to negative 20 degrees, and the $Bi-Cubic$ interpolation mode was used. Random perspective is another augmentation used in this study, which distorted the image within scale of 0.6 to 1 with creating $\alpha, \beta, \gamma, \psi$ angles distortion around the sides using $Bi-linear$ interpolation. Also, the scaled down image was filled with 0 or q value. Along with the on-the-fly augmentations, all the models were trained from scratch. The selected models are described in Fig. 6.

MobileNet_V2 MobileNet_V2 is a CNN which performs efficiently on mobile devices [39]. This state-of-the-art CNN model is developed on inverted residual block where the residual connection was implemented between the bottleneck layers. The whole architecture contains initial convolutional layer with 32 filters and following that 19 inverted residual bottleneck layers. MobileNet_V2's superiority over other CNN architectures is due to its depth-wise separable convolutions and bottlenecks. The bottlenecks of MobileNet_V2 encode the intermediate inputs and outputs. Furthermore, the model's inner layer improves the model's capacity to convert from a low-level concept to a higher-level descriptor [4,46]. Fig. 7 illustrates building block of MobileNet_V2. Mainly, the features are point-wise convoluted at the beginning and end of the block, while depth-wise separable convolution is done at the middle. Shortcuts allow for faster training and improved accuracy using conventional residual connections.

Self-ChakmaNet In this research, SelfONN based a new architecture is proposed for Chakma Handwritten Character recognition. SelfONN is a variant of Operational Neural Network (ONN) [22,25,19,51,21]. Oper-

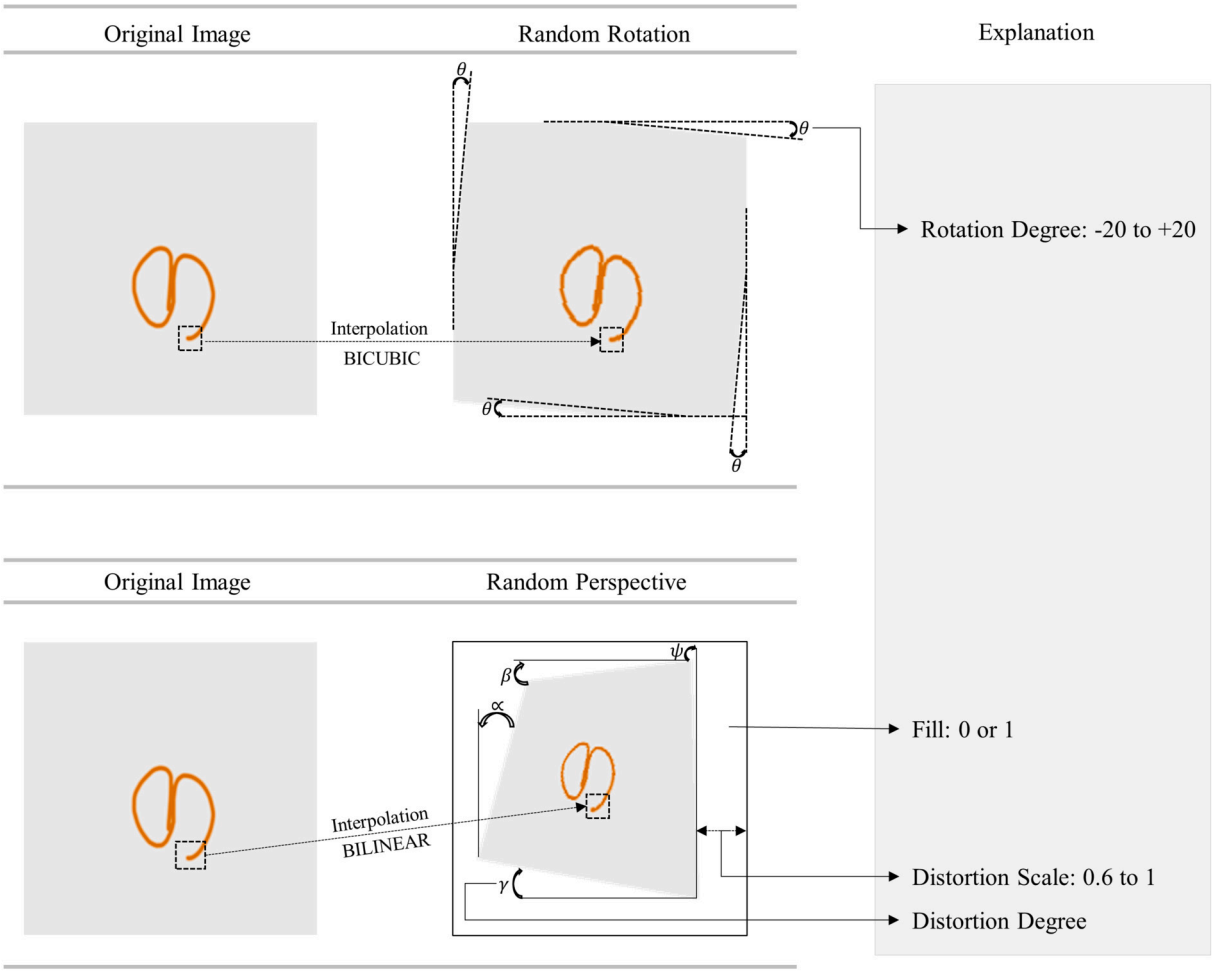


Fig. 5. The Random Rotation and Random Perspective augmentation on a image sample from the dataset.

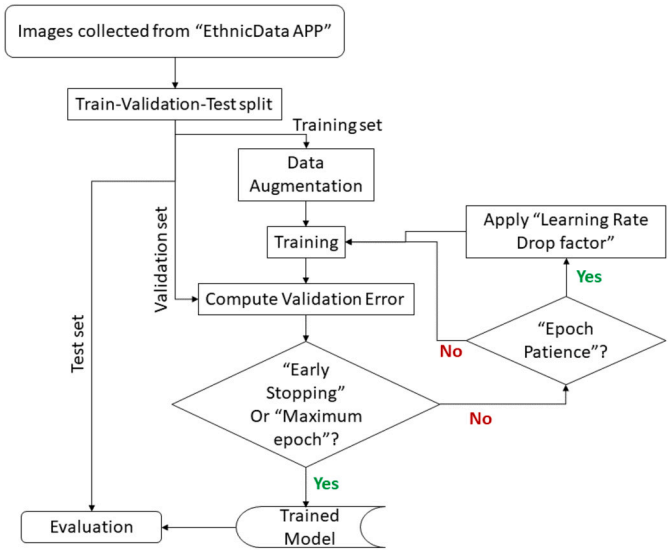


Fig. 6. Flowchart of Handwriting Training Module.

ation Neural networks (ONNs) are conceptualized on Generative Operational Perceptrons (GoPs) [21]. By incorporating generative neurons, ONNS or SelfONN replaces the homogeneous linear approximation of CNN. These ONNs or SelfONNs [21,51,25] imitate the genuine biological neuron with varied synaptic connections since biological neurons carry out a wide range of neurochemical processes. In ONNs or Self-

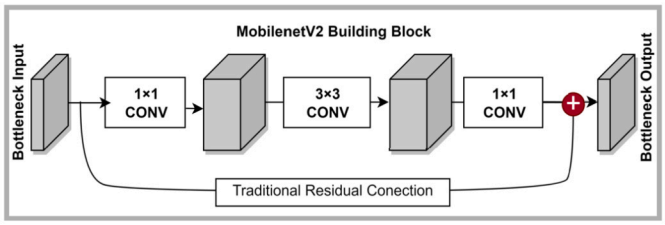


Fig. 7. Basic building block of MobileNet_V2 CNN architecture.

ONNs, the non-linear synaptic connections as well as the integration process that takes place in the soma of a human neuron model have been imitated [22,25,19,51,21]. ONNs or SelfONNs employ "Nodal" operations, which are analogous to synaptic connections, and "Pool" operations, which are analogous to integration in the soma, although "Activation" operators have been directly adopted. During training, the operator can self-organize and generate any family of nodal operators [24]. Fig. 8 illustrates the operation of SelfONN with nodal operator Ψ and pooling operator P. If y^{n-1} is the input to m^{th} neuron of n^{th} layer, the output x_m^n can be calculated from Equation (1).

$$x_m^n = P\left(\sum_{i=1}^{N_{n-1}} \psi_{mi}^n(\omega_{mi}^n, y_i^{n-1})\right) \tag{1}$$

From Equation (1), ω_{mi}^n is the weights. Here, ω is an array of parameters in q dimensions made up of internal parameters and weights for each

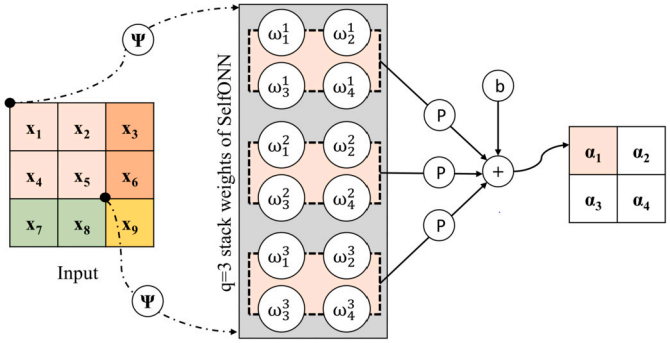


Fig. 8. SelfONN operation on a input feature with nodal operator Ψ and pooling operator P.

unique function. The nodal operator can be approximated by Taylor Series approximation. So, the approximation can be written as,

$$f(x) = f(x_0) + \frac{f'(x_0)}{1!}(x - x_0) + \frac{f''(x_0)}{2!}(x - x_0)^2 + ... + \frac{f^q(x_0)}{q!}(x - x_0)^q \tag{2}$$

$$f(x) = f(0) + \frac{f'(x_0)}{1!}(x) + \frac{f''(x_0)}{2!}(x)^2 + ... + \frac{f^q(x_0)}{q!}(x)^q \tag{3}$$

$$f(x) = b + \omega_1(x) + \omega_2(x)^2 + ... + \omega_q(x)^q \tag{4}$$

Here, b is the bias. If a $tanh$ activation is applied to the overall feature map or the Equation (1), the approximation can be bounded between $[-1, 1]$. The $tanh$ function and its application on Equation (1) can be shown as,

$$tanh = \frac{1 - e^{-2x}}{1 + e^{-2x}} \tag{5}$$

$$Actx_m^n = tanh(x_m^n) \tag{6}$$

$$Actx_m^n = \frac{1 - e^{-2x}}{1 + e^{-2x}}(P(\sum_{i=1}^{N_{n-1}} \psi_{mi}^n(\omega_{mi}^n, y_i^{n-1}))) \tag{7}$$

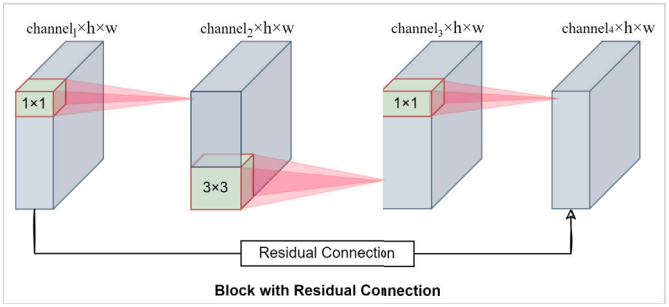
SelfONN exhibits better performance than CNN in many tasks [25, 19,51]. Therefore, as shown in Fig. 1, SelfONN generates a q set of weights; in this study, $q = 3$ is employed; all of these weights are then pooled via a pooling operation that incorporates bias to construct the resulting feature map.

In this study, two types of blocks are used to construct the model architecture, 1) Inverted residual Block, and 2) Non-residual block. Fig. 9 represents the two building blocks of Self-ChakmaNet. Both of these blocks have 1×1 point-wise convolution, 3×3 convolution, and 1×1 point-wise convolution sequentially, but only inverted residual blocks has the residual connection to counter the vanishing gradient. Another difference between these two blocks is in the middle layer of 3×3 convolution of non-residual block has stride of 2 which is returns reduce the spatial dimensional of features from $(height \times width)$ to $(\frac{height}{2} \times \frac{width}{2})$.

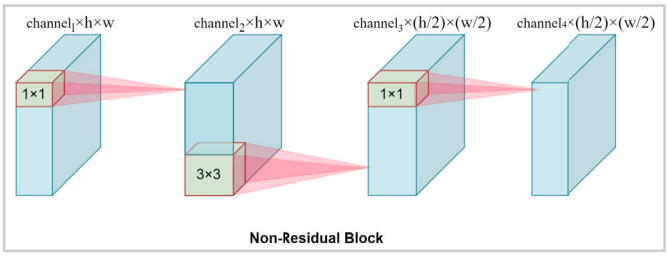
The Self-ChakmaNet architecture consists of five inverted residual blocks and four non-residual blocks. The spatial dimension of the input image is reduced by four non-residual blocks by the factor of 2 and before flatting the feature at the end average pooling of 7 is used. For the classification part, Self-MLP is used which is a variant of SelfONN. Unlike the identical "linear" neuron model of Multi-Layer Perceptrons (MLPs), Self-MLP also employs non-linearity and generative neurons as SelfONN. (See Fig. 10.)

3.2.4. Ethnic handwriting learning module: "Swakkhor App"

"Swakkhor App" is an app which helps users to learn indigenous language. This app is currently under development. Fig. 3 block D represents the app and how it is connected to the whole system. This app provides a canvas to draw "Chakma" characters. Users draw the character they are asked to draw and the trained model from handwritten



(a) Block with Residual Connection



(b) Block with no Residual Connection

Fig. 9. Building blocks of Self-ChakmaNet, a) Inverted residual block with residual connection and (b) Non-residaual block.

module recognizes the character. The probability of recognized character sets the achievement of user in developing the written skills of "Chakma" language. Every user is set draw a character for 10 times and based on recognized probabilities they progress to learn all the alphabets of "Chakma" language.

3.3. Visualization technique

Deep learning models are often considered as a black box. Knowing the attributes that a deep learning model uses for predictions is necessary to increase the model's credibility with its users. Utilizing different visualization techniques, the areas from which the networks generate decisions have been verified visually. CAM [52], GradCAM [41], SmoothGrad++ [30], and ScoreCAM [49] are the popular visualization techniques for deep learning models' decision interpretation. In this study, GradCAM visualization techniques is used. GradCAM [41] is an extensive version or generalization of Class Activation Mapping (CAM) [52]. CAM visualization is sensitive of particular deep learning models, which is a major drawback of CAM. GradCAM eliminates the CAM requirement of a fully connected layer be followed by a global average pooling [41]. GradCAm uses "alpha values" that are computed based on gradients to weight the feature maps [41]. The hot part in the visualization using GradCAM represents the "class-discriminative localization map" or the heatmap. However, GradCAM was chosen for this work because of its promising results in contemporary computer vision research. It generates heatmaps to the parts of the input image that model considers when making predictions. This could make it easier for users to understand how the network makes predictions.

4. Experimental setup

In deep learning experiment, the dataset needs to be divided into training, validation, and testing set. The "Indigenous Handwritten Character Dataset- Chakma" contains 1,000 images for each 47 classes which is in total of 47,000 images. The dataset was divided into training, validation, and test by the ratio of 70%, 10%, 20% of each class respectively. The illustration of the data splitting is given in Fig. 11. As a result, 32,900 images were used for the training of deep learning models in classifying 47 handwritten Chakma Character. For hyper-parameter

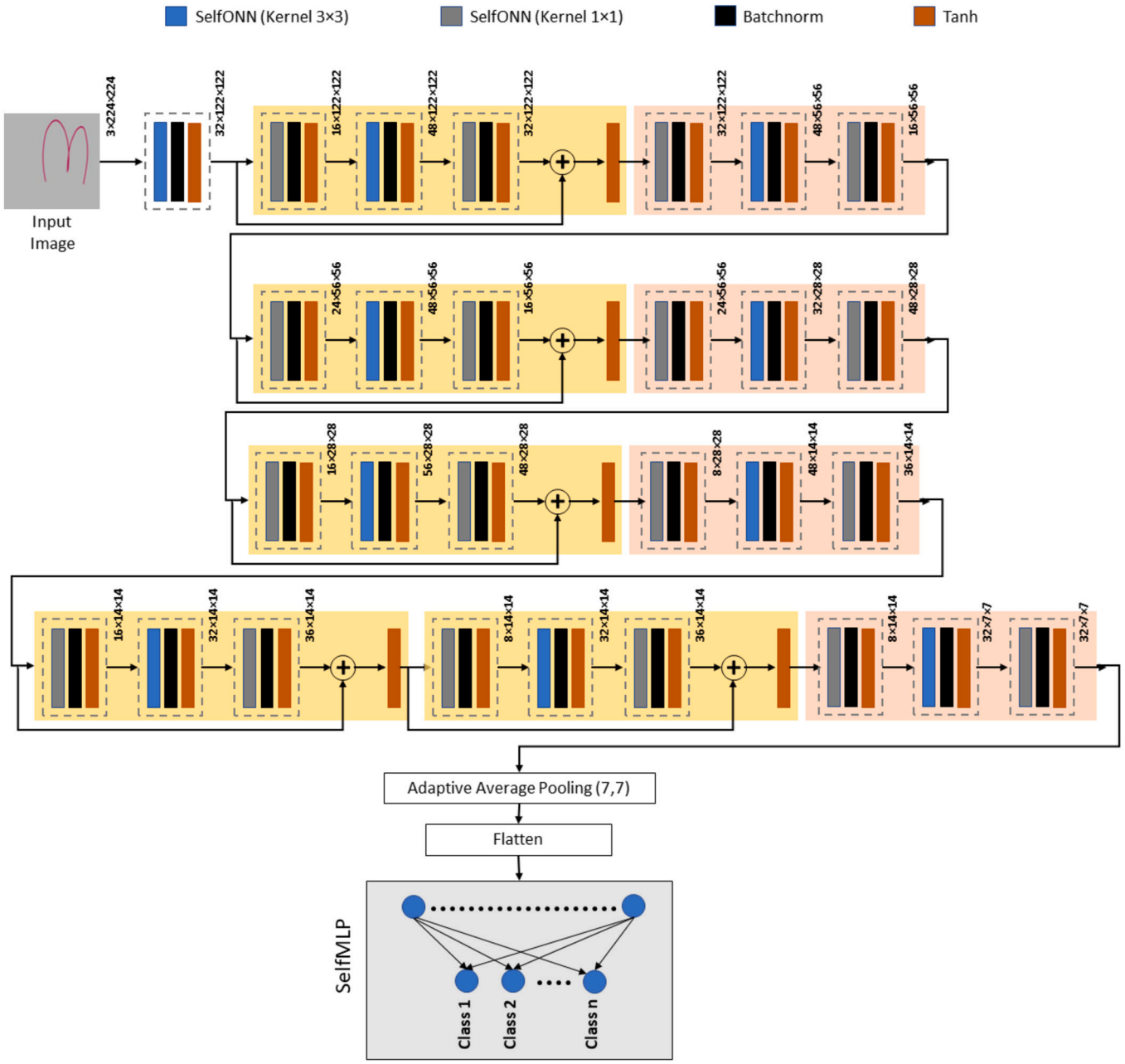


Fig. 10. Architecture of Self-ChakmaNet with inverted residual blocks and non-residual blocks with Self-MLP.

tuning (learning rate drop, early stopping) of the model 4700 images were used while the models were evaluated on 9400 images.

The model under consideration underwent training and testing on additional datasets pertaining to the recognition of handwritten characters in the Bengali language. For Bangla handwritten character recognition, the Self-ChakmaNet model was trained on single fold of the datasets. The dataset known as CMATERdb 3.1.2 consisted of a total of 15,000 images, which were utilized for the purpose of training and validation (12,000 images) as well as testing (3,000 images) of 50 fundamental Bangla characters. The CMATERdb 3.1.1 dataset consisted of 4,000 samples for both training and validation, and 2,000 images for testing purposes. The dataset encompassed 10 distinct Bangla numerals. The BanglaLekha Numerals refer to a specific subset of the original BanglaLekha dataset, consisting solely of images related to the ten Bangla numerals. The BanglaLekha Numerals dataset underwent partitioning into three distinct sets, namely the training set, validation set, and test set. The dataset consisted of a total of 13,833 images for the training set, 1975 images for the validation set, and 3949 images for the test set. In order to maintain consistency with the other two numeral datasets, the images of BanglaLekha numerals were inverted to ensure that the black stroke remained on a white canvas.

Data cleaning was performed exclusively on the ISI numerals and EKush datasets. Following the cleaning process, the ISI numerals dataset was reduced to 19,392 images for the training and validation set, and 4,000 images for the testing set, resulting in a total of 23,392 images. It is important to mention that the original dataset prior to cleaning contained 27,500 images. Conversely, the Ekush dataset underwent a cleaning process resulting in a total of 17,745 images allocated to the training set, 5,053 images assigned to the test set, and 2,530 images designated for the validation set.

Datasets containing numerals and basic characters from the same domain were assessed against another dataset. The evaluation process involved utilizing the complete dataset to assess the performance of the trained model across various datasets. For instance, the Self-ChakmaNet

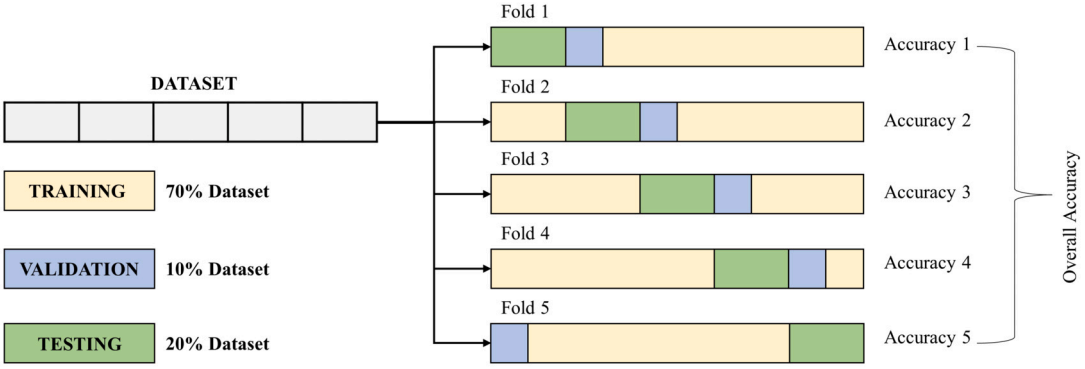


Fig. 11. An illustration of training, validation and test set splitting from main dataset using cross validation.

Table 2
Details of training parameters of MobileNet_V2,and Self-ChakmaNet.

Training Parameters	MobileNet_V2	Self-ChakmaNet
Batch Size	4	4
Learning rate	0.0001	0.0001
Learning Rate Drop factor	0.1	0.1
Normalization	True	False
Max Epochs	100	100
Epoch Patience	5	5
Early Stopping	13	13
Optimizer	Adam	Adam

model was trained on 3,600 samples from the CMATERdb 3.1.1 dataset, with 400 samples reserved for validation purposes, specifically in the numeral's domain. The performance of that model was evaluated on two numeral datasets, namely BanglaLekha Numerals and ISI numerals, consisting of 19,757 and 23,392 samples, respectively. Additionally, for basic character recognition for Bangla, the Self-ChakmaNet was trained using 10,800 training samples from the CMATERdb 3.1.2 dataset, with an additional 1,200 validation samples. Subsequently, that model was tested separately on 25,328 samples from the Ekush dataset. The same method of unseen data evaluation was done on other dataset considered for Bangla handwritten character recognition.

This research is done using Pytorch library with Python 3.7. All the model was trained in Google Colab Pro. The specifications were used through Google Colab for this experiment were 16GB Tesla T4 GPU, and 120GB High RAM.

4.1. Hyperparameters

The training parameters were used in this experiment is given in Table 2.

4.2. Evaluation metrics

Deep learning curves are widely used to examine trends in the learning of models (optimizing the parameters) versus each epoch or over time. There are two categories into which learning curves may be divided: optimization learning curves and performance learning curves. While the model performance or accuracy plotted curve is known as the Performance Learning Curves, the learning curves including the optimization parameters or loss of the model are known as Optimization Learning Curves. A model's overfit, underfit, and well-fit characteristics may be understood by comparing training learning curves to trends in accuracy and loss during validation and testing. Evaluation metrics of the model are another way to investigate at the performance of the models. Proposed SelfONN based model and the mentioned architectures' performances were estimated through evaluation metrics, such as overall and weighted accuracy, Sensitivity, Specificity, Precision and F1_score. The terms used to define the evaluation metrics are noted below:

β = number of true positive instances,
κ = number of false-positive instances,
ζ = number of true negative instances, and
η = number of false-negative instances.

Here, a deep learning model's precision—or the standard for a correct prediction—is one measure of how accurate the model performs. To calculate precision, divide the total number of true positive predictions by the total number of true positives:

$$Precision = \frac{\beta}{\beta + \kappa} \tag{8}$$

Specificity is another criterion for assessing deep learning models. The specificity is defined as the ratio of true predicted negatives to negatively identified samples which can be expressed as:

$$Specificity = \frac{\zeta}{\zeta + \kappa} \tag{9}$$

Sensitivity is the proportion of test samples that were properly predicted in positive class samples. The model performance on indemnifying positive instances for positive classes, or Sensitivity can be written as:

$$Sensitivity = \frac{\beta}{\beta + \eta} \tag{10}$$

where the F1-score is an important evaluation metric in deep learning. The harmonic mean of sensitivity/recall and precision is the F1 score. By combining two apparently at variance criteria—precision and sensitivity/recall—it concisely sums up a model's predictive ability.

$$F1\_Score = \frac{2 \times (Precision \times Sensitivity)}{Precision + Sensitivity} \tag{11}$$

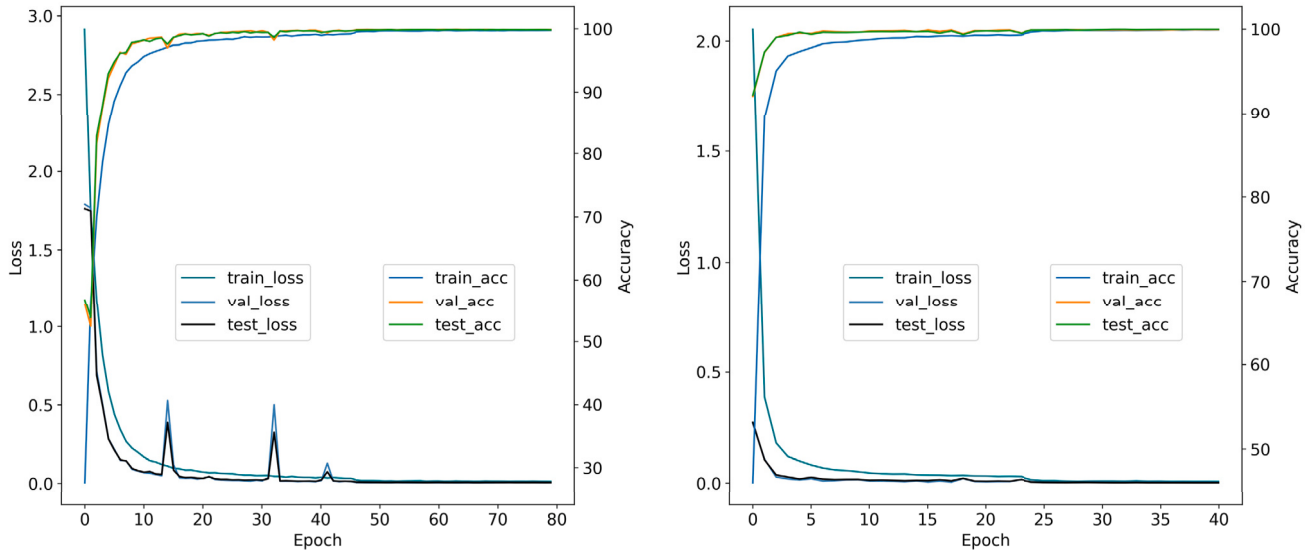
Finally, the overall accuracy is the percentage of true positives, true negatives, false positives, and false negatives combined.

$$OverallAccuracy = \frac{\beta}{\beta + \zeta + \kappa + \eta} \tag{12}$$

A receiver operating characteristic curve (ROC curve) is a graph that plots the true positive rate and false positive rate to show how well a classification model performs across all classification thresholds. The two-dimensional region between 0 and 1 beneath a ROC curve is known as the area under the curve (AUC). A model can distinguish between true positive and negative classifications better the higher the value of AUC:

$$FalsePositiveRate = \frac{\chi}{\tau + \chi} \tag{13}$$

Additionally, two criteria for evaluation that show how light or heavy and fast the model performs, are the total number of trainable parameters and the inference time. Trainable parameters are those which value is adjusted/modified during training as per their gradient. The more number of trainable parameters indicates the model is heavy, while less



(a) Learning curve of Self-ChakmaNet (b) Learning curve of MobileNet_V2

Fig. 12. Learning curves of first folds evaluation of (a) Self-ChakmaNet aND (B) MobileNet_V2. The primary y-axis in the left side represents the Optimization learning curves, and secondary y-axis in the right side represents the Performance learning curve of the models.

trainable parameters indicate the lightness of the model. Inference time indicates the processing time the model takes to predict a single prediction over a single sample. Let, t is the processing time for predicting one single pre-processed sample and the sample is predicted $N = 1000$ times, then inference time T_{inf} can be written as follows,

$$T_{inf} = \frac{\sum_{i=1}^{N=100} t_i}{N} \tag{14}$$

5. Result analysis

The performance of MobileNet_V2, and Self-ChakmaNet is reported in this section. The performance of MobileNet_V2, and Self-ChakmaNet is evaluated with learning curve, and comparative result analysis of existing other handwritten character recognition models.

5.1. Learning curves comparison

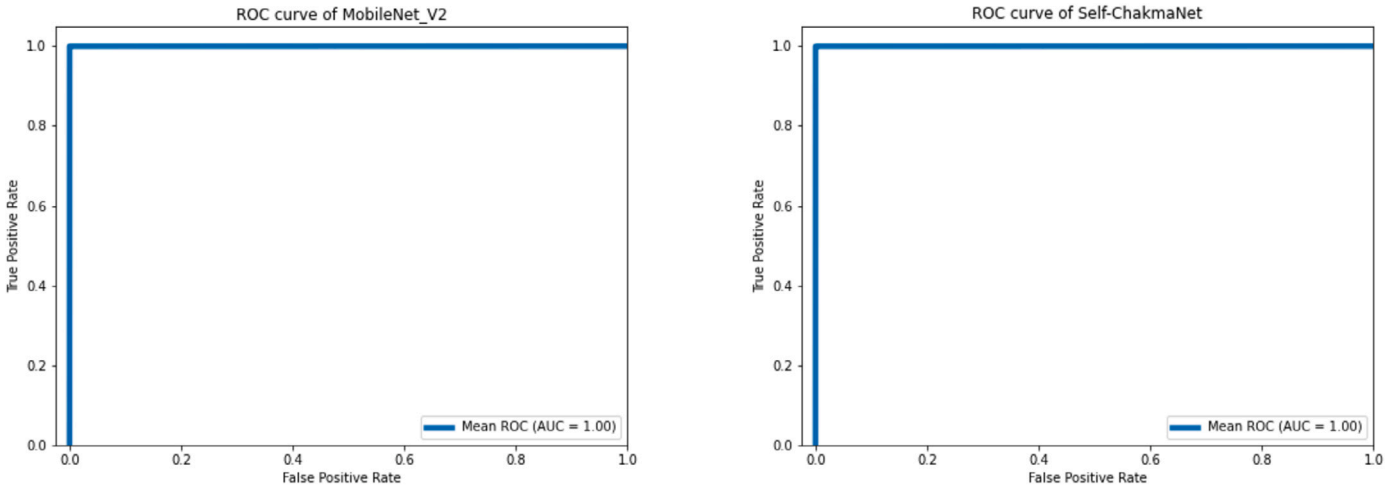
Learning curve analysis helps to diagnose a complex deep learning model for different scenarios of underfitting, overfitting, and well-fitting characteristics. Fig. 12 represents the learning curves of first fold of the models. All the learning curves of both of the models for all folds are available in Supplementary Table S1 and Supplementary Table S2. From Fig. 12 and Supplementary Table S1, it is evident that MobileNet_V2 converge earlier than Self-ChakmaNet. Self-ChakmaNet converged perfectly after few more epochs than MobileNet_V2, but showed the trend of getting almost saturated between 20 to 30 epochs. As the loss plot is not a flat line at higher loss and the loss got saturated at earlier epochs, leaving no opportunity for improvement, the optimization learning curves of MobileNet_V2 and Self-ChakmaNet support that the models are not underfitted. Additionally, the models are not overfitted as the model gradually learned features from training data, the models became more generalized in the new data such as validation and test set. Such an assertion is supported by the models' improving accuracy and declining loss in the test and validation sets as training proceeds. The learning curves in Fig. 12, Supplementary Table S1 and Supplementary Table S2 also represent that the validation and test set is a well representation of the problem statement. A well-fit model is defined by a training, validation and test loss that declines or accuracy that improved to a stable point with a small difference between the two final loss values. In early epochs, the Self-ChakmaNet optimization learning curves exhibited notches for validation and test, but as the epoch continued on, the loss reduced toward zero with a saturation trend. Overall, the Self-ChakmaNet is well-fitted model as state-of-the-art MobileNet_V2 for Chakama Handwritten Character recognition.

5.2. Evaluation metrics comparison

Accuracy, precision, f1 score, sensitivity/recall, and specificity were the evaluation metrics used for performance evaluation of trained models. All these matrics can be formulated using $\beta, \kappa, \zeta, and \eta$ used in Equation (12), (11), (8), (10), and (9). Supplementary Figure S1 and Supplementary Figure S3 contain the confusion matrics of MobileNet_V2 and Self-ChakmaNet and using confusion matrics $\beta, \kappa, \zeta, and \eta$ were calculated for Accuracy, precision, f1 score, sensitivity/recall, and specificity formulation. All the trained models were tested on the test set which comprises 20% of the entire dataset. The number of instance for each class was considered for metrics calculation. Table 3 represents the comparison of the two models. The best accuracy, precision, f1 score, sensitivity/recall, and specificity were achieved by MobileNet_V2. The Self-ChakmaNet performed similar to MobileNet_V2 with only 0.06% decrease in accuracy, precision, f1 score, and sensitivity/recall. The close precision and recall values of Self-ChakmaNet to MobileNet_V2 represents that the models are capable at predicting true positive instances for multi-class classification. The f1 score of the both models which is a combined metric of precision and recall, also indicates the models superiority of true positive instance (for multi-class classification) classification over the false positive and false negative. Though the self-ChakmaNet under-performed by a very slight margin for accuracy, precision, f1 score, and sensitivity/recall than MobileNet_V2. Self-ChakmaNet was highly capable of predicting true negative instances (for multi-class classification) over all negative instances as the specificity of Self-ChakmaNet is 100%. So, on five fold cross validation of Chakma Handwritten character recognition, MobileNet V2 and Self-ChakmaNet performed similarly in evaluation metrics.
However, comparing trainable parameter and inference time reveals the difference in architecture and processing speed. MobileNet V2 has over 5.037 times more trainable parameters than Self-ChakmaNet, which has 453,443 trainable parameters. MobileNet_V2 is one of the lightest state-of-the-art CNN architecture for implementation on mobile devices.

Table 3
Result Analysis of MobileNet_V2 and Self-ChakmaNet on Chakma Handwritten Character recognition.

Model	Inference Time(ms)	Trainable Parameters	Accuracy	Precision	F1 Score	Sensitivity	Specificity
MobileNet_V2	8.4801	2,284,079	**99.90**	**99.90**	**99.90**	**99.90**	**100**
Self-ChakmaNet	**8.0775**	**453,443**	99.84	99.84	99.84	99.84	**100**



(a) ROC curves on MobileNet_V2

(b) ROC curves on Self-ChakamNet

Fig. 13. Receiver Operating Characteristic (ROC) curves of (a) MobileNet_V2 and (b) Self-ChakmaNet. A high resolution version of these ROC curves can be found in the Supplementary Figure S2 and Supplementary Figure S4.

Self-ChakmaNet mimicked the MobileNet V2 basic blocks but had less trainable parameters and operation neurons, which allowed it to compute predictions more swiftly than MobileNet V2. Self-ChakmaNet outperformed MobileNet V2 by having an inference time of 8.0775 ms as opposed to 8.4801 ms. Self-ChakmaNet and MobileNet V2 are comparable models overall, with Self-ChakmaNet being lighter and faster, but also performing similarly accurate and efficiently.

5.3. ROC curves and AUC comparison

The ROC curves for the MobileNet V2 and Self-ChakmaNet are shown in Fig. 13 by showing the True Positive Rate and False Positive Rate at various thresholds. A lower X-axis value on the ROC curve of Fig. 13 indicates a lower proportion of True negatives to False positives. A higher Y-axis number, on the other hand, indicates a higher ratio of True positives to False negatives. Both ROC curves of MobileNet V2 and Self-ChakmaNet have an Area Under the Curve (AUC) of 1.00, which indicates that both models are capable of correctly classifying the sample across all classification thresholds.

5.4. GradCAM visualization of the MobileNet_V2 and self-ChakmaNet predictions

Fig. 14 represent the visualization analysis of Chakma Handwritten character recognition using GradCAM. GradCAM is used in this study to understand the decision making features for Chakma Handwritten character recognition by MobileNet_V2 and Self-ChakmaNet. The heatmap generated using GradCAM for four different Chakma character are given in Fig. 14. The four heatmaps of MobileNet_V2 for four characters represent that the model is selecting the appropriate features or the character region pixels for the classification. The character stroke region in the visualization using GradCAM is hotter or mapped as red for the true classification by MobileNet_V2. In comparison with visualization of Self-ChakmaNet predictions, the heatmap is not as spread over the region of the Chakma characters as MobileNet_V2. However, it is evident from the Self-ChakmaNet visualizations that Self-ChakmaNet predictions based on the stroke pattern of the character. All the predictions of Self-ChakmaNet follows the stroke pattern of the character with hotter or red mapping in the heatmap. All these visualization outcomes support the interpretability of the models. These models are not making predictions on arbitrary features, rather than focusing important features as stroke pattern of the handwritten character. Overall, the Self-ChakmaNet is classifying the instances as MobileNet_V2 with a high degree of accuracy as well as from the relevant features.

5.5. Comparative result analysis with existing literature

As this research is the first approach of Chakma Handwritten Character recognition model, there is no other literature to compare. But, as handwritten character recognition problem, Table 4 illustrates the comparison of our proposed models with other handwritten character recognition models. Previous literatures on Bangla Handwritten character recognition are considered for comparison. From Table 4, authors in literature [38] used classical machine learning approach for Bangla Handwritten character recognition. In the literature [15,20], Convolutional Neural Networks were adopted to gain significantly good result on Bangla handwritten character recognition. All the CNN models in [15,20] have more number of trainable parameters than Self-ChakmaNet.

For comparative analysis and the evaluation of the proposed model across the different character recognition dataset, the Self-ChakmaNet was trained and tested on five different Bangla handwritten character datasets, such as CMATERdb 3.1.1(numerals), ISI numerals, BanglaLekha Numerals, CMATERdb 3.1.2 (basic characters), and Ekush (basic characters). The models trained with one dataset were tested on the test set of the same dataset along with other dataset/datasets of same domain, such as a model trained on numerals dataset was tested on other two numerals dataset. All the evaluation is tabulated in Table 4. Self-ChakmaNet achieved 97.30% accuracy on the test set of the ISI numerals dataset while the best results were on this dataset was 95.10%, 99.78%, and 99.36% reported in literature [5], [45], and [43] respectively. The result represents that Self-ChakmaNet outperformed literature [5], but under-performed 2.48% and 2.06% than literature

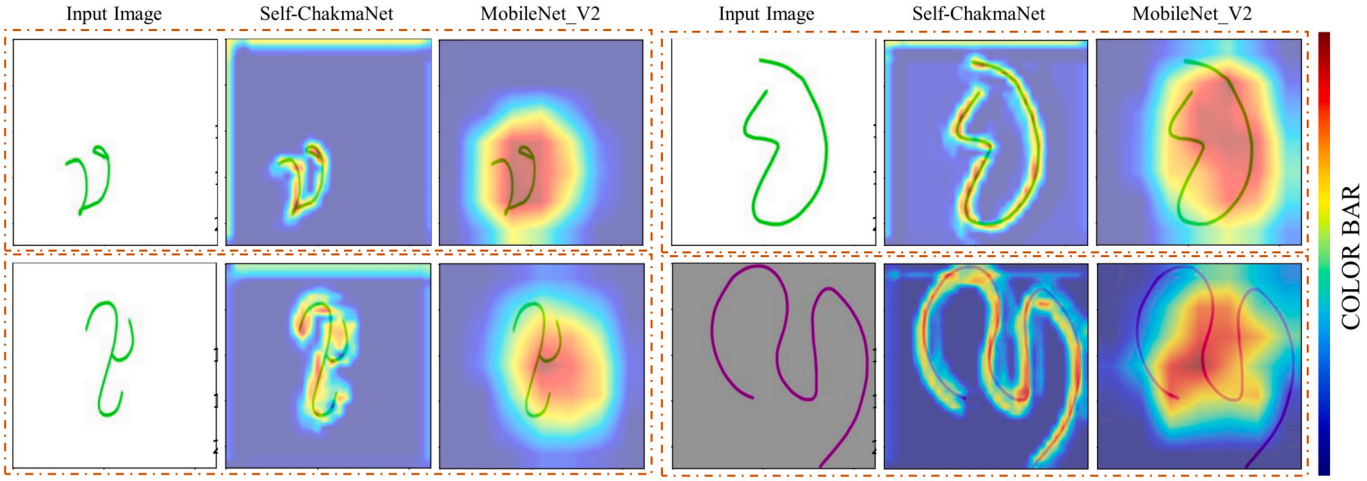


Fig. 14. GradCAM visualization of MobileNet_V2 and Self-ChakmaNet model on recognizing the Chakma Handwritten Character dataset.

Table 4
Comparative analysis of handwritten character recognition models.

References	Technique Used	Language	Dataset	Number of Classes	Accuracy
[38]	SVM	Bangla	[38]	50	83.68
[15]	Combination of InceptionResNetV2, InceptionNetV3 and DenseNet121	Bangla	CMATERdb	231	97.69
[20]	Squeeze and excitation ResNeXt	Bangla	BanglaLekha-Isolated 2 dataset	50	99.82
[5]	MLP	Bangla	ISI Numerals	10	95.10
[45]	BDNet CNN	Bangla	ISI Numerals	10	99.78
[36]	CNN	Bangla	CMATERdb 3.1.2	50	98.00
[36]	BornoNet CNN	Bangla	CMATERdb 3.1.2	50	98.00
[42]	CNN	Bangla	CMATERdb 3.1.1	10	99.50
[40]	BengaliNet CNN	Bangla	CMATERdb 3.1.1	10	99.01
			CMATERdb 3.1.2	50	98.97
			BanglaLekha Numerals	50	98.97
			Ekush	50	98.36
[43]	Skip-connected Multi-column CNN	Bangla	CMATERdb 3.1.1	10	98.15
			ISI Numerals	10	99.36
			CMATERdb 3.1.2	50	96.65
Our proposed method	Self-ChakmaNet	Chakma	Indigenous Handwritten Character Dataset - Chakma	47	99.84
		Bangla	CMATERdb 3.1.1	10	96.08
			Trained: CMATERdb 3.1.1 Tested: BanglaLekha Numerals	10	81.26
			Trained: CMATERdb 3.1.1 Tested: ISI Numerals	10	91.25
		Bangla	BanglaLekha Numerals	10	95.67
			Trained: BanglaLekha Numerals Tested: CMATERdb 3.1.1	10	94.15
			Trained: BanglaLekha Numerals Tested: ISI Numerals	10	94.70
		Bangla	ISI Numerals	10	97.30
			Trained: ISI Numerals Tested: CMATERdb 3.1.1	10	98.68
			Trained: ISI Numerals Tested: BanglaLekha Numerals	10	93.10
		Bangla	CMATERdb 3.1.2	50	94.83
			Trained: CMATERdb 3.1.2 Tested: Ekush	50	91.61
		Bangla	Ekush	50	95.59
			Trained: Ekush Tested: CMATERdb 3.1.2	50	76.53

[45], and [43]. The total number of parameter of BDNet [45] was more than 1.71 millions and the model proposed in literature [43] had even more than this number of trainable parameters. Compared to these giant models, Self-ChakmaNet was trained on only 453k parameters and produced close accuracy to literature [43,45]. Similarly, CMATERdb 3.1.1 dataset was tested on the model trained on ISI dataset using Self-ChakmaNet which showed 98.68% accuracy. The accuracy achieved in literature [40] on testing CMATERdb 3.1.1 dataset was 99.01% with

more than 2.24 millions of parameters. In this instance, Self-ChakmaNet attained a level of accuracy that was nearly equivalent, while utilizing only one-fifth of the number of trainable parameters. Similar patterns are evident in the results of other evaluations (numerals, and basic characters datasets) presented in Table 4 for Self-ChakmaNet. This model demonstrated comparable performance to previous studies when tested on the same or different datasets that were not included in the training set.

The performance degradation of Self-ChakmaNet in Bangla Handwritten character recognition, as compared to other studies, may be attributed to the resizing of the input image. The input size of the proposed architectures exceeds the dimensions of all available datasets of Bangla Handwritten Characters. The increase in size of the input image resulted in the loss of data, which can be mitigated through various preprocessing techniques, including white padding, black padding, and others. As the scope of this study was limited to the Chakma Handwritten Character, the comprehensive analysis of these findings was intended to inform future research endeavors. The Self-ChakmaNet model has demonstrated a notable level of accuracy in recognizing handwritten Chakma characters when compared to other current studies in the field of Bangla handwritten character recognition. The Self-ChakmaNet model exhibited notable accuracy in the domain of handwritten character recognition, while utilizing only a small number of trainable parameters and demonstrating expedited processing capabilities.

6. Conclusion and future work

A significant number of languages are dying each year due to lack of availability of resource, education and practice. The learning and practice of indigenous language can be done digitally if the written characters can be recognized correctly and proper feedback given after practice. This research focused developing language resources which can help to develop deep learning systems to recognize Chakma characters. A SelfONN based complex model Self-ChakmaNet is proposed in this study and also the performance compared with state-of-the-art CNN model MobileNet_V2. The proposed model was also tested on five different Bangla handwritten characters' dataset. The aim of developing a lighter and faster model than MobileNet with same accuracy is fulfilled in this research. Therefore, Self-ChakmaNet may be used in the "Swakkhor App" for mobile device implementation in the future. In future, Indigenous Handwriting Recognition can be achievable using this method which will be helpful to build "Indigenous Language Learning" system along with other three modules such as listening, reading and speaking of different Indigenous languages. This approach can be replicated and expanded for other Indigenous languages.

Funding

This research received no external funding.

CRediT authorship contribution statement

Kanchon Kanti Podder: Conceptualization, Methodology, Software, Writing – Original draft preparation, Writing – Reviewing and Editing. **Ludmila Emdad Khan:** Conceptualization, Methodology, Software, Writing – Original draft preparation, Writing – Reviewing and Editing. **Jyoti Chakma:** Validation, Data collection, Writing – Reviewing and Editing. **Muhammad E. H. Chowdhury:** Conceptualization, Methodology, Supervision, Funding Acquisition, Writing – Original draft preparation, Writing – Reviewing and Editing. **Proma Dutta:** Validation, Writing – Original draft preparation, Writing – Reviewing and Editing. **Khan Md Anwarus Salam:** Conceptualization, Supervision, Writing – Original draft preparation. **Amith Khandakar:** Methodology, Software, Validation, Writing – Original draft preparation **Mohamed Arselene Ayari:** Validation, Supervision, Writing – Original draft preparation, Writing – Reviewing and Editing, Funding Acquisition. **Bikash Kumar Bhawmick:** Validation, Writing – Original draft preparation, Writing – Reviewing and Editing. **S M Arafin Islam:** Methodology, Software, Validation, Writing – Original draft preparation. **Serkan Kiranyaz:** Conceptualization, Supervision, Writing – Original draft preparation, Writing – Original draft preparation.

Declaration of competing interest

The authors declare that they have no conflict of interest.

Data availability

The dataset is publicly available in this link for further interest of the researchers around the globe.

Appendix A. Supplementary material

Supplementary material related to this article can be found online at https://doi.org/10.1016/j.eij.2023.100413.

References

[1] Afreen N. Language usage in different domains by the chakmas of Bangladesh. Int J Linguist Lit Transl 2020;3. https://doi.org/10.32996/ijllt.2020.3.6.13.
[2] Alom MZ, Sidike P, Hasan M, Taha TM, Asari VK. Handwritten bangla character recognition using the state-of-the-art deep convolutional neural networks. Comput Intell Neurosci 2018;2018:1–13. https://doi.org/10.1155/2018/6747098.
[3] Alom MZ, Sidike P, Taha TM, Asari VK. Handwritten bangla digit recognition using deep learning. CoRR, arXiv:1705.02680, 2017. http://arxiv.org/abs/1705.02680.
[4] Balaha HM, Ali HA, Youssef EK, Elsayed AE, Samak RA, Abdelhaleem MS, et al. Recognizing Arabic handwritten characters using deep learning and genetic algorithms. Multimed Tools Appl 2021;80:32473–509. https://doi.org/10.1007/s11042-021-11185-4.
[5] Basu S, Sarkar R, Das N, Kundu M, Nasipuri M, Basu DK. Handwritten bangla digit recognition using classifier combination through ds technique. In: Pal SK, Bandyopadhyay S, Biswas S, editors. Pattern recognition and machine intelligence. Berlin, Heidelberg: Springer Berlin Heidelberg; 2005. p. 236–41.
[6] Bhattacharya U, Chaudhuri B. Handwritten numeral databases of Indian scripts and multistage recognition of mixed numerals. IEEE Trans Pattern Anal Mach Intell 2009;31:444–57. https://doi.org/10.1109/TPAMI.2008.88.
[7] Bhattacharya U, Shridhar M, Parui SK, Sen PK, Chaudhuri BB. Offline recognition of handwritten bangla characters: an efficient two-stage approach. PAA Pattern Anal Appl 2012;15:445–58. https://doi.org/10.1007/s10044-012-0278-6.
[8] Biswas M, Islam R, Shom GK, Shopon M, Mohammed N, Momen S, et al. Banglalekha-isolated: a multi-purpose comprehensive dataset of handwritten bangla isolated characters. Data Brief 2017;12:103–7. https://doi.org/10.1016/j.dib.2017.03.035. https://www.sciencedirect.com/science/article/pii/S2352340917301117.
[9] Boric S, Schiebel E, Schloegl C, Hildebrandt M, Hofer C, Macht D. Research in autonomous driving – a historic bibliometric view of the research development in autonomous driving. Int J Innov Econ Dev 2021;7:27–44. https://doi.org/10.18775/ijied.1849-7551-7020.2015.74.2003.
[10] Bromham L, Dinnage R, Skirgård H, Ritchie A, Cardillo M, Meakins F, et al. Global predictors of language endangerment and the future of linguistic diversity. Nat Ecol Evol 2021;6:163–73. https://doi.org/10.1038/s41559-021-01604-y.
[11] Butarbutar R, Simatupang E. The impact of technology hello English application in efl classroom. Lingual, J Lang Cult 2020;8:11. https://doi.org/10.24843/LJLC.2019.v08.i02.p03.
[12] Cireşan DC, Meier U, Gambardella LM, Schmidhuber J. Deep, big, simple neural nets for handwritten digit recognition. Neural Comput 2010;22:3207–20. https://doi.org/10.1162/NECO_a_00052. https://direct.mit.edu/neco/article-pdf/22/12/3207/842857/neco_a_00052.pdf.
[13] Deusen-Scholl NV. Measuring Babbel's efficacy in developing oral proficiency; 2019.
[14] Gangaiamaran R, Pasupathi M. Review on use of mobile apps for language learning. Int J Appl Eng Res 2017;12:11242–51.
[15] Ghosh T, Abedin MHZ, Banna HA, Mumenin N, Yousuf MA. Performance analysis of state of the art convolutional neural network architectures in bangla handwritten character recognition. Pattern Recognit Image Anal 2021;31:60–71. https://doi.org/10.1134/s1054661821010089.
[16] Hoffmann M. Endangered languages, linguistics, and culture: researching and reviving the unami language of the lenape. Unpublished bachelor's thesis. Pennsylvania, PA: Bryn Mawr College; 2009.
[17] Jindal T, Bhattacharya U. Recognition of offline handwritten numerals using an ensemble of mlps combined by adaboost. In: Proceedings of the 4th international workshop on multilingual OCR; 2013. p. 1–5.

[18] Kavallieratou E, Liolios N, Koutsogeorgos E, Fakotakis N, Kokkinakis G. The gruhd database of Greek unconstrained handwriting. In: Proceedings of sixth international conference on document analysis and recognition, IEEE; 2001. p. 561–5.
[19] Keleş O, Tekalp AM, Malik J, Kıranyaz S. Self-organized residual blocks for image super-resolution. arXiv:2105.14926, 2021.
[20] Khan MM, Uddin MS, Parvez MZ, Nahar L. A squeeze and excitation resnext-based deep learning model for bangla handwritten compound character recognition. J King Saud Univ, Comput Inf Sci 2021. https://doi.org/10.1016/j.jksuci.2021.01.021. https://www.sciencedirect.com/science/article/pii/S1319157821000392.
[21] Kiranyaz S, Ince T, Iosifidis A, Gabbouj M. Operational neural networks. Neural Comput Appl 2020;32:6645–68. https://doi.org/10.1007/s00521-020-04780-3.
[22] Kiranyaz S, Malik J, Abdallah HB, Ince T, Iosifidis A, Gabbouj M. Self-organized operational neural networks with generative neurons. Neural Netw 2021;140:294–308. https://doi.org/10.1016/j.neunet.2021.02.028. https://www.sciencedirect.com/science/article/pii/S0893608021000782.
[23] Klimova B. The importance of writing. Paripex, Indian J Res 2012;2:9–11. https://doi.org/10.15373/22501991/JAN2013/4.
[24] Malik J, Kiranyaz S, Gabbouj M. Self-organized operational neural networks for severe image restoration problems. Neural Netw 2021;135:201–11. https://doi.org/10.1016/j.neunet.2020.12.014. https://www.sciencedirect.com/science/article/pii/S0893608020304391.
[25] Malik J, Kiranyaz S, Yamac M, Guldogan E, Gabbouj M. Convolutional versus self-organized operational neural networks for real-world blind image denoising. arXiv:2103.03070, 2021.
[26] Meier U, Ciresan DC, Gambardella LM, Schmidhuber J. Better digit recognition with a committee of simple neural nets. In: 2011 international conference on document analysis and recognition, IEEE; 2011. p. 1250–4.
[27] Munday P. The case for using duolingo as part of the language classroom experience. Rev. Iberoam. Educ. Distancia 2015;19. https://doi.org/10.5944/ried.19.1.14581.
[28] Nuralisah A, Kareviati E. The effectiveness of using memrise application in teaching vocabulary. PROJECT (Professional Journal of English Education) 2020;3:494. https://doi.org/10.22460/project.v3i4.p494-500.
[29] Nushi M, Jenabzadeh H. Busuu: a mobile app. The TESL Reporter Journal 2016;49:30–8.
[30] Omeiza D, Speakman S, Cintas C, Weldermariam K. Smooth grad-cam++: an enhanced inference level visualization technique for deep convolutional neural network models. arXiv:1908.01224, 2019.
[31] Podder KK, Chowdhury M, Mahbub Z, Kadir M. Bangla sign language alphabet recognition using transfer learning based convolutional neural network. Bangladesh J Sci Ind Res 2020;31–33:20–6.
[32] Podder KK, Chowdhury MEH, Tahir AM, Mahbub ZB, Khandakar A, Hossain MS, et al. Bangla sign language (bdsl) alphabets and numerals classification using a deep learning model. Sensors 2022;22. https://doi.org/10.3390/s22020574. https://www.mdpi.com/1424-8220/22/2/574.
[33] Podder KK, Tabassum S, Khan LE, Salam KMA, Maruf RI, Ahmed A. Design of a sign language transformer to enable the participation of persons with disabilities in remote healthcare systems for ensuring universal healthcare coverage. In: 2021 IEEE technology engineering management conference - Europe (TEMSCON-EUR); 2021. p. 1–6.
[34] Purkaystha B, Datta T, Islam MS. Bengali handwritten character recognition using deep convolutional neural network. In: 2017 20th international conference of computer and information technology (ICCIT); 2017. p. 1–5.
[35] Rabby ASA, Haque S, Islam MS, Abujar S, Hossain SA. Ekush: a multipurpose and multitype comprehensive database for online off-line bangla handwritten characters. In: Santosh KC, Hegadi RS, editors. Recent trends in image processing and pattern recognition. Singapore: Springer Singapore; 2019. p. 149–58.
[36] Rabby ASA, Haque S, Islam S, Abujar S, Hossain SA. Bornonet: bangla handwritten characters recognition using convolutional neural network. Proc Comput Sci 2018;143:528–35. https://doi.org/10.1016/j.procs.2018.10.426. https://www.sciencedirect.com/science/article/pii/S1877050918321240. 8th International Conference on Advances in Computing & Communications (ICACC-2018).
[37] Rahman T, Khandakar A, Abir FF, Faisal MAA, Hossain MS, Podder KK, et al. Qcovsml: a reliable Covid-19 detection system using cbc biomarkers by a stacking machine learning model. Comput Biol Med 2022;143:105284. https://doi.org/10.1016/j.compbiomed.2022.105284. https://www.sciencedirect.com/science/article/pii/S0010482522000762.
[38] Riasat Azim, Fazlul Karim M, Rahman Wahidur. Bangla hand written character recognition using support vector machine. https://zenodo.org/record/60329, 2016. https://doi.org/10.5281/ZENODO.60329.
[39] Sandler M, Howard A, Zhu M, Zhmoginov A, Chen LC. Mobilenetv2: inverted residuals and linear bottlenecks. arXiv:1801.04381, 2019.
[40] Sayeed A, Shin J, Hasan MAM, Srizon AY, Hasan MM. BengaliNet: a low-cost novel convolutional neural network for Bengali handwritten characters recognition. Appl Sci 2021;11:6845. https://doi.org/10.3390/app11156845.
[41] Selvaraju RR, Cogswell M, Das A, Vedantam R, Parikh D, Batra D. Grad-CAM: visual explanations from deep networks via gradient-based localization. Int J Comput Vis 2019;128:336–59. https://doi.org/10.1007/s11263-019-01228-7.
[42] Shopon M, Mohammed N, Abedin MA. Bangla handwritten digit recognition using autoencoder and deep convolutional neural network. In: 2016 international workshop on computational intelligence (IWCI); 2016. p. 64–8.
[43] Singh A, Sarkhel R, Das N, Kundu M, Nasipuri M. A skip-connected multi-column network for isolated handwritten bangla character and digit recognition. arXiv:2004.12769, 2020.
[44] Song W, Uchida S, Liwicki M. Comparative study of part-based handwritten character recognition methods. In: 2011 international conference on document analysis and recognition, IEEE; 2011. p. 814–8.
[45] Sufian A, Ghosh A, Naskar A, Sultana F, Sil J, Rahman MH. Bdnet: Bengali handwritten numeral digit recognition based on densely connected convolutional neural networks. J King Saud Univ, Comput Inf Sci 2022;34:2610–20. https://doi.org/10.1016/j.jksuci.2020.03.002. https://www.sciencedirect.com/science/article/pii/S1319157820303220.
[46] Sutramiani NP, Suciati N, Siahaan D. Mat-agca: multi augmentation technique on small dataset for balinese character recognition using convolutional neural network. ICT Express 2021;7:521–9. https://doi.org/10.1016/j.icte.2021.04.005. https://www.sciencedirect.com/science/article/pii/S2405959521000497.
[47] Tahir A, Qiblawey Y, Khandakar A, Rahman T, Khurshid U, Musharavati F, et al. Deep learning for reliable classification of Covid-19, Mers, and Sars from chest x-ray images. Cogn Comput 2022:1–21. https://doi.org/10.1007/s12559-021-09955-1.
[48] Tran DT, Kiranyaz S, Gabbouj M, Iosifidis A. Knowledge transfer for face verification using heterogeneous generalized operational perceptrons. In: 2019 IEEE international conference on image processing (ICIP); 2019. p. 1168–72.
[49] Wang H, Wang Z, Du M, Yang F, Zhang Z, Ding S, et al. Score-cam: score-weighted visual explanations for convolutional neural networks. arXiv:1910.01279, 2020.
[50] Yin F, Wang QF, Zhang XY, Liu CL. Icdar 2013 Chinese handwriting recognition competition. In: 2013 12th international conference on document analysis and recognition, IEEE; 2013. p. 1464–70.
[51] Yılmaz MA, Keleş O, Güven H, Tekalp AM, Malik J, Kıranyaz S. Self-organized variational autoencoders (self-vae) for learned image compression. arXiv:2105.12107, 2021.
[52] Zhou B, Khosla A, Lapedriza À, Oliva A, Torralba A. Learning deep features for discriminative localization. In: 2016 IEEE conference on computer vision and pattern recognition (CVPR); 2016. p. 2921–9.
[53] Zhu B, Zhou XD, Liu CL, Nakagawa M. A robust model for on-line handwritten Japanese text recognition. Int J Doc Anal Recognit 2010;13:121–31.