Article

A Fine-Tuned Hybrid Stacked CNN to Improve Bengali Handwritten Digit Recognition

Ruhul Amin [1], Md. Shamim Reza [1], Yuichi Okuyama [2], Yoichi Tomioka [2] and Jungpil Shin [2,*]

1 Department of Statistics, Pabna University of Science and Technology, Pabna 6600, Bangladesh; shamim.reza@pust.ac.bd (M.S.R.)
2 School of Computer Science and Engineering, The University of Aizu, Aizuwakamatsu 965-8580, Fukushima, Japan; okuyama@u-aizu.ac.jp (Y.O.); ytomioka@u-aizu.ac.jp (Y.T.)
* Correspondence: jpshin@u-aizu.ac.jp

Abstract: Recognition of Bengali handwritten digits has several unique challenges, including the variation in writing styles, the different shapes and sizes of digits, the varying levels of noise, and the distortion in the images. Despite significant improvements, there is still room for further improvement in the recognition rate. By building datasets and developing models, researchers can advance state-of-the-art support, which can have important implications for various domains. In this paper, we introduce a new dataset of 5440 handwritten Bengali digit images acquired from a Bangladeshi University that is now publicly available. Both conventional machine learning and CNN models were used to evaluate the task. To begin, we scrutinized the results of the ML model used after integrating three image feature descriptors, namely Binary Pattern (LBP), Complete Local Binary Pattern (CLBP), and Histogram of Oriented Gradients (HOG), using principal component analysis (PCA), which explained 95% of the variation in these descriptors. Then, via a fine-tuning approach, we designed three customized CNN models and their stack to recognize Bengali handwritten digits. On handcrafted image features, the XGBoost classifier achieved the best accuracy at 85.29%, an ROC AUC score of 98.67%, and precision, recall, and F1 scores ranging from 85.08% to 85.18%, indicating that there was still room for improvement. On our own data, the proposed customized CNN models and their stack model surpassed all other models, reaching a 99.66% training accuracy and a 97.57% testing accuracy. In addition, to robustify our proposed CNN model, we used another dataset of Bengali handwritten digits obtained from the Kaggle repository. Our stack CNN model provided remarkable performance. It obtained a training accuracy of 99.26% and an almost equally remarkable testing accuracy of 96.14%. Without any rigorous image preprocessing, fewer epochs, and less computation time, our proposed CNN model performed the best and proved the most resilient throughout all of the datasets, which solidified its position at the forefront of the field.

Keywords: Bengali handwritten digit; handcrafted feature extractors; machine learning; deep learning; CNN; staking; recognition

check for updates

Citation: Amin, R.; Reza, M.S.; Okuyama, Y.; Tomioka, Y.; Shin, J. A Fine-Tuned Hybrid Stacked CNN to Improve Bengali Handwritten Digit Recognition. *Electronics* **2023**, *12*, 3337. https://doi.org/10.3390/electronics12153337

Academic Editors: D. J. Lee, Tomasz Trzcinski and Dong Zhang

Received: 22 July 2023
Revised: 1 August 2023
Accepted: 2 August 2023
Published: 4 August 2023

1. Introduction

In recent years, one of the most important tasks has been Handwritten Digit Recognition for various purposes, including bank checking of handwritten digits and conversion into machine-readable formats, vehicle number plate recognition for efficient monitoring and identification, NID (National Identification) verification, etc. Furthermore, the field of education extensively relies on handwritten digit recognition, especially in the realm of online educational support. Boosting the efficiency of digit-based data processing, enhancing security measures, and enabling seamless interaction with digital devices all depend on the accurate identification and classification of hand-drawn digits [1]. These tasks are possible through the development of machine learning. There are lots of fields where ML is extensively used, including computer vision, medical imaging, chemistry, physics, etc. [2–5].

Handwritten digit identification has received a lot of interest from the technological community in terms of online educational support. We can swiftly scan, translate, and search any sign, characters [6,7], or documents utilizing some apps, such as CamScanner, Google Lens, Adobe Scan, Microsoft Office Lens, and Scanbot [8,9]. The English, Chinese, Japanese, Arabic, and Spanish languages' recognition are becoming popular [10,11].The recognition rate has increased for computers and typewriters with high accuracies, but there has not been as much progress in the field of handwriting. The Bengali language is used by a large number of people for communication. As well as being extensively spoken in India, it serves as the official language of Bangladesh. According to the number of speakers, Bengali is the fifth most popular language in Asia and is among the top 10 most widely spoken languages worldwide [12]. There are 265 million people worldwide who speak Bengali, both native and non-native speakers [13]. The Bengali language is thousands of years old, but researchers have not taken the initiative to develop recognition techniques, as they have in English or other popular languages [14]. The English recognition task is less difficult due to the simplicity of its shape. Because of the size and structure of the Bengali digits, recognizing them is more difficult than recognizing English digits. With the advancement of technology in all sectors, every country like Bangladesh has been changing its working activity style, as have more technologically developed countries. By performing a thorough examination of the background and the issue of handwritten digit recognition, with an emphasis on Bengali digits specifically, this paper seeks to address these problems. Using a robust and trustworthy model, our research aims to improve the performance and accuracy of the recognition of handwritten Bengali digits.

The limitations and difficulties that now exist in handwritten digit recognition systems serve as the driving force behind this research. There is still room for improvement, especially in terms of the robustness, imbalanced datasets, flexibility with various writing styles, and generalization across datasets, even though prior studies have had varying degrees of success. Our research addresses these challenges, adding to the knowledge in the field, and gives significant recommendations for applications that rely on the recognition of specific handwritten digits. We aim to develop creative strategies and techniques that can increase handwritten digit recognition in the Bengali language by exploring the difficulties and complexities involved. The subsequent sections of this study explore various approaches and procedures used in handwritten digit recognition while providing an in-depth examination of the available literature. However, while concurrently addressing the unique needs and requirements of the Bengali language, we hope to contribute to the field of digit recognition as a whole through our study. Our ultimate objective is to provide a recognition system that accurately reads and understands handwritten Bengali digits through extensive experimentation and analysis. This study offers the following contributions:

- We produced a new Bengali handwritten digit dataset. The collection contains 5440 images.
- The collection strategy distinguishes this dataset from other ordinary datasets.
- We employed a robust handcrafted image feature extraction method on this dataset and applied the machine learning algorithm to recognize the images.
- To effectively and efficiently recognize the handwritten digits, we developed three customized convolutional neural networks (CNN) model using a fine-tuning approach and then stacked them with the produced CNN model layers.
- Our proposed CNN model three achieved the highest accuracy of 97.43%, a training accuracy of 99.66%, and a testing accuracy of 97.43% on our new dataset. And the proposed stacked model achieved the highest accuracy of 96.14%, a training accuracy of 99.26%, and a testing accuracy of 96.14% on another dataset.
- To validate our study, we performed cutting-edge action on several research datasets of Bengali handwritten digits.

This research paper proceeds as follows: Section 1 provides an introduction, and the related work is presented in Section 2. The Bengali handwritten digit dataset description is illustrated in Section 3. The handcrafted feature extractor and learning algorithm are described in Section 4. In Section 5, we detail the Bengali handwritten digit recognition strategy. In Section 6, the proposed methodology is described. Section 7 describes the experimental results and discussion. Finally, we draw conclusions and suggest future work.

2. Related Work

There are several studies on Bengali handwritten digit recognition. Liu et al. [11] implemented the reorganization techniques on three datasets, the ISI Bengali numeral dataset, the CENPARMI Farsi numerals dataset, and the IFHCDB Farsi numerals dataset. They worked on three types of images, used three normalization techniques, and implemented six classifiers. Within the six classifiers, the four classifiers of the SVM, the discriminative learning quadratic discriminant function (DLQDF), the polynomial network classifier (PNC), and the class-specific feature polynomial classifier (CFPC) gave the highest results. On grayscale images, the CFPC had the best recognition test accuracy of 99.40%. By leveraging the distribution of local stroke orientation/direction for image feature extraction, the author of this study was able to effectively capture the image texture variable and structural features. Even though the local stroke distribution was efficient, the author may have used handcrafted image feature extraction techniques that could help advance state-of-the-art recognition. To recognize the handwritten digit, Sufian et al. [12] utilized the ISI dataset and their own generated dataset named the BDNet dataset. In this dataset, they used deep convolutional neural networks (CNNs) on the training, testing, and cross-validation methods.

The ISI dataset remained in the training phase while the BDNet was tested, resulting in a maximum training accuracy of 99.80% and testing accuracy of 99.78%, respectively, and the maximum cross validation achieved an accuracy of 100% after the 97th, 107th, 118th, and 120th epochs. Regarding their work, they achieved high accuracy in training and testing recognition performance.Despite its impressive performance, the customized deep learning model is not suitable for all aspects of digit recognition for the iteration process because it takes more time for each epoch. Alam et al. [14] worked to assemble six different databases, including the BHDDB (owner-BUET) dataset, the BUET101 Database, the Ongko database (owner-BUET), the ISRT (Institute of Statistical Research and Training, Dhaka University) database, the BanglaLekha-Isolated Numerals database, and the UIU (United International University, Bangladesh) database, and they combined the whole dataset (85,000 images) named the NumtaDB dataset. The total database was gathered based on age, geographic region, and sex, and it was split into training and testing sets with an 85% to 15% ratio, respectively, without the UIUDB, which was retained at 100% for the testing phase. By assembling six distinct datasets gathered through the work of various researchers, the author made an important contribution. This effort also offered valuable evaluation and may be a significant contribution to the data science community. However, they did not use any predictive models for digit data recognition in this work.

Islam et al. [15] employed three transfer learning models including Resnet-50, Inception-v3, and EfficientNetB0, to classify the Bengali handwritten digits. To classify Bengali digits, a novel approach was used to modify the pretrained model's layer so that the input layer size was changed to 96 × 96 and the output layer to 10 classes. Using the Resnet-50 model, a training accuracy of 97% was achieved, and the testing accuracy was 96%; the Inception-v3 model training accuracy was 90%, and the testing accuracy was 88%; the EfficientNetB0 model achieved a training accuracy of 95%, and the testing accuracy was 94%. The author only used one dataset, which was a drawback, because it did not reflect the most recent developments in computer vision topology. The authors could have used an additional strong pretrained model to enable a thorough assessment of the effectiveness of their approach. Basri et al. [16] worked on the NumtaDB dataset using various pretrained models including AlexNet, MobileNet, GoogleLeNet, and CapsuleNet. In their suggested strategy, MobileNet had the lowest accuracy of 83.58%, while AlexNet had the highest accuracy of 99.01%.

In this work, the author used four state-of-the-art deep CNN model architecture models to attain good results for both the normal and augmented datasets, but no prior research was compared. They stated that training the model took more time and required a powerful GPU, indicating accessibility and scalability issues for researchers with minimal resources. Basu et al. [17] used the Dempster–Shafer (DS) technique to apply an individual MLP classifier and subsequently a combination of MLP classifiers. On the ISI dataset, the average recognition rate achieved 95.1%, which was increased compared to the individual MLP classifiers followed by threefold cross validation. Shopon et al. [18] used the two most popular datasets, the CMATERDB and ISI numeral datasets, for their experiment. Their proposed approach to classifying the Bengali handwritten digits used an unsupervised pretraining autoencoder with a convolutional neural network. They operated four types of experiments keeping the ISI dataset as a training set and the CMATERDB as a testing set, finally reaching an accuracy of 99.50% with the ACMA. Nasir et al. [19] attempted to solve issues with handwritten numerical recognition, such as distortions, various writing styles, thickness differences, scales, rotation, noise, occlusion, and missing sections. Support Vector Machine (SVM) was then used to recognize the retrieved features. They collected four different handwritten digits from 75 people, and the total number of images was 300. The recognition rate of their proposed method using SVM was 96.80%. They used Bengali datasets for recognition in the abovementioned work, which are not as plentiful or publicly available as English digits in terms of the quantity and quality, for example, the MNIST dataset [12].

In the context of Bengali handwritten digit recognition, large and diversified datasets are scarce. Prior research has mainly relied on two well-known databases, the NumtaDB and the Indian Statistical Institute (ISI). While they constitute a sizeable portion of the resources that are accessible, not all of the datasets in the database are available to the public, and of those that are, many of them require labels. It is important to note that more data would be beneficial for the Bengali digit recognition area because machine learning models often produce more accurate results when trained on larger datasets. We have developed the PUST DB dataset to better analyze and recognize Bengali digits in order to satisfy this need. We intend to improve the diversity of the data used for Bengali digit recognition tasks by including the datasets in our study. In order to explore and boost the state-of-the-art in Bengali digit recognition algorithms and methodologies, this dataset offers a valuable supplement to the resources already available.

3. Materials and Methods

3.1. Dataset Description

We introduce a dataset in this study known as the PUST DB dataset, which contains 5440 handwritten digit images in Bengali. From August 2021 to October 2021, this dataset was collected at Pabna University of Science and Technology (PUST). PUST is a renowned institution in Bangladesh; so a large number of students from different regions in Bangladesh study there. Hence, we can argue that this dataset represents the majority of the regions, because students from all regions participated in our study. The images were acquired using a digital device called the VEIKK A15 Graphics Drawing Tablet (10 × 6 Inch Digital Drawing Tablet) with an 8192 Levels Battery and 12 Hot Keys that supports Win/Mac/Linux/Android OS, Graphics Tablet for Painting, and Online Teaching. Every participant wrote the 0–9 Bengali digits, and each image contained 120 × 120 pixels. The images were taken using a digital device in PNG (Portable Network Graphics) format. This format is superior to others in terms of reducing the size of any image without losing any detailed information, allowing for transparency, and being ideal for editing [20]. Each image pixel has a single integer value (0 to 255) that represents the brightness of the pixel.

In addition, our study included another dataset collected from the Kaggle repository, a sizable collection of handwritten digit images. The 15,620 images in this dataset considerably increased the amount of data that was accessible for our investigation. Notably, this dataset accommodated numerous image formats frequently seen in real-world applications by incorporating a variety of file types, including png, jpeg, and jpg. The images in this dataset also showed a wide variety of pixel dimensions, which accurately captured the variability that may be found when working with hand-drawn digits in practical systems. With the help of this dataset, we were able to increase the robustness and generalizability of our research findings by ensuring thorough and accurate coverage of handwritten digit variants. Figure 1 displays the frequency of our PUST and Kaggle Bengali handwritten digits dataset. The sample of the two Bengali digit dataset images is shown in Figure 2.

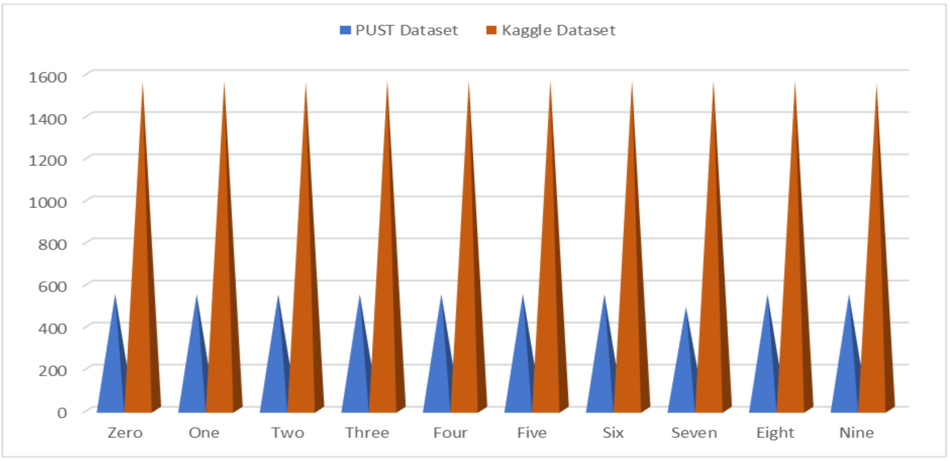

Figure 1. Frequency of the two Bengali hand-digit datasets.

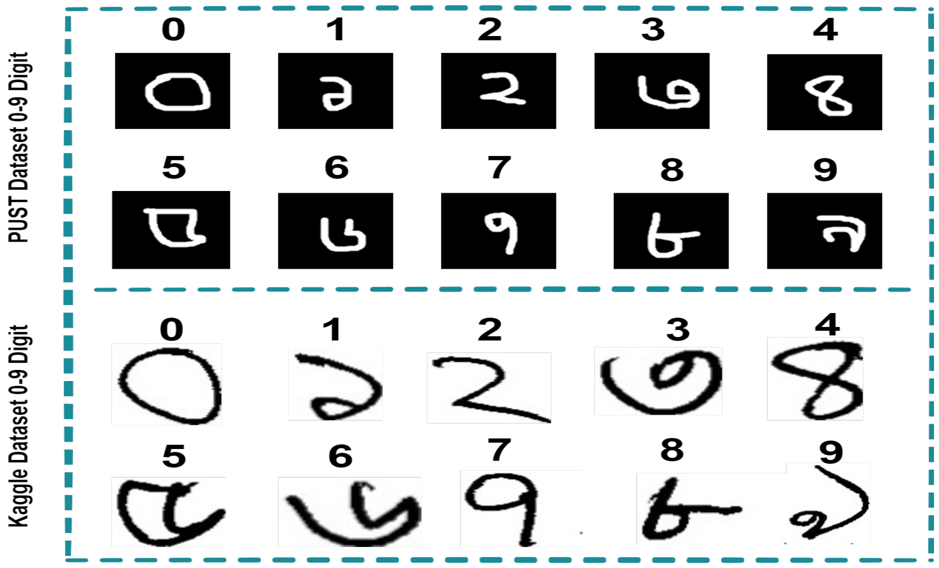

Figure 2. PUST and Kaggle Bengali dataset 0 to 9 digits and their corresponding English digits.

3.2. Feature Extractor and Learning Algorithm

A feature extractor is an algorithm used in machine learning (ML) and deep learning (DL) that automatically extracts key features from input data. These algorithms take the features that are important and change them into something else that may be used to advance the ML and DL algorithms and provide the desired outcomes [21]. Particularly in the processing of high dimensional data, images, audio, texts, and natural language, among other applications, feature extractors are important. In machine learning, manually created feature extractors like the LBP, CLBP, SIFT, etc., are frequently used to extract features, which are subsequently fed into the ML algorithm. Among the well-known learning algorithms used in ML are Logistic Regression (LR), Random Forest (RF), Decision Tree (DT), Extreme Gradient Boosting (XGBoost) [22], and others. On the other hand, DL uses a neural network, polling, and recurrent layers to automatically extract features from the input raw data. ML and DL are used to detect fraud activity, disease prediction, image classification, pattern recognition, spam filtering, and many other things [23]. DL algorithms function similarly to the neural network architecture of the human brain. There are many different kinds of models, including convolution neural networks (CNNs), deep Boltzmann machine (DBM), deep conventional extreme machine learning (DC-ELM), artificial neural networks (ANNs), recurrent neural networks (RNNs), and more [24,25].

3.3. Handcrafted Feature Extractors

Handcrafted features are employed in many different contexts, including computer vision, image processing, and other fields. There are many handcrafted feature extractors, including the Histogram of Oriented Gradients (HOG), Scale-Invariant Feature Transform (SHIFT), Speed-Up Robust Features (SURF), Local Binary Pattern (LBP), and Complete Local Binary Pattern (CLBP), among others. Although the handcrafted feature extractor has many advantages, the main limitation is the need for domain knowledge and the expert design of the feature engineering. The LBP, CLBP, and HOG feature extractors were employed in the analysis of the image data we proposed.

3.3.1. Local Binary Pattern (LBP)

The LBP is a simple but effective feature extraction operator that resembles a hand-crafted method. Due to its straightforward computation, it has gained a lot of popularity. The LBP operation's fundamental algorithm entails thresholding a 3 × 3 neighborhood around each pixel to determine the binary value and give labels to each pixel's value in an image [26]. Consider that there are P sampling points on a circle of radius R. Based on the circle's center position pixel value, the operation technique assigns a neighborhood value of 1 when P neighbors' pixel intensities are more than or equal to the center value and a value of 0 otherwise. Multiplying the threshold value by the binomial weights yields the LBP code. Consequently, the LBP operator's equation is as follows:

$$LBP_{(x_a,y_a)} = \sum_{n=0}^{n-1} \zeta_n(\tau_n - \tau_c)2^n; \quad \zeta_n = \begin{cases} 1, & \text{if } (\tau_n - \tau_c) \geq 0 \\ 0, & \text{if } (\tau_n - \tau_c) < 0 \end{cases}, \tag{1}$$

where τ_c is the gray value of the central pixel, τ_n is the value of its neighbors, ζ_n indicates the Kronecker function, and 2^n is the binomial factor [27].

3.3.2. Complete Local Binary Pattern (CLBP)

An updated version of the LBP operator is the CLBP feature extraction operator. Alongside the traditional LBP, the CLBP extracts feature from images by taking into account P circularly sampling points in a circle with an R radius. Because it uses magnitude ∂_n, sign (S), and the center pixel, in addition to pixel sampling as LBP, the CLBP is a complete feature extractor modeling of the LBP. The distinction is that each pixel's sign and magnitude were used by the CLBP to encode the binary code. Suppose we have a difference between τ_n neighbors and τ_c central pixel values. We can define the local differences between them by $d_n = (\tau_n - \tau_c)$. Then, the decomposition can be written as

$$d_n = S_n \times \partial_n; \quad \{S_n = \text{sign}(d_n),\ \partial_n = |d_n|\}, \tag{2}$$

where the term S_n indicates the signs of d_n, and ∂_n indicates the magnitude of d_n. The above d_n term is called the local difference sign-magnitude transforms (LDSMT) [28].

3.3.3. Histogram of Oriented Gradients (HOG)

A common feature extraction technique used in computer vision and image processing operations is the histogram of oriented gradients (HOG). Navneet Dalal and Bill Triggs first proposed it in their 2005 publication titled "Histograms of Oriented Gradients for Human Detection". By constructing histograms of directed gradients, the HOG approach collects information about an image's local shape and texture. It is based on the hunch that the distribution of local gradient orientations might describe the forms and structures of objects.

The HOG algorithm performs the following actions during the feature extraction process: preparing the input image by turning it into grayscale and, if necessary, applying contrast normalization, utilizing gradient operators, calculating the image's gradients (magnitude and orientation), and splitting the image into tiny cells (e.g., 10 × 10 pixels) to record regional data. Each cell's gradient orientations are calculated as a histogram, and then normalization strategies are used to aggregate neighboring cells into larger blocks (e.g., L2 normalization). The resulting feature vector, which represents the image or local features, is created by joining the normalized block histograms [29].

Overall, in the case of the handcrafted feature extractors, the researcher is required to have a domain-specific understanding of the facts to extract the best relevant features that are difficult to extract using other techniques [30]. Since a deep learning model extracts features automatically, the researcher need not address them manually. Therefore, the CNN model can be used to extract features more quickly [31].

3.4. Convolution Neural Network (CNN)

Convolutional Neural Network (CNN) architecture has gained popularity over the past several years among both academic researchers and a variety of industries. The CNN models have made a considerable contribution to image recognition, computer vision, and pattern recognition, which was unthinkable a few decades ago. The CNN was specifically created for image analysis applications based on computer vision. The input picture pixel is taken into consideration as the input layer in the CNN architecture. The output layer, which is where the model eventually classifies the target item, is made up of numerous hidden layers from which it learns. The features are automatically extracted during the hidden layer phases rather than using the handcrafted feature extraction methods that are typically used. This feature extraction task is designed and built by many internal operations, including convolution, kernel filter mapping, and polling (max, average, and global). The input layer, the hidden layer, which is made up of convolutional layers, ReLUs (rectified linear units), pooling layers, fully connected flattening layers, and ultimately the output layer are the fundamental units of the CNN model. The CNN model's three components are referred to as its building blocks. Using the kernel filter mapping during the process phases, the convolution layer convoluted several images from the grayscale image using the ReLU function [32].

The kernel applies a different weight to image pixels with and without strides. So, the mathematical form of the convolution can be written as: $y[i][j] = \sum_{k=1}^{m} \sum_{l=1}^{n} x[i+k][j+l] \cdot kernel[k][l])$, where the output $y[i][j]$ is the output feature map at the ith row and the jth column, the input feature $x[i+k][j+l]$ is the input feature map at the $(i+k)$th row and the $(j+l)$th column, and the kernel $[k][l]$ is the kernel value size of the kth row and the lth column. The following formula may be used to calculate the convolution layer output size: $\frac{W-F+2P}{S+1}$ where W refers to the input image matrix, F is the kernel filter mapping size, P is the padding, and S refers to the stride. Following these in-depth tasks in the feature map stage, the features are flattened for use in fully connected neural networks to classify the images. Figure 3 depicts the entire CNN procedure in detail.

3.5. Pretrained CNN Features

A kind of feature neural network model that has been trained on a massive amount of image data is called the pretrained CNN. The weights that the CNN model learns are referred to as the pretrained features. Following several learning algorithms, this weight is utilized as a feature extractor [33]. We can save a lot of time and effort by using these pretrained models. These previously trained features are now valuable for a variety of computer vision tasks, especially when there is a dearth of labeled data. Utilizing these pretrained models often recognized as transfer learning, led to the creation of the new scheme [34].

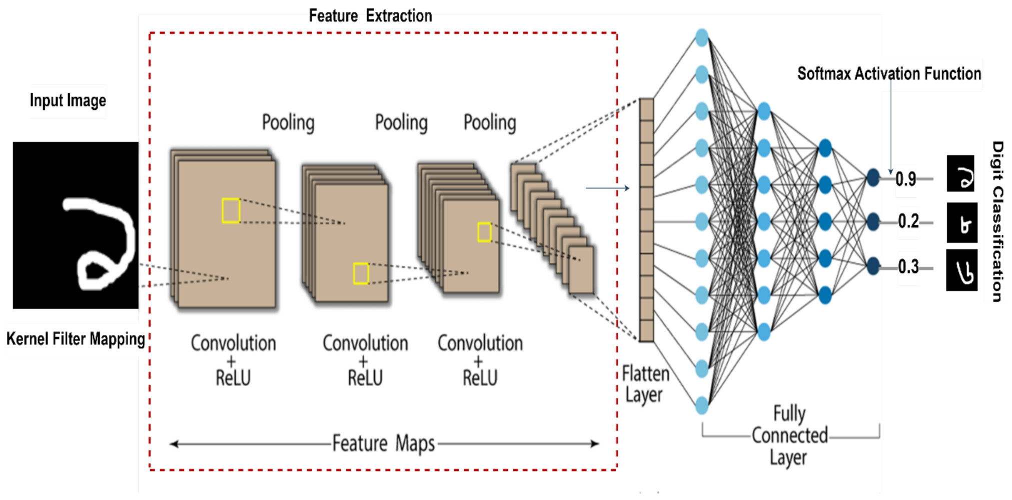

Figure 3. Architecture of the convolution neural network model.

4. Bengali Handwritten Digit Recognition Pipeline

To build a robust model that can accurately recognize the digits, the handwritten digit collection and processing approach is a vital stage. This strategy aimed to collect a wide group of sample handwritten digit datasets that were utilized for developing and testing the recognition model. In the subsequent section, we introduce our proposed Bengali handwritten digit collection strategy and image preprocessing.

4.1. Data Collection Approach

Using Python programming, we designed an automated device-based system that automatically gathered the respondent's handpicked digit. In this system, once a participator wrote a digit on the VEIKK A15 graphics drawing tablet screen that displayed the paint app plate, a screenshot of the computer screen was captured in less than eight seconds for each digit. Then, the captured images were stored the in the selected folders named 0, 1, 2, 3, 4, 5, 6, 7, 8, and 9. All the digit data folders were imported into a Google Drive collab notebook folder. Finally, we read the images from this directory for analysis. The detailed data collection system infrastructure of the PUST Bengali handwritten digit is illustrated in Figure 4.

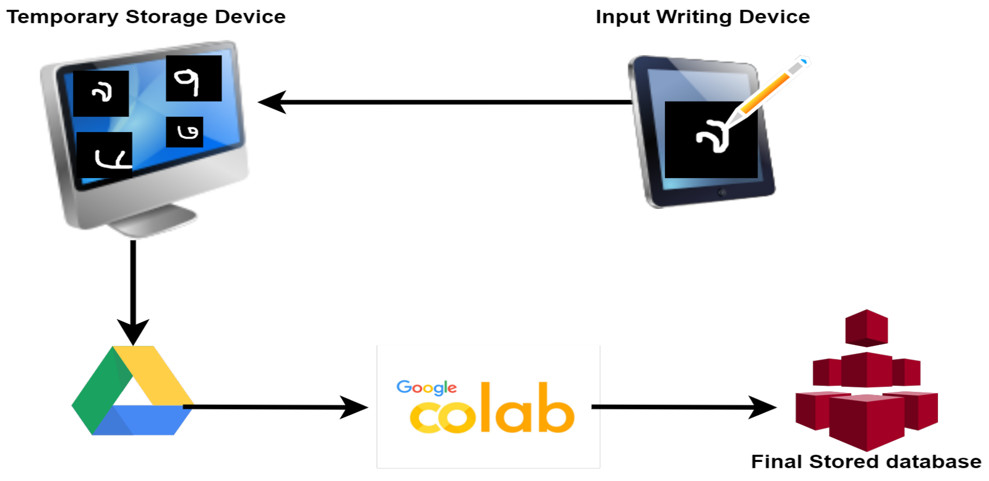

Figure 4. Our novel Bengali handwritten digit data collection system infrastructure.

4.2. Image Preprocessing

An essential function in image applications is image processing. When there is unclear, noisy, or undesirable data present that causes the real-life application to be incorrectly classified, we can correctly identify our desired image by applying a variety of image processing techniques [35]. There are many real-life applications in image processing, such as in the medical field, remote sensing, image sharping and restoration, face detection, etc. Image processing raises the quality of the image data that enhance image recognition [36]. We used the Bengali digit, as we previously described, to classify it. These images were not instantaneously fed into the algorithm, though. The dataset was worthless when it was first gathered, and all of the image pixels were 120 × 120. We experimented with various dataset processing techniques to increase the proposed method's accuracy and effectiveness. To save computing cost and time while also boosting the uniformity we initially reduced all of the images to a size of 28 × 28 pixels. In general, higher pixel value images typically result in a deep learning model having more parameters, storing the model in more memory, and sometimes overfitting, especially for specific deep learning architectures like CNNs [37]. Then we transformed the RGB (Red, Green, and Blue) format of the full set of images to grayscale. Although converting RGB images to grayscale may substantially reduce a model's computational complexity and increase speed processing, it might not immediately address all kinds of image noise. To efficiently reduce different kinds of noise and artifacts, substantial research on image filters remains necessary. The details are depicted in Figure 5.

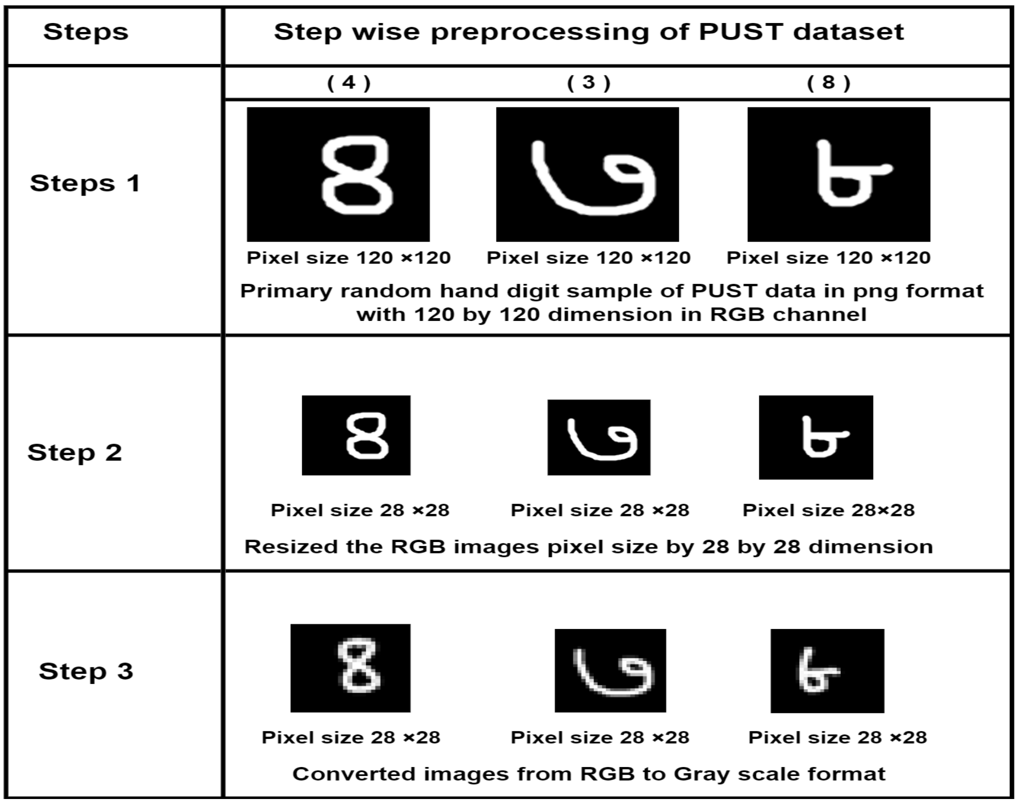
Steps
Step wise preprocessing of PUST dataset
(4)
(3)
(8)
Steps 1
Pixel size 120 ×120
Pixel size 120 ×120
Pixel size 120 ×120
Primary random hand digit sample of PUST data in png format with 120 by 120 dimension in RGB channel
Step 2
Pixel size 28 ×28
Pixel size 28 ×28
Pixel size 28×28
Resized the RGB images pixel size by 28 by 28 dimension
Step 3
Pixel size 28 ×28
Pixel size 28 ×28
Pixel size 28 ×28
Converted images from RGB to Gray scale format

Figure 5. Step-wise image preprocessing workflow.

5. Research Methodology

In this study, we attempted two approaches: a machine learning approach applied on handcrafted features and deep learning models. This section is separated into two parts. In the first, we explore the handcrafted feature descriptors, and in the second, we describe the construction of a customized deep learning model, namely the CNN model. In the subsection that follows, we further detail the two-way Bengali digit recognition system.

5.1. First Proposed Approach

Handcrafted Model Details

In this study, we employed three handcrafted feature extractors, including the HOG, LBP, and CLBP. In Section 3.1, the specifics of these extractors are covered. Employing a handcrafted feature extraction approach, we applied principle component analysis (PCA) with a 95% explanation of variation on this dataset after using the suggested extractor. We retrieved 1744, 2, and 789 features after applying the up to 95% cumulative variation contribution ratio criterion to the LBP, CLBP, and HOG extracted features, respectively. After that, we integrated the extracted features to produce the optimal result. The proposed details are shown in Figure 6.

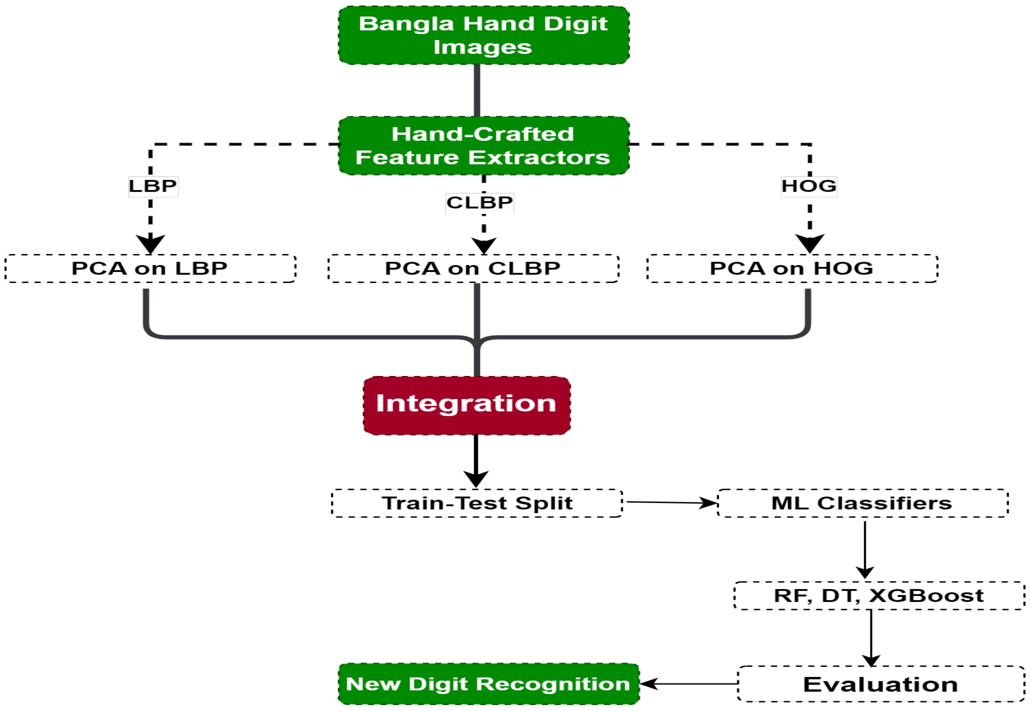

Figure 6. Proposed Bengali handwritten digit classification using handcrafted feature extractors.

5.2. Second Proposed Approach

5.2.1. CNN Model Details

We discussed the reasons why a CNN produces promising outcomes for the classification of image data in Sections 3.2 and 3.3. We used the fine-tuned hyperparameters approach to set various hyperparameters in our CNN model architecture for the three CNN models and their stacking model. The proposed model architecture's workflow is shown in Figure 7. The set of model-building hyperparameters' tuning is discussed below. The proposed overview for building a deep learning model is provided in Table 1.

Conv2D: The conv2D is a two-dimensional convolution neural network design that is used to convolute the input features with a set of the linear filters, such as kernel or weight, to produce a set of output features followed by a pooling layer and some activation functions [38]. We applied 32 kernel filters of sizes (5 × 5) for the first model and (3 × 3) for the second two models with the same padding to each of the images that comprised stacks of 32 feature maps using the ReLU activation function $f(x) = max(0, x)$, where x indicates the pixel value of the images.

Maxpolling2D: Maxpolling2D is a layer within Conv2D that aids in sampling the feature map and reducing the input feature's spatial dimension. The (2 × 2) size is used for all three of the CNN models, with the maximum value during each polling window being chosen for the new feature map.

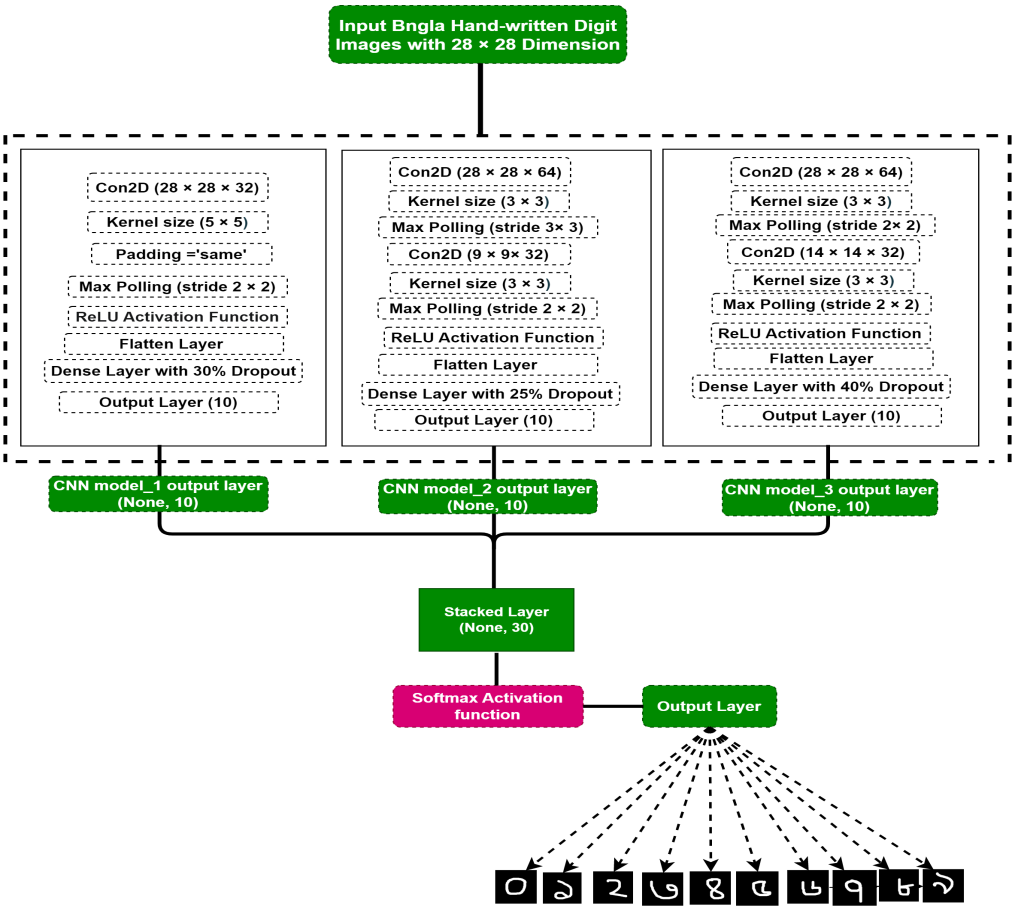

Figure 7. The proposed Bengali handwritten digit recognition workflow diagram using the CNN model.

Hidden Layer and Dropout Regularization: We set a different number of hidden layers for each of the three CNN models after flattening the images to produce a more accurate and effective result. We monitored closely to set the hidden layer because a model can experience underfitting when the number of hidden layers is low. However, if there are more hidden layers, the model may experience overfitting. To prevent overfitting, we added a hidden layer and set several dropout regulations, such as 0.25, 0.30, and 0.40, which randomly dropped the assigned layer throughout the training phase. Due to this, the output layer was protected from the dominant neuron. Finally, to classify our desired digit, we set the softmax activation function $\hat{y} = \frac{f^{(z_i)}}{\sum_{j=1}^{10} f^{(z_i)}}$ as an output layer [12].

Optimizer: In this research, we employed the Adam (Adaptive Moment Estimation) optimization algorithm, which automatically updates the model parameter based on computing the exponential moving average of the gradient and squared gradient [39].

Early Stopping: A key strategy for preventing overfitting during the training period is early halting. It evaluates a chosen metric, often the validation score, to keep track of the model's performance, and it suspends the training process when the metric does not improve after a specified number of iterations [40].

Epoch: The number of times the full training dataset passes through the model is controlled by the epoch. We set 150 epochs to lower the training error rate and move closer to the validation score. It took around 3 s~40 s in the environment described in Sections 7 and 8 to run the entire training dataset through the model for each epoch [12].

Table 1. Summary of the proposed customized CNN models' architecture.

Model Name	Layers	Shape	Parameters	Total Parameters
CNN model one	Conv2D	(28 × 28 × 32)	832	779,934
	Kernel size	(5 × 5)	0	
	MaxPooling2D	(14 × 14 × 32)	0	
	Flatten	(None, 6272)	0	
	Dense layer	124	777,852	
	Dropout	0.30	0	
	Dense	10	1250	
CNN model two	Conv2D	(28 × 28 × 64)	640	86,058
	Kernel size	(3 × 3)	0	
	MaxPooling2D	(9 × 9 × 64)	0	
	Conv2D	(9 × 9 × 32)	18,464	
	Kernel size	(3 × 3)	0	
	MaxPooling2D	(4 × 4 × 32)	0	
	Flatten	(None, 512)	0	
	Dense	128	65,664	
	Dropout	0.25	0	
	Dense	10	1290	
CNN model three	Conv2D	(28 × 28 × 64)	640	423,338
	Kernel size	(3 × 3)	0	
	MaxPooling2D	(14 × 14 × 64)	0	
	Conv2D	(14 × 14 × 32)	18,466	
	Kernel size	(3 × 3)	0	
	MaxPooling2D	(7 × 7 × 32)	0	
	Flatten	(None, 1568)	0	
	Dense	256	401,664	
	Dropout	0.4	0	
	Dense	10	2570	
Stacked model	CNN model_1	(None, 10)	7,799,934	1,425,050
	CNN model_2	(None, 10)	86,058	
	CNN model_3	(None, 10)	423,338	
	Stacked layer	(None, 30)	0	
	Dense	256	12,400	
	Dropout	0.4	0	
	Dense	150	120,300	
	Dropout	0.3	0	
	Dense	10	3010	

5.2.2. Stacked Convolution Neural Network

The stacked CNN is a method that combines distinct CNN models to achieve the optimum performance. The fundamental concept is to feed a stacked model with the output of one CNN model [41]. Each proposed unique CNN model gathers information from the features of the images, and by stacking these models, the model can learn more intricate and abstract features than from just one particular CNN model. The CNN architecture is trained up with the various image features, such as one model might be trained to recognize the edge, while another is trained to recognize texture. The stacked model may learn a complicated and varied set of image features by training a diverse CNN model with a variety of distinct architectures and hyperparameters. In the end, the image is processed by the CNNs in the stack, and their outputs are often concatenated or combined in some way before being input into a fully connected layer for the final classification or regression task. The unique way that the CNN stacking algorithm is used will determine the exact formula for combining the CNN results. Concatenation, addition, averaging, or max pooling are some typical techniques for merging the outputs [42].

6. Experimental Setup and System Specification

We used several handcrafted image feature extractors employing the Random Forest (RF), XGBoost, and Support Vector Machine (SVM) classifiers as well as various convolution neural network algorithms to recognize the digit data correctly and efficiently. We split our dataset into 70% for the training phase and 30% for the testing phase before using it to feed the model. The validation data parameter was used to evaluate the performance of the model during training. In order to avoid overfitting, it is crucial to assess the model's performance using hypothetical data. The validation data parameter comes into play at this point. At the end of each epoch during training, the model calculated the accuracy and loss using both the training data and validation data. We may track the model's performance during training and identify occurrences of overfitting by specifying the validation data in the validation data parameter. By doing so, the model's generalization performance is enhanced, and its ability to perform effectively on new unseen data is ensured. The process iterated for each cross-validation fold. For each fold, the data were split into training and validation sets, and a new instance of the CNN was created. Details of the system specification used for this experiment are stated in Table 2.

Table 2. Usable system specification for this study.

Resources	Usable System Specification
CPU	Intel^R Core(TM) i3 11 Gen-7100U CPU @ 2.40 GHz
System Type	64-bit operating system, x64-based processor
Environment	Python 3 Google Colab
RAM	12.68 GB
Disk	1 TB Hard disk & 128 GB SSD
Packages	Keras with TensorFlow, Sci-kit learn, Matplotlib, Pandas, Numpy

7. Experimental Results and Discussion

The Bengali handwritten digit's incredibly complex structure makes it more difficult to recognize. We used both conventional machine learning models and convolutional neural network (CNN) models. We were able to create a robust model in this research that accurately recognized the Bengali numeral digits. In this research, the accuracy, precision, recall, F1 score, and ROC AUC were taken into account when evaluating the recognition performance of these models. Using data from the Bengali handwritten digits, Table 3 demonstrates how well the ML classifier performed using integrated handcrafted features. Table 4 illustrates the performance of the CNN models on our proposed Bengali handwritten digits. Table 5 depicts the training, validation, and testing accuracy of the CNN models. Finally, we present in a table the ranking of the CNN models' performance based on their accuracy. The XGBoost classifier performed quite well on our own dataset, in contrast to the other two classifiers, as seen in Table 3. The best accuracy, precision, recall, F1 score, and ROC AUC were attained by the XGBoost classifier with scores of 85.29%, 85.11%, 85.18%, 85.08%, and 98.67%, respectively. The other two classifiers exhibited approximately equivalent performance. Figure 8 visualizes the recognition results of the ML classifiers on the handcrafted features for the PUST dataset.

Table 3. Recognition performance of the ML classifier on the handcrafted features for the PUST dataset.

Classifier	Accuracy (%)	Precision (%)	Recall (%)	F1 Score (%)	ROC AUC Score (%)
SVM	58.08	58.61	58.05	58.91	87.86
XGBoost	85.29	85.11	85.18	85.08	98.67
RF	74.39	75.33	74.38	74.43	95.27

Table 4. Recognition performance of the customized CNN models for the PUST and Kaggle datasets.

Data Source	Model Name	Accuracy (%)	Precision (%)	Recall (%)	F1 Score (%)	ROC AUC Score (%)
PUST	CNN model_1	93.87	93.91	93.87	93.86	99.61
	CNN model_2	97.12	97.15	97.12	97.12	99.94
	CNN model_3	97.43	97.46	97.43	97.43	99.91
	Stack model	97.06	97.09	97.06	97.06	99.92
Kaggle	CNN model_1	95.35	95.37	95.35	95.35	99.88
	CNN model_2	97.19	97.21	97.19	97.19	99.92
	CNN model_3	97.22	97.21	97.19	97.19	99.92
	Stack model	96.14	96.18	96.14	96.14	99.90

Table 5. Training, validation, and testing accuracy of the customized CNN models for the PUST and Kaggle datasets.

Data Source	Model Name	Training Accuracy (%)	Validation Accuracy (%)	Testing Accuracy (%)	Epoch
PUST	CNN model_1	99.19	93.87	93.87	29/150
	CNN model_2	99.53	97.12	97.12	48/150
	CNN model_3	99.66	97.43	97.43	32/150
	Stack model	99.37	97.58	97.57	32/150
Kaggle	CNN model_1	99.07	95.35	93.35	26/150
	CNN model_2	99.12	97.19	97.18	32/150
	CNN model_3	99.24	97.19	99.18	23/150
	Stack model	99.26	96.14	96.14	15/150

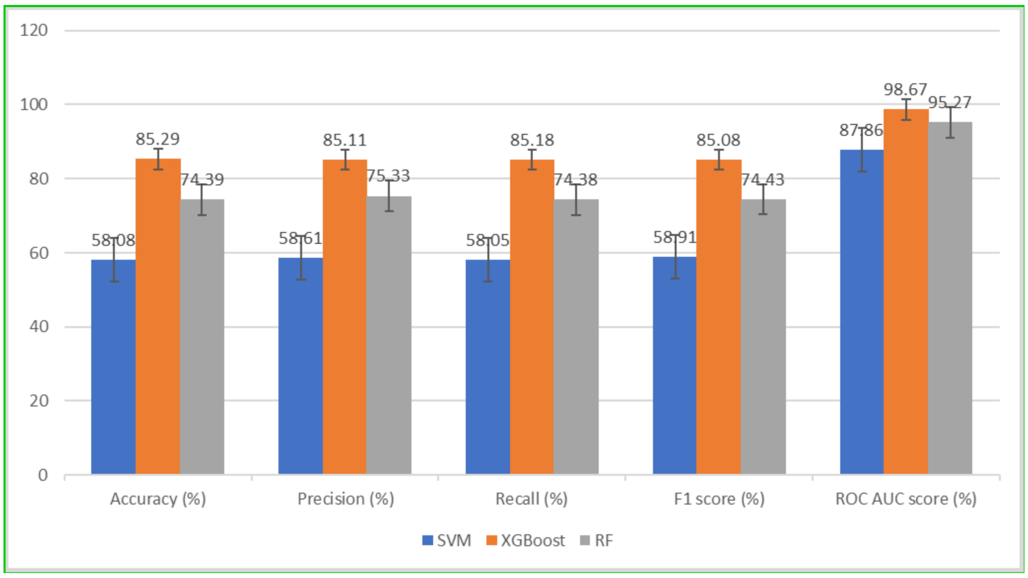

Figure 8. ML classifier results bar diagram on the handcrafted features for the PUST dataset.

The proposed customized three CNN models for the two datasets and their stacked results are shown in Table 4. Our fully customized CNN models generated impressive results on the novel dataset; in particular, CNN model three stood out for its boosting performance, achieving accuracy rates of 97.43%, a precision of 97.46%, a recall of 97.43%, an F1 score of 97.43%, and an ROC AUC of 99.91%. We also included another dataset of Bengali handwritten digits for our models to use in order to assess the strengths of our model. CNN model three attained the best recognition performance over the other models. This model achieved an accuracy rate of 97.22%, a precision of 97.21%, a recall of 97.19%, an F1 score of 97.19%, and an ROC AUC score of 99.92. In terms of these performances for the two datasets, the proposed CNN model three produced superior results. Figure 9 depicts the proposed CNN model recognition performance for the both the PUST and Kaggle datasets.

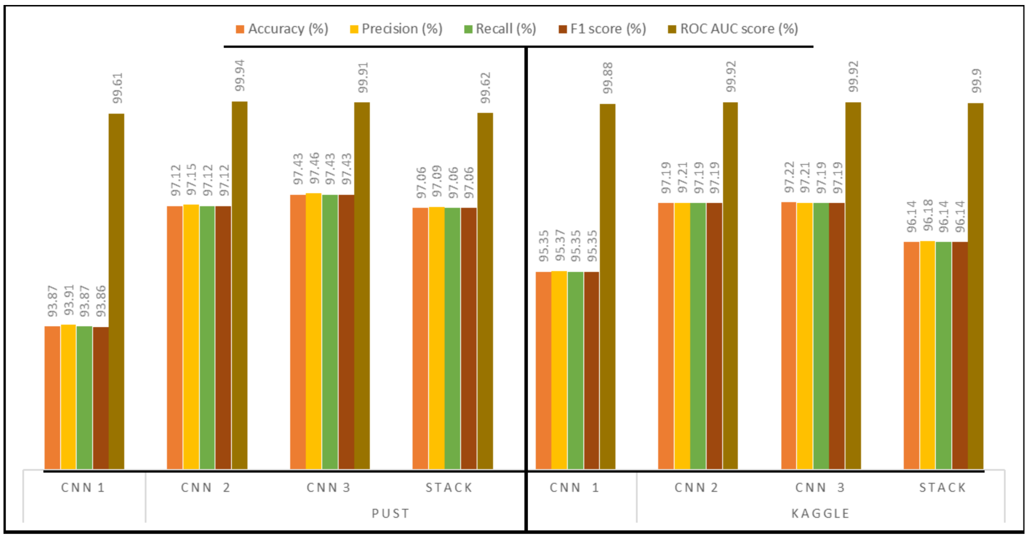

Figure 9. CNN model's recognition performance results using multiple bar diagrams for the PUST and Kaggle datasets.

Table 5 also displays the training, validation, and testing accuracy of our proposed datasets. We configured the validation to minimize overfitting during the training phase. For the PUST dataset, CNN model three also attained the highest accuracy in this domain. At 32 out of a total of 150 epochs, this model provided the highest training accuracy score of 99.66%, a validation score of 97.43%, and a testing accuracy score of 97.43%. As opposed to this, the stacked model performed best on the Kaggle dataset, with a training accuracy of 99.26%, a testing accuracy of 96.14%, and a validation accuracy of 96.14%. In 150 epochs, the training phase required each model 3 s~1.5 m to complete on average. When the model experienced value loss, the iteration process was interrupted to define early stopping criteria for five patients. Table 6 provides a final overview of the summary of the model's accuracy performance rating. Surprisingly, CNN Model three's recognition rate outperformed that of all other models for both of the proposed datasets, as shown in the table.

Table 6. Recognition performance comparison among the proposed customized CNN models.

PUST			Kaggle		
Ranking	**Model Name**	**Accuracy (%)**	**Ranking**	**Model Name**	**Accuracy (%)**
First	CNN model_3	97.43	First	CNN model_3	97.22
Second	CNN model_2	97.12	Second	CNN model_2	97.19
Third	Stack model	97.06	Third	Stack model	96.14
Fourth	CNN model_1	93.87	Fourth	CNN model_1	95.35

A graphical representation of the proposed customized CNN models' accuracy and loss curves are shown in Figure 10. Notably, despite the fact that all the models functioned admirably, CNN Model three outperformed the others in terms of the classification performance. Though they did not outperform Model three's performance, it is important to note that the remaining models also produced remarkable results. When the ML algorithms and CNN models were compared for classification performance on our proposed Bengali handwritten digit image dataset, it became clear that the CNN model consistently surpassed the ML technique, especially when utilizing handcrafted feature extraction techniques. Although the overall accuracy of our proposed stacked model shown in Figure 11 may not have been the highest, its findings showed robustness. According to this robustness, the stacked model was less likely to overfit or be affected by data noise. Therefore, although not having attained the highest accuracy, the stacked model can still be regarded as a stable and reliable model to ensure consistency.

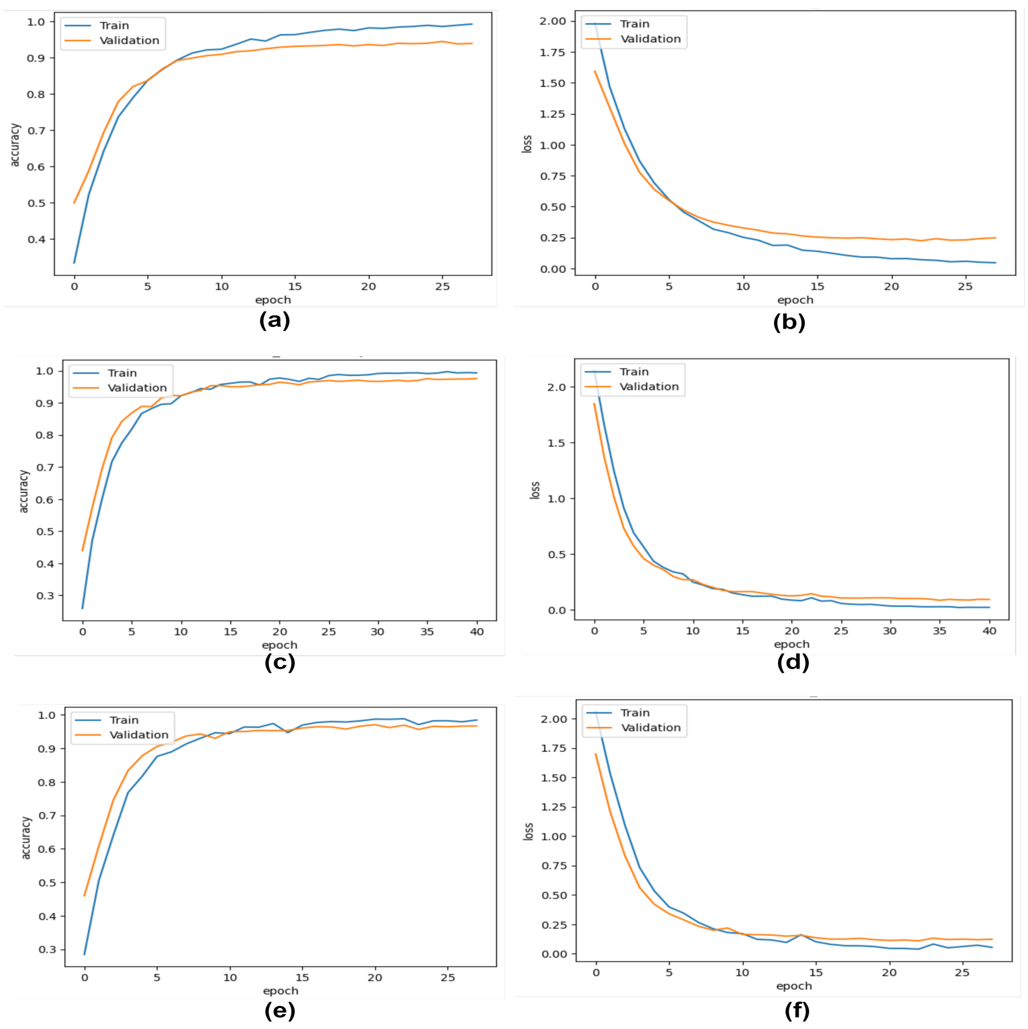

Figure 10. Proposed customized convolution neural network model and loss accuracy curve: (**a**,**b**) CNN model one accuracy and loss curve, (**c**,**d**) CNN model two accuracy and loss curve, (**e**,**f**) CNN model three accuracy and loss curve.

7.1. State-of-the-Art Action

In this study, we gave a thorough assessment of our proposed CNN approach for Bengali handwritten digit recognition, contrasting it with other earlier research studies that used several datasets for Bengali handwritten digit recognition. We utilized several freely accessible datasets that are often used in the literature for benchmarking. We evaluated the digit recognition performance of our proposed CNN model compared to other models that performed better than other Bengali digit recognition. We took into account a number of evaluation criteria during the evaluation, including the training accuracy, the testing accuracy, the validation accuracy, and the overall accuracy. To ensure a thorough and effective recognition of the digit data, the precision, F1 score, recall, and ROC AUC calculations were also performed. Table 7 discusses in detail the several Bengali handwritten digit data recognition performance and our benchmark data set using the proposed approach. In the context of the dataset for Bengali handwritten digits, our proposed CNN models outperformed the previous different studies. The customized CNN model three achieved a training accuracy of 99.66% and a testing accuracy of 97.43% across the entire assessment. Remarkably, our stacked model that used another Bengali dataset showed even better accuracy, attaining a training accuracy of 99.26% and a testing accuracy of 99.14%. These findings unequivocally demonstrate the cutting-edge status of our suggested strategy across several datasets of Bengali handwritten digits. The development of Bengali handwritten digit recognition is furthered by the effectiveness and resilience of our models in handling variations in handwriting styles and noisy images. We believe this approach improves the validity of our findings and that our proposed CNN models make important contributions to improving the state-of-the-art in Bengali handwritten digit recognition.

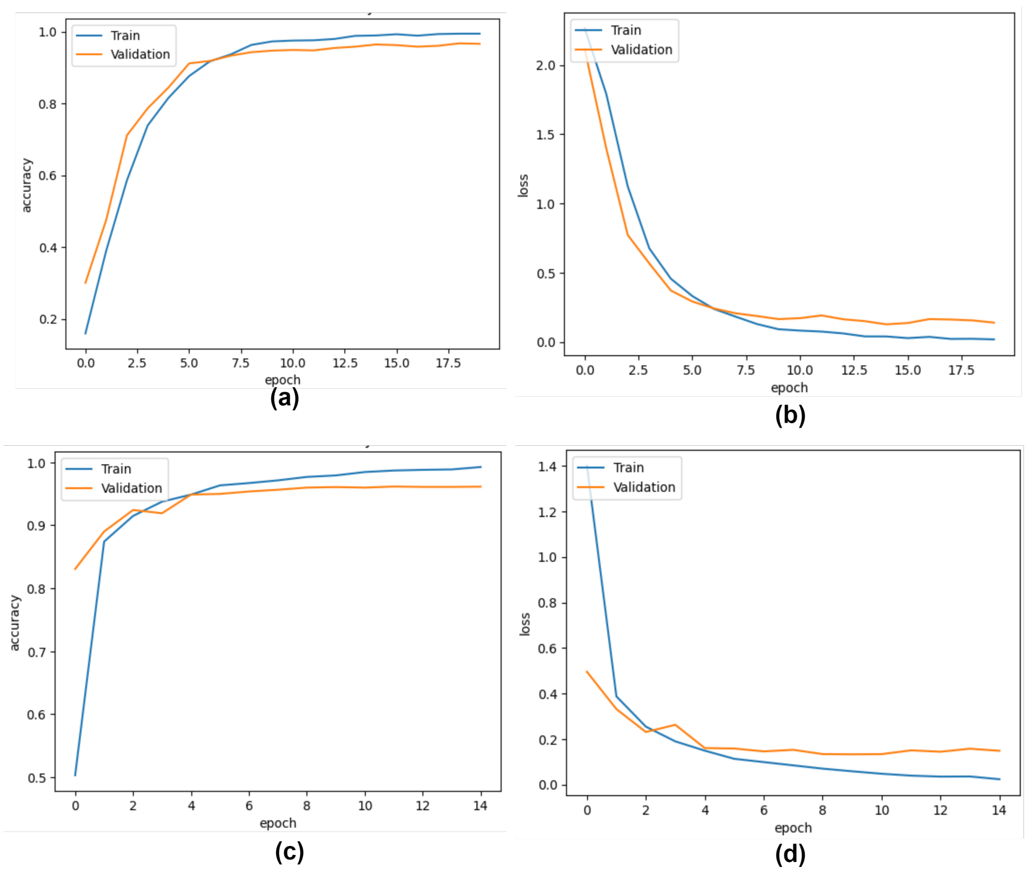

Figure 11. PUST and Kaggle datasets' stacked model accuracy and loss curve: (**a**,**b**) PUST dataset stacked model accuracy and loss curve, (**c**,**d**) Kaggle dataset stacked model accuracy and loss curve.

7.2. Strengths and Weaknesses

To assist researchers in exploration and analysis in the field of Bengali handwritten digit recognition, image processing, and computer vision, this paper introduces a new dataset. In contrast to the previous state-of-the-art works, this work developed the best CNN model architecture utilizing the fine-tuned hyperparameters approach, which is an indication of the proposed architecture's strength and effectiveness. In addition, the proposed framework was applied to another Bengali handwritten digit dataset with diverse writing styles, noise, and other dataset factors. Our proposed approach also achieved state-of-the-art performance, which indicates that our customized CNN model can handle other digit image handling without difficulty. Although this research developed a new dataset, the dataset was small in the case of the deep learning model and lacking in heterogeneity. The lack of implementation of many popular pretrained CNN models is another possible issue. However, this method can be significantly benchmarked and compared to the most recent studies that used pretrained CNN models. This may offer insightful information on our proposed plan of action.

Table 7. Recognition performance of several Bengali handwritten digit datasets and our proposed dataset using the proposed approach.

Authors	Dataset	Digit Data	Algorithm	Highest Result
M. Adnan et al. [43]	CMATERdb	Bengali	VGG Net, ResNet, FractalNet, DenseNet, Nin, All-Conv	They compared the results of all algorithms based on the testing results. The DenseNet had the highest accuracy of 99.13%.
Basu et al. [17]	CVPR unit, ISI, Kolkata	Bengali	Multilayer perceptron one, Multilayer perceptron two, Dempster–Shafer (DS)	On average, the DS algorithm achieved the highest recognition rate of 95.1%.
U. Pal et al. [44]	Own dataset	Bengali	Thinning- and normalization-free automatic recognition	The overall recognition rate was 91.98%.
Y. Wen et al. [45]	Supported by Bangladesh Post	Bengali	Original + SVM, PCA + SVM, KPCA+SVM, IRPCA, KPS, PCA+SVM+IRPCA+KPS	The average recognition rate attained by combining PCA+SVM+IRPCA+KPS was 95.05%.
T. Hassan et al. [46]	CMATERdb	Bengali	KNN, ANN, SVM	The highest accuracy of 96.7% was achieved by ANN.
O. Paul et al. [37]	NumtaDB	Bengali	CNN, CapsNet, KNN, KNN with PCA, SVM, SVM with PCA, LR, LR with PCA, DT	The CNN model achieved the highest accuracy of 91.30%.
Y. Wen et al. [47]	Dataset was acquired from live letters by the automatic letters sorting machine in the Dhaka mail processing center of Bangladesh Post Office	Bengali	Euclidean Distance, BP, SVM, RIPCA, KPS, BD, KBD-P, KBD-G	Among all classifiers, the KBD-G had the best accuracy of 96.91%.
S. Basu et al. [48]	CVPR unit, Indian Statistical Institute, Kolkata and CMATER, Jadavpur University	Bengali	MLP, SVM	The MLP and SVM algorithms on a variety of handcrafted feature extractor methods reached 97.15% accuracy.
C. Saha et al. [49]	BanglaLekha- Isolated dataset	Bengali	Convolution Neural Network	The recognition performance for the training, testing, and validation accuracy was 93.29%, 98.37%, and 96%, respectively.
U. Bhattacharya [50]	Devanagri numeral Bengali dataset	Bengali	Three-stage multilayer perceptron	The proposed system had the highest training accuracy of 99.26% and the highest testing accuracy of 98.01%.
Our proposed approach	PUST dataset	Bengali	Proposed CNN model and their stacked model	On this dataset, our proposed customized CNN model three gave the best training accuracy of 99.66%, testing accuracy of 97.43%, and validation accuracy of 97.43%.
(continued)	Kaggle dataset	Bengali	Proposed CNN model and their stacked model	Among the entire assessment, the stacked model provided the very impressive and robust training accuracy of 99.26%, testing accuracy of 96.14%, and validation accuracy of 96.14%.

8. Conclusions and Future Work

In this paper, the task associated with the dataset is the automated classification of these images, which would be valuable contributions to various fields, including Bengali historical document digitization, the banking sector, postal services, education, security system, etc. We applied several machine learning (ML) methods to handcrafted image features to boost the accuracy recognition rate of Bengali handwritten digits, and we developed CNN models that produced cutting-edge results. The implementation of fine-tuned hyperparameters was considered in order to develop the proposed CNN model. The first experiment involved three image descriptors and their integrated statistical features obtained from PCA. Among the three ML algorithms, XGBoost performed better in most of the indices of the evaluation matrices than the others. On our novel dataset, the proposed customized CNN model three produced state-of-the-art results, while the combined stacked model produced robust outcomes on the Kaggle dataset of the Bengali handwritten digits. As a matter of fact, by contributing a novel dataset and creating precise models, the proposed method also boosts machine learning and computing vision technologies in the Bengali-speaking world.

However, to address the concerns of the variations in handwriting styles and the noise in the images, it was shown that there is still room for improvement in terms of the model efficacy and transfer-learning-based feature extraction approaches. By implementing several data augmentation techniques, we can artificially expand the size and diversity of the datasets, which can aid in enhancing the generalizability of the models and improving their suitability for practical applications. When dealing with sensitive documents, we can look into privacy-preserving machine learning approaches to handle potential privacy and security concerns. Additionally, we can explore online learning approaches that allow the model to constantly update itself as new data become available. This adaptability is especially important in instances where the dataset evolves over time as a result of updates or additions.

Author Contributions: Conceptualization, R.A., M.S.R. and J.S.; methodology, R.A., M.S.R. and J.S.; investigation, R.A., M.S.R., Y.O., Y.T. and J.S.; data collection, R.A. and M.S.R.; data curation, R.A. and M.S.R.; writing—original draft preparation, R.A. and M.S.R.; writing—review and editing, R.A., M.S.R., Y.O., Y.T. and J.S.; visualization, R.A., Y.O., Y.T. and M.S.R.; supervision, M.S.R. and J.S.; funding acquisition, J.S. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the Competitive Research Fund of the University of Aizu, Japan.

Data Availability Statement: Our collected PUST Bengali handwritten digit dataset utilized in this study is available on GitHub at https://github.com/ruhul256/PUST-Database-A-New-Bangal-Hand-Digit-Recognition (accessed on 1 August 2023). To access additional datasets (Kaggle repository), visit https://www.kaggle.com/datasets/wchowdhu/bengali-digits (accessed on 18 July 2023).

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Jana, R.; Bhattacharyya, S.; Das, S. Handwritten digit recognition using convolutional neural networks. *Deep. Learn. Res. Appl.* **2020**, *4*, 51–68. [CrossRef]
2. Ivanov, A.S.; Nikolaev, K.G.; Novikov, A.S.; Yurchenko, S.O.; Novoselov, K.S.; Andreeva, D.V.; Skorb, E.V. Programmable soft-matter electronics. *J. Phys. Chem. Lett.* **2021**, *12*, 2017–2022. [CrossRef] [PubMed]
3. Vadyala, S.R.; Betgeri, S.N.; Matthews, J.C.; Matthews, E. A review of physics-based machine learning in civil engineering. *Results Eng.* **2022**, *13*, 100316. [CrossRef]
4. Amin, R.; Yasmin, R.; Ruhi, S.; Rahman, M.H.; Reza, M.S. Prediction of chronic liver disease patients using integrated projection-based statistical feature extraction with machine learning algorithms. *Inform. Med. Unlocked* **2023**, *36*, 101155. [CrossRef]
5. Chai, J.; Zeng, H.; Li, A.; Ngai, E.W. Deep learning in computer vision: A critical review of emerging techniques and application scenarios. *Mach. Learn. Appl.* **2021**, *6*, 100134. [CrossRef]
6. Shin, J.P. Optimal stroke-correspondence search method for on-line character recognition. *Pattern Recognit. Lett.* **2002**, *23*, 601–608. [CrossRef]
7. Shin, J. On-line cursive hangul recognition that uses DP matching to detect key segmentation points. *Pattern Recognit.* **2004**, *37*, 2101–2112. [CrossRef]

8. Gopalakrishan, V.; Arun, R.; Sasikumar, L. Handwritten Digit Recognition for Banking System. *Int. J. Eng. Res. Technol.* **2021**, *9*, 313–314.
9. Karakaya, R.; Kazan, S. Handwritten Digit Recognition Using Machine Learning. *Sak. Univ. J. Sci.* **2021**, *25*, 65–71. [CrossRef]
10. Shin, J.; Maniruzzaman, M.; Uchida, Y.; Hasan, M.A.M.; Megumi, A.; Suzuki, A.; Yasumura, A. Important features selection and classification of adult and child from handwriting using machine learning methods. *Appl. Sci.* **2022**, *12*, 5256. [CrossRef]
11. Liu, C.L.; Suen, C.Y. A new benchmark on the recognition of handwritten Bangla and Farsi numeral characters. *Pattern Recognit.* **2009**, *42*, 3287–3295. [CrossRef]
12. Sufian, A.; Ghosh, A.; Naskar, A.; Sultana, F.; Sil, J.; Rahman, M.M.H. BDNet: Bengali Handwritten Numeral Digit Recognition based on Densely connected Convolutional Neural Networks. *J. King Saud Univ. Comput. Inf. Sci.* **2020**, *4*, 2610–2620. [CrossRef]
13. Sen, O.; Fuad, M.; Islam, M.D.N.; Rabbi, J.; Masud, M.; Hasan, K.; Awal, M.D.A.; Fime, A.A.; Fuad, M.D.T.H.; Sikder, D.; et al. Bangla natural language processing: A comprehensive analysis of classical, machine learning, and deep learning-based methods. *IEEE Access* **2022**, *10*, 38999–39044. [CrossRef]
14. Alam, S.; Reasat, T.; Doha, R.M.; Humayun, A.I. NumtaDB—Assembled Bengali Handwritten Digits. *arXiv* **2018**, arXiv:1806.02452.
15. Islam, M.; Shuvo, S.A.; Nipun, M.S.; Sulaiman, R.B.; Nayeem, J.; Haque, Z.; Sourav, M.S.U. Efficient approach of using CNN based pretrained model in Bangla handwritten digit recognition. *arXiv* **2022**, arXiv:2209.13005.
16. Basri, R.; Haque, M.R.; Akter, M.; Uddin, M.S. Bangla handwritten digit recognition using deep convolutional neural network. In Proceedings of the International Conference on Computing Advancements, Dhaka, Bangladesh, 10–12 January 2020; pp. 1–7.
17. Basu, S.; Sarkar, R.; Das, N.; Kundu, M.; Nasipuri, M.; Basu, D.K. Handwritten Bangla digit recognition using classifier combination through DS technique. In *Pattern Recognition and Machine Intelligence, Proceedings of the First International Conference, PReMI 2005, Kolkata, India, 20–22 December 2005*; Proceedings 1; Springer: Berlin/Heidelberg, Germany, 2005; pp. 236–241.
18. Shopon, M.; Mohammed, N.; Abedin, M.A. Bangla handwritten digit recognition using autoencoder and deep convolutional neural network. In Proceedings of the 2016 International Workshop on Computational Intelligence (IWCI), Dhaka, Bangladesh, 12–13 December 2016; IEEE: Piscataway Township, NJ, USA, 2016; pp. 64–68.
19. Nasir, M.K.; Das, T.R.; Hasan, S.; Jani, M.R.; Tabassum, F.; Islam, M.I. Hand Written Bangla Numerals Recognition for Automated Postal System. *IOSR J. Comput. Eng.* **2013**, *09*, 158–171. [CrossRef]
20. Scarmana, G. Lossless data compression of grid-based digital elevation models: A PNG image format evaluation. *ISPRS Ann. Photogramm. Remote Sens. Spat. Inf. Sci.* **2014**, *2*, 313–319. [CrossRef]
21. Mubarak, A.S.; Serte, S.; Al-Turjman, F.; Ameen, Z.S.; Ozsoz, M. Local binary pattern and deep learning feature extraction fusion for COVID-19 detection on computed tomography images. *Expert Syst.* **2022**, *39*, 1–13. [CrossRef]
22. Chen, T.; Guestrin, C. XGBoost: A Scalable Tree Boosting System. In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, 13–17 August 2016; ACM: New York, NY, USA, 2016; pp. 785–794. [CrossRef]
23. Alanne, K.; Sierla, S. An overview of machine learning applications for smart buildings. *Sustain. Cities Soc.* **2022**, *76*, 103445. [CrossRef]
24. Zamzami, I.F. Deep Learning Models Applied to Prediction of 5G Technology Adoption. *Appl. Sci.* **2023**, *13*, 119. [CrossRef]
25. Razzak, M.I.; Naz, S.; Zaib, A. Deep learning for medical image processing: Overview, challenges and the future. *Lect. Notes Comput. Vis. Biomech.* **2018**, *26*, 323–350. [CrossRef]
26. Sliti, O.; Hamam, H.; Amiri, H. CLBP for scale and orientation adaptive mean shift tracking. *J. King Saud Univ. Comput. Inf. Sci.* **2018**, *30*, 416–429. [CrossRef]
27. Spanhol, F.A.; Oliveira, L.S.; Petitjean, C.; Heutte, L. A Dataset for Breast Cancer Histopathological Image Classification. *IEEE Trans. Biomed. Eng.* **2016**, *63*, 1455–1462. [CrossRef] [PubMed]
28. Guo, Z.; Zhang, L.; Zhang, D. A completed modeling of local binary pattern operator for texture classification. *IEEE Trans. Image Process.* **2010**, *19*, 1657–1663. [CrossRef] [PubMed]
29. Dalal, N.; Triggs, B. Histograms of oriented gradients for human detection. In Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05), San Diego, CA, USA, 20–26 June 2005; IEEE: Piscataway Township, NJ, USA, 2005; Volume 1, pp. 886–893.
30. Ameh Joseph, A.; Abdullahi, M.; Junaidu, S.B.; Hassan Ibrahim, H.; Chiroma, H. Improved multi-classification of breast cancer histopathological images using handcrafted features and deep neural network (dense layer). *Intell. Syst. Appl.* **2022**, *14*, 200066. [CrossRef]
31. Ahmed, M.Z.I.; Sinha, N.; Phadikar, S.; Ghaderpour, E. Automated Feature Extraction on AsMap for Emotion Classification Using EEG. *Sensors* **2022**, *22*, 2346. [CrossRef]
32. Sadik, R.; Majumder, A.; Biswas, A.A.; Ahammad, B.; Rahman, M.M. An in-depth analysis of Convolutional Neural Network architectures with transfer learning for skin disease diagnosis. *Healthc. Anal.* **2023**, *3*, 100143. [CrossRef]
33. Srinivas, C.; Nandini, N.P.; Zakariah, M.; Alothaibi, Y.A.; Shaukat, K.; Partibane, B.; Awal, H. Deep Transfer Learning Approaches in Performance Analysis of Brain Tumor Classification Using MRI Images. *J. Healthc. Eng.* **2022**, *2022*, 3264367. [CrossRef]
34. Stančić, A.; Vyroubal, V.; Slijepčević, V. Classification Efficiency of Pre-Trained Deep CNN Models on Camera Trap Images. *J. Imaging* **2022**, *8*, 20. [CrossRef]
35. Yadav, S.S.; Jadhav, S.M. Deep convolutional neural network based medical image classification for disease diagnosis. *J. Big Data* **2019**, *6*, 1–18. [CrossRef]

36. Sann, S.S.; Win, S.S.; Thant, Z.M. An analysis of various image pre-processing techniques in butterfly image. *Int. J. Adv. Res. Dev.* **2021**, *6*, 1–4.
37. Paul, O. Image Pre-processing on NumtaDB for Bengali Handwritten Digit Recognition. In Proceedings of the 2018 International Conference on Bangla Speech and Language Processing, ICBSLP 2018, Sylhet, Bangladesh, 21–22 September 2018; pp. 1–6. [CrossRef]
38. Renjith, S.; Abraham, A.; Jyothi, S.B.; Chandran, L.; Thomson, J. An ensemble deep learning technique for detecting suicidal ideation from posts in social media platforms. *J. King Saud Univ. Comput. Inf. Sci.* **2022**, *34*, 9564–9575. [CrossRef]
39. Kabir, M.H.; Ahmad, F.; Hasan, M.A.M.; Shin, J. Gender Recognition of Bangla Names Using Deep Learning Approaches. *Appl. Sci.* **2022**, *13*, 522. [CrossRef]
40. Caruana, R.; Lawrence, S.; Giles, L. Overfitting in neural nets: Backpropagation, conjugate gradient, and early stopping. *Adv. Neural Inf. Process. Syst.* **2000**, *13*, 402–408.
41. Wolpert, D. Stacked Generalization (Stacking). *Neural Netw.* **1992**, *5*, 241–259. [CrossRef]
42. Gour, M.; Jain, S. Automated COVID-19 detection from X-ray and CT images with stacked ensemble convolutional neural network. *Biocybern. Biomed. Eng.* **2022**, *42*, 27–41. [CrossRef] [PubMed]
43. Adnan, M.; Rahman, F.; Imrul, M.; Al, N.; Shabnam, S. Handwritten Bangla character recognition using inception convolutional neural network. *Int. J. Comput. Appl.* **2018**, *181*, 48–59. [CrossRef]
44. Pal, U.; Chaudhuri, B.B. Automatic recognition of unconstrained off-line Bangla handwritten numerals. In Proceedings of the Advances in Multimodal Interfaces—ICMI 2000: Third International Conference, Beijing, China, 14–16 October 2000; Springer: Berlin/Heidelberg, Germany, 2001; pp. 371–378.
45. Wen, Y.; Lu, Y.; Shi, P. Handwritten Bangla numeral recognition system and its application to postal automation. *Pattern Recognit.* **2007**, *40*, 99–107. [CrossRef]
46. Hassan, T.; Khan, H.A. Handwritten bangla numeral recognition using local binary pattern. In Proceedings of the 2015 International Conference on Electrical Engineering and Information Communication Technology (ICEEICT), Dhaka, Bangladesh, 21–23 May 2015; IEEE: Piscataway Township, NJ, USA, 2015; pp. 1–4.
47. Wen, Y.; He, L. A classifier for Bangla handwritten numeral recognition. *Expert Syst. Appl.* **2012**, *39*, 948–953. [CrossRef]
48. Basu, S.; Das, N.; Sarkar, R.; Kundu, M.; Nasipuri, M.; Basu, D.K. A novel framework for automatic sorting of postal documents with multi-script address blocks. *Pattern Recognit.* **2010**, *43*, 3507–3521. [CrossRef]
49. Saha, C.; Masuma, F.; Ahammad, K.; Muzammel, C.S.; Mohibullah, M. Real time Bangla Digit Recognition through Hand Gestures on Air Using Deep Learning and OpenCV. *Int. J. Curr. Sci. Res. Rev.* **2022**, *5*. [CrossRef]
50. Bhattacharya, U.; Chaudhuri, B.B. Handwritten numeral databases of Indian scripts and multistage recognition of mixed numerals. *IEEE Trans. Pattern Anal. Mach. Intell.* **2008**, *31*, 444–457. [CrossRef] [PubMed]