IEEE Access
Multidisciplinary : Rapid Review : Open Access Journal

Received 18 April 2024, accepted 1 June 2024, date of publication 6 June 2024, date of current version 20 June 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3410837

APPLIED RESEARCH

MultiModal Ensemble Approach Leveraging Spatial, Skeletal, and Edge Features for Enhanced Bangla Sign Language Recognition

KHAN ABRAR SHAMS[1], MD. RAFID REAZ[1], MOHAMMAD RYAN UR RAFI[1], SANJIDA ISLAM[1], MD. SHAHRIAR RAHMAN[1], RAFEED RAHMAN[1], MD. TANZIM REZA[1], MOHAMMAD ZAVID PARVEZ[2], SUBRATA CHAKRABORTY[3,4,5], (Senior Member, IEEE), BISWAJEET PRADHAN[4,6], (Senior Member, IEEE), AND ABDULLAH ALAMRI[7]

[1]Department of Computer Science and Engineering, School of Data and Sciences, Brac University, Dhaka 1212, Bangladesh
[2]School of Computing, Mathematics and Engineering, Charles Sturt University, Bathurst, NSW 2795, Australia
[3]School of Science and Technology, University of New England, Armidale, NSW 2351, Australia
[4]Centre for Advanced Modelling and Geospatial Information Systems (CAMGIS), Faculty of Engineering and Information Technology, University of Technology Sydney, Ultimo, NSW 2007, Australia
[5]Griffith Business School, Griffith University, Nathan, QLD 4111, Australia
[6]Earth Observation Centre, Institute of Climate Change, Universiti Kebangsaan Malaysia (UKM), Bangi, Selangor 43600, Malaysia
[7]Department of Geology and Geophysics, College of Science, King Saud University, Riyadh 11451, Saudi Arabia

Corresponding authors: Subrata Chakraborty (subrata.chakraborty@une.edu.au) and Biswajeet Pradhan (biswajeet.pradhan@uts.edu.au)

The work of Subrata Chakraborty and Biswajeet Pradhan was supported in part by the Centre for Advanced Modelling and Geospatial Information Systems (CAMGIS), University of Technology Sydney; and in part by King Saud University, through the Researchers Supporting Project under Grant RSP2024 R14.

ABSTRACT Sign language is the predominant mode of communication for individuals with auditory impairment. In Bangladesh, BdSL or Bangla Sign Language is widely used among the hearing-impaired population. However, because of the general public's limited awareness of sign language, communicating with them using BdSL can be challenging. Consequently, there is a growing demand for an automated system capable of efficiently understanding BdSL. For automation, various Deep Learning (DL) architectures can be employed to translate Bangla Sign Language into readable digital text. The automation system incorporates live cameras that continuously capture images, which a DL model then processes. However, factors such as lighting, background noise, skin tone, hand orientations, and other aspects of the image circumstances may introduce uncertainty variables. To address this, we propose a procedure that reduces these uncertainties by considering three modalities: spatial information, skeleton awareness, and edge awareness. We introduce three image pre-processing techniques alongside three CNN models. The CNN models are combined using nine distinct ensemble meta-learning algorithms, with five of them being modifications of averaging and voting techniques. In the result analysis, our individual CNN models achieved higher training accuracy at 99.77%, 98.11%, and 99.30%, respectively, than most of the other state-of-the-art image classification architectures, except for ResNet50, which achieved 99.87%. Meanwhile, the ensemble model attained the highest accuracy of 95.13% on the testing set, outperforming all individual CNN models. This analysis demonstrates that considering multiple modalities can significantly improve the system's overall performance in hand pattern recognition.

INDEX TERMS Bangla sign language (BdSL), convolutional neural network, ensemble method.

I. INTRODUCTION

A. PROBLEM STATEMENT

The World Health Organization (WHO) estimates that 430,000,000 people, or 1 in every 20 people worldwide, are diagnosed with hearing disabilities [1]. This means more than 5% of the world's population suffers from hearing impairment. According to a study, this ratio is expected to grow, with 8.87% of the population, or more than 700 million people, projected to experience auditory issues by the year 2050 [2]. In Bangladesh, the prevalence of impairment is nearly double the global average, standing at 9.6% [3]. Due to the increasing

The associate editor coordinating the review of this manuscript and approving it for publication was Joewono Widjaja.

© 2024 The Authors. This work is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 License.
For more information, see https://creativecommons.org/licenses/by-nc-nd/4.0/

trend in hearing impairment, sign language plays a vital role in helping, assisting, and fostering a connection between individuals with normal hearing and those from the hearing impaired community. BdSL, or Bangla Sign Language, is one of the major sign languages used in Bangladesh. While BdSL and West Bengal Sign Language (WBSL) are lexically similar, they are prominently used in their respective regions [4]. BdSL is a comprehensive language with symbols or alphabets that mirror numerous linguistic similarities in spoken languages. Despite the importance of learning sign language, it is challenging for individuals with normal hearing to quickly master it, highlighting the urgent need for a system to interpret sign language. This research aims to fulfill the demand for interpreting Bangla Sign Language and provides comprehensive guidelines to enhance the system's accuracy. Interpreting sign language falls under the classification paradigm of supervised learning. Classification, which involves identifying or categorizing an object/observation, is presently among the extensively researched areas in computer vision and pattern recognition. The classification problem can be generalized into two types: Binary classification and Multi-class classification [5]. Binary classification is straightforward, while multi-class classification complexity depends on the number of classes, requiring sufficient samples to train a model with high accuracy [6]. The scarcity of quality datasets in Bangla Sign Language is a current challenge. Until now, only a few alphabet-level sign language datasets are available, and word-level datasets are even scarcer. Bangla Sign Language comprises 38 hand signs representing 51 alphabets [7], and these 38 alphabets can form any word. Conversely, a word-level Bangla Sign Language dataset, MVBSL-W50, contains 50 classes of words [8]. While a word-level classification model may be user-friendly, it could demand significant computational resources and physical memory during real-life implementation.

B. BACKGROUND INFORMATION

Communication is a crucial skill for socialization. Individuals with hearing impairment require special modes of communication, and extensive research has been done on sign language across various linguistic contexts. Thomas Hopkins Gallaudet dedicated over three centuries to refining American Sign Language. In Hartford, Connecticut, Gallaudet University was founded as an American Asylum, becoming the first national school for hearing impaired students in the United States. Currently, American Sign Language is one of the most commonly used languages in the American judicial system, with its influence spreading to many countries. Meanwhile, in both West Bengal and Bangladesh, people generally follow their own methods of sign language, namely WBSL and BdSL. The Calcutta Hearing Impaired and Dumb School, established in 1893, played a crucial role in developing BdSL. Although both WBSL and BdSL were initially a single sign language, the Indo-Pakistan partition and the Independence of Bangladesh in 1971 caused them to diverge [4]. A notable dataset of Bangla sign language, BdSL49, comprises 29,490 images of 49 individual Bangla alphabet signs in the Bangla Sign Language (BdSL). The dataset includes images of 14 adult individuals with diverse backgrounds and appearances. Careful preparation and various noise elimination strategies have been employed in creating this dataset [9].

Detection problems, such as sign language detection, begin with data collection to create a dataset. A lack of samples in the training data can lead to a faulty and poor approximation. To address this issue, a dataset consisting of the most prominent image database for BdSL Alphabets and Numerals was introduced. This dataset is designed to minimize similarity between different classes while handling diverse information, including a range of skin tones and backgrounds. Continuous sign language is often equated with sentence-level American Sign Language. Notably, sign language in different languages may yield different outcomes for similar approaches. Testing with a single and simple model, such as Inception V3 and RNN, can pose challenges in detecting facial features, and accuracy may only be sufficient if the data is trained on specific skin tones [10]. Feeding multi-modal information or information from multiple data sources to the system can help enhance the classification network [11]. Most studies on BdSL have provided extensive guidelines on model definition, but a gap has been identified in studies that pre-process these data.

C. RESEARCH OBJECTIVE

Our study aims to achieve better accuracy in recognizing Bangla Sign Language and translating it into human-readable language. Datasets containing images of various hand gestures are used to train multiple CNN models. The proposed CNN models are designed to enhance the identification of BdSL with improved accuracy by aggregating the outcomes from different CNN architectures. We utilized a dataset comprising numerous images featuring diverse hand gestures in BdSL to fulfill our objective. Subsequently, three suitable CNN models are trained on this dataset, and the average output score from these models is considered to achieve better accuracy by amalgamating the learning from all models. Multiple images for identical expressions, signed by different native signers, contribute to more efficient model learning. These data are processed and fed into our models.

Following an ensemble approach, multiple CNN models are trained, with each CNN model structured with a suitable sequence of convolutional, max-pooling, and fully connected layers. Each CNN model is trained on a different modality of the base image, allowing consideration of different patterns from the images. For instance, Model 2 is trained to classify based on the hand landmark pattern, while Model 3 focuses on classifying on-edge patterns of the hand. Ultimately, averaging the output score from all the models is performed to achieve improved predictive performance and high accuracy rates for recognizing the sign gestures of BdSL.

D. RESEARCH CONTRIBUTION

In this paper, we propose a novel procedure-based sign language recognition system. We trained light CNN models with three modalities to overcome environment-dependent obstacles such as lighting issues, skin color, and different hand structures. After training, we cross-checked validation results of our models individually against many state-of-the-art predefined image classification architectures such as Inception-V3, DenseNet, Xception, VGG19, and ResNet50. Afterward, we tested multiple meta-learning algorithms for the ensemble model and cross-checked the testing data with all mentioned state-of-the-art classification models. The main contributions of this research are as follows:

- We propose a system where the pre-processing technique takes precedence over the model's architecture. This paper provides extensive details on how we can pre-process samples so the classification model can provide a prediction with a more meaningful explanation.
- We propose a novel procedure-based system to tackle environment-dependent uncertainty variables like skin color, different lighting, noisy background, and hand structure. By reducing the uncertainty variables, classification models can provide a prediction with higher precision. This approach was inspired by the process of classifying hand signs, which considers modalities such as skin color, skeleton position, and the edges of the hand.
- We propose three lighter CNN classification models instead of a heavy neural network architecture for an efficient system. A lighter CNN model has fewer parameters, requires less computational power, and is faster and more efficient than their popular counterpart. Even in terms of training the models, it will require less resource utilization as each CNN model can be trained in parallel across separate devices. Our proposed models performed better at generalizing unseen data as they are less prone to overfitting while training.
- We propose a particular type of ensemble meta-learning algorithm for the best combination of output where the meta-learning algorithm is integrated with the training average precision value of the models. We tested out five different modified meta-learning algorithms for predicting the best outcome. This approach also reduces any weak predictions by considering all the outputs from the models.

II. LITERATURE REVIEW

A. GESTURE RECOGNITION

A study on a gesture identification model utilizing an ensemble of three CNN models was conducted. For training, the hand gesture was segmented using the binary thresholding method and then fed to three CNN-based classifiers: VGGNet, GoogLeNet, and AlexNet. The output accuracy of the models is then calculated by a meta-learner built upon the averaging technique for optimum prediction. Among the two datasets that were used, the result on Dataset-i was 99.80%, on Dataset-ii was 96.50%, and on a self-constructed dataset it was 99.76% accurate [12].

B. PATTERN RECOGNITION IN SIGN LANGUAGE RECOGNITION

Sign language recognition is inherently intertwined with the principles of pattern recognition. Intricate patterns, such as specific handshapes, movements, and gestures, convey meaning in sign language. Recognition and interpretation of these gestures and movements can be achieved through a classification model if the model can effectively learn the underlying patterns. Therefore, it is crucial that the model learns essential features such as hand shape, movement trajectory, and facial expressions for each sign. One study proposes an approach that utilizes co-independent data streams to capture various sign language gestures. The authors specifically focus on synchronizing and capturing dependencies between sign language components. By incorporating an attention mechanism, the model effectively combines hand characteristics (features) such as positions and structure with their relevant spatiotemporal context, leading to enhanced hand-sign classification. In terms of performance, the reported error rates on the dev and test sets are 32.74% and 33.29%, respectively [13].

In continuous sign language recognition, where there are no direct links between video frames and signs, prior approaches, such as 1D-CNN models, struggled to apprehend nuanced relationships between neighboring frames. To address this challenge, Pan Xie et al. proposed mLTSF-Net. This approach begins by identifying similar neighboring frames using multi-scale receptive regions to adapt to various gloss times. The authors employed position-aware convolution for temporal consistency and a content-dependent aggregator to capture local-temporal frame representations [14]. For isolated Sign Language Recognition (SLR), Songyao Jiang et al. propose the novel SAM-SLR-v2 (Skeleton Aware Multi-modal Framework with a Global Ensemble Model) model. The suggested framework combines multi-modal feature representations to increase SLR accuracy, achieving state-of-the-art performance with significant improvements over existing approaches [15].

C. BANGLA SIGN LANGUAGE RECOGNITION

In a study, a comprehensive dataset for Bengali Sign Language (BdSL) was published and trained using CNN. The study demonstrates how convolutional neural networks can be utilized to accurately identify various BdSL indications. In the suggested model, the testing accuracy for numbers was 100%, while the testing accuracy for the Bengali Sign Language alphabet's characters was 99.83%. Finally, an accuracy of 99.80% was achieved by combining characters and numbers [16]. Shahjalal Ahmed et al. examined the evaluation scores of their CNN model using multiple datasets in their paper. The model comprises two convolution layers with 2 × 2 max pooling for each layer and classifies digit-based hand signs with 92% accuracy. It was trained for 100 epochs with RMSProp optimizer, and CCE (Categorical Cross Entropy) was used as the cost function [17].

In the paper "Bangla Sign Digits: A Dataset For Real Time Hand Gesture Recognition" [39], a preprocessing step similar to our first technique was proposed.This step involves normalizing and resizing the images to 64 × 64, followed by using a custom CNN model. Tasmere et al. focused solely on sign digits (0-9) in their study and achieved an accuracy of 97.63%. Our work builds upon these preprocessing steps and introduces two additional techniques and models aimed at enhancing the system's robustness.

Abedin et al., proposed a new architecture called the "Concatenated BdSL Network" that combines a Convolutional Neural Network (CNN) based image network with a pose estimation network. The proposed approach aimed to improve recognition accuracy by incorporating visual features and hand posture. The novel approach scored 91.51% on the test set, indicating promising results. The additional pose estimation network proved beneficial in dealing with the intricacies of BdSL symbols, as suggested by the experimental outcomes [33].

In another study, M.A Hossen researched BdSLR (Bangla Sign Language Recognition) using DCNN (Deep Convolutional Neural Networks). They used the Bengali Sign Language (BSL) dataset and planned to work with one-handed signs. Using the concept of fine-tuning and transfer learning, they have modified the VGG19 pre-trained model. They achieved this by adjusting the weights in small increments with a slower learning rate. Their model performed with a validation accuracy of 84.68% which is pretty high considering the small dataset [34].

D. SIGN LANGUAGE RECOGNITION

A study shows that an efficient technique for identifying Indian Sign Language (ISL) uses a Convolutional Neural Network, an architecture consisting of convolutional layers, ReLU layers, and max-pooling layers. The system achieved an accuracy of 99.90% on 35,000 samples of 100 classes. In the performance comparison, SGD demonstrated superior results over Adam and RMSProp optimizers [18]. A paper on CSLR showcased a video-based identification technique for the CNN system. A gesture recognition model draws out upper body photos directly from moving images and identifies the gestures using a pre-trained CNN model. The technique produced an accuracy of 99% on their self-build dataset [19]. It is possible to create a custom new dataset from an existing dataset while completely altering the modality of the image. B Sundar et al., demonstrated how Google's Mediapipe can help generate hand landmarks by capturing images for 30 images. Using the dataset to train an RNN-based architecture (LSTM), they achieved 99% accuracy [20]. A different approach offers an architecture for developing a ConvNet, which lays hold of data depth and image intensity. The evaluation demonstrates that the developed convolutional network transcends previous research with 82% precision and 80% recall [21]. Most of the datasets on American sign language are usually produced in a studio, though it creates carefully refined data which is not very suitable for real-life implementation, where noise and low-light conditions can be abundant. Therefore, a proposed system is established on an iterative attention mechanism that focuses on the Region Of Interest (ROI) synchronously with increasing levels of resolution to overcome the segmentation problem [22]. Alkhalid [38], demonstrated how background subtraction and edge detection can produce better result.

E. METHODOLOGICAL WORKFLOW

The primary objective of our research is to introduce an innovative approach to enhance the recognition accuracy of Bangla sign language while concurrently developing lightweight CNN models that mitigate computational demands. The dataset used for the research consists of 11,061 training images over 38 different signs. Each hand sign is considered a class consisting of approximately 300 samples. First, the training set was divided into validation and training sets. The pre-processing technique, denoted as 1, 2, and 3, converts the original image to three different modalities. An additional 1,520 photos were used as the testing set. When the processed data was prepared, we constructed three distinct CNN models. The models were trained using the training and validation sets. Various evaluation matrices were considered to determine whether the CNN model is adequate. Afterward, the model was saved when it reached a satisfactory result. Ensemble learning was used to consider the prediction of all the models, increasing the final prediction's precision. Ensemble learning is robust for enhanced classification, clustering, and regression problems [23]. It is a technique that combines the output findings or predictions of numerous deep learning or machine learning models on the same dataset. The base model and the meta-model make up the ensemble architecture. As weak learners, the base models serve as the framework for our meta-model, while the meta-model discovers the best method for combining base model predictions. A meta-model can be both a classification model and a predetermined algorithm. Figure 1 represents the main workflow of this paper in a flowchart. During the ensemble testing, the static weight ensemble model provided the best result.

III. DATASET

A. DATA CONFIGURATION AND ANALYSIS

The dataset tested in our study was referenced by Rafi et al. in their paper on image-based Bengali Sign Language Recognition for Hearing Impaired and Dumb Communities [24]. This dataset comprises 38 different hand signs from 320 different people. Each class contains a similar number of instances so it is well balanced, a factor that significantly impacts performance metrics [25]. Each of the

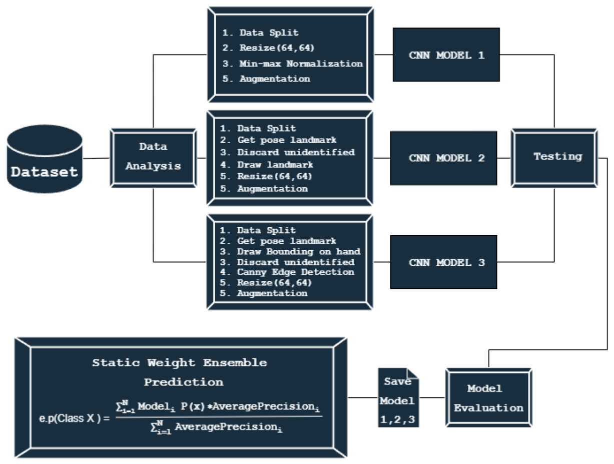


FIGURE 1. Research workflow.

38 hand signs represents a Bangla alphabets and in some cases, multiple alphabets are represented through the same hand sign. Most of the images contain hand signs made using a single hand, except for one class where both hands are used to complete the sign. This aligns with the dataset's focus on single-hand sign Bangla Sign Language. The images are mainly captured against complex backgrounds, adding to the challenge of accurate hand-sign detection. Additionally, some hand signs are photographed in front of densely cluttered backgrounds, further complicating the visual context and testing the robustness of the recognition model. Figure 2 and Figure 3 depict the training and testing samples obtained from the dataset.

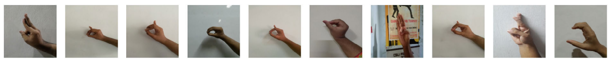


FIGURE 2. Random images from the training set.

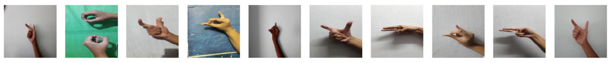


FIGURE 3. Random images from the testing set.

B. DATA PREPROCESSING

Hand landmarks were extracted from the images utilizing the state of the art capabilities of Google's MediaPipe framework. These hand landmarks offered invaluable insights on the hand gestures which helped to estimate the hand signs vividly. Furthermore, we computed a bounding box on the images leveraging the extracted landmarks to find the ROI that effectively helped the training of our model.

Moreover, canny edge detection is also used for preprocessing to highlight the significant boundaries and contours in the images. Given the backdrop of hands against different backgrounds, canny edge detection helped to filter out the noise and other irrelevant details from the images, speeding up the learning and increasing the accuracy of our model.

1) MODEL 1 PREPROCESSING TECHNIQUES

Proper pre-processing techniques can reduce the training complexity and make the models more accurate. A vast number of training data is required to achieve a well-performing model. For general practice, researchers suggest at least 1,000 instances per class to achieve a robust system, whereas only 320 instances per class are provided in the obtained dataset. Thus, we augmented each photo six times using rotation, width shift, height shift, shear, and zoom. Subsequently, we split our 11,061 × 7 images into training and validation sets.

$$x_{scaled} = \frac{x - x_{min}}{x_{max} - x_{min}}$$

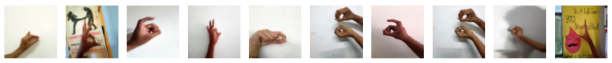


FIGURE 4. Random images from the preprocessed training set.

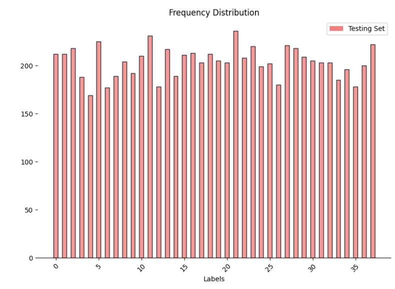


FIGURE 5. Validation set.

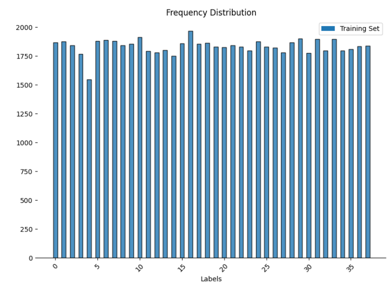


FIGURE 6. Training set.

Another 1,520 images of all 38 classes were used for our testing set. These 1,520 images were not augmented since they were to be used for the evaluation of our proposed model. During augmentation, we intentionally avoided horizontal flips on the image samples, as it can change the meaning of the hand sign. After augmentation, we used the min-max normalization to scale the pixel values of the dataset within a range of 0-1. The normalization process can help us to limit computational usage and provide faster predictions [26].

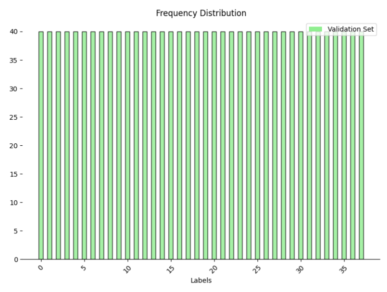


FIGURE 7. Testing set.

2) MODEL 2 PREPROCESSING TECHNIQUES

Similarly, for model 2, the augmentation process is set to take place. However, before augmentation, a few more steps are required. Model 2 focuses on addressing the weaknesses identified in our Model 1. Model 1 may perform poorly in situations where the background is very noisy, particularly when background objects are larger than the arm or hand in the photo. The aim is also to utilize different modalities to enhance predictions in the ensemble test. To achieve these goals, utilizing MediaPipe's hand landmarks detection is an ideal choice [27]. The workflow for the pre-processing technique for model 2 is simple. The first step involves taking the image and resizing it to (128,128).

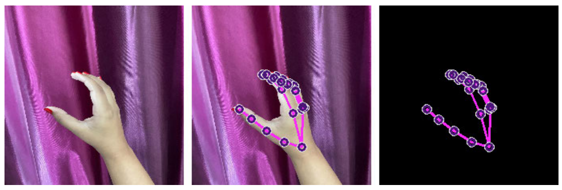


FIGURE 8. Pre-processing technique visual.

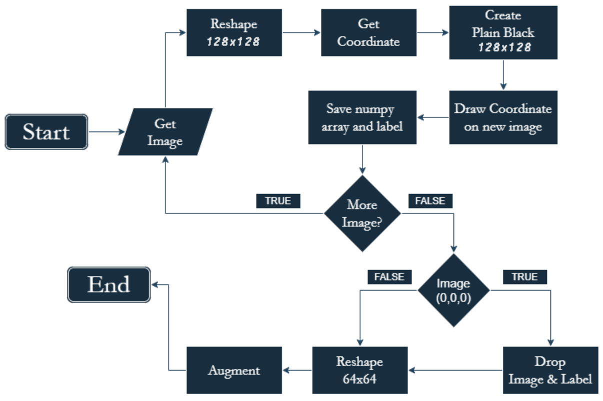


FIGURE 9. Flowchart pre-processing model 2.

The hand landmark detection algorithm is used while setting minimum detection confidence to 0. After fetching the coordinates for the hand and finger landmarks, the crossroads on the 128 × 128 image are marked where every pixel value is set to (0,0,0). Afterward, the images are resized to 64 × 64 dimension. Sometimes, the MediaPipe cannot draw the hand pose, resulting in a completely black image. For such cases, the sample has been discarded. This same processing technique is applied to all the training samples (11,061).

After creating the new pre-processed dataset, the dataset was augmented using similar techniques that were used in case of model 1. Figure 9 is a flowchart of the entire pre-processing steps for model 2. Figure 8 visualizes how these data are processed and what these data might look like in each step. Figure 10 displays a collection of randomly selected samples from the preprocessed training dataset specifically designed for model 2.

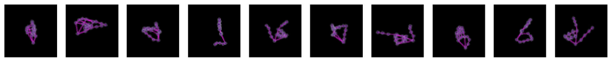


FIGURE 10. Random images from the pre-processed training set model 2.

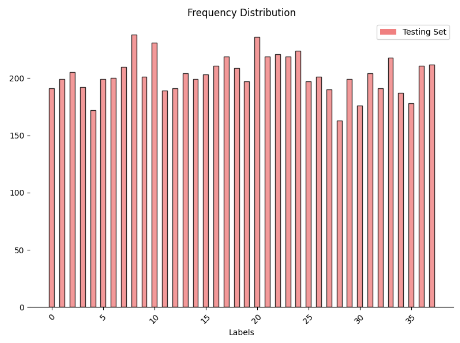


FIGURE 11. Validation set.

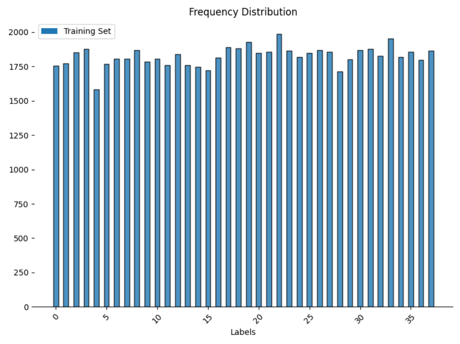


FIGURE 12. Training set.

Figure 11, 12, and 13 are the frequency distribution tables for validation data, training data, and testing data. On the x-axis, you have individual labels per unit, and the y-axis represents the number of instances for each unit.

3) MODEL 3 PREPROCESSING TECHNIQUES

Model 3 requires some extensive pre-processing techniques. The target for Model 3 was to overcome noisy backgrounds and remove complete dependency on MediaPipe. A problem with model 2 was that it could sometimes not find the

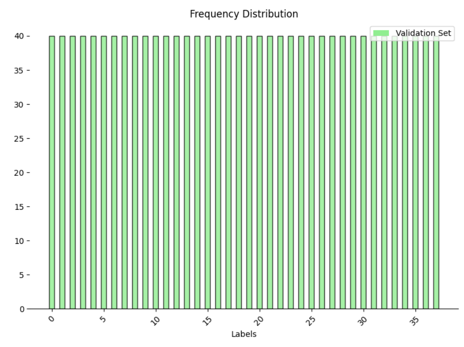


FIGURE 13. Testing set.

hand and ended up forwarding a complete black image for prediction. To overcome the problem, the MediaPipe hand pose estimation technique was used to find the maximum and minimum x and y coordinates.

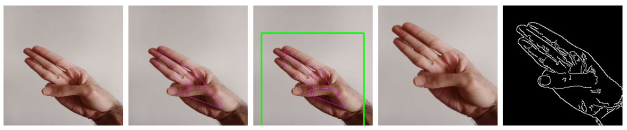


FIGURE 14. Pre-processing technique visual.

The pre-processing workflows are straightforward, and several minor techniques are integrated to enhance the quality of the input data. Let us consider a single input image with the corresponding label. We start with reshaping the image into a dimension of (128,128). Later on, MediaPipe hand landmark detection [27] is used to filter out x-max, y-max, x-min, and y-min values. Using these filtered values, we crop the image with the following coordinates: (x-min, y-min), (x-min, y-max), (x-max, y-min), and (x-max, y-max). If hand landmarks are not found, the process will continue without cropping, and the image will be reshaped with dimensions (64, 64). The cropping helps to remove most of the noisy background and brings the hand to the center of the image. The Canny edge detection algorithm [28] is used in the next step to bring out the edges. It not only serves as the edge detection technique but also drastically reduces the amount of data to be processed.

Firstly, the image was blurred using a Gaussian filter with a kernel size of 3 × 3:

$$H_{ij} = \frac{1}{2\pi\sigma^2} \exp\left(-\frac{(i-2)^2 + (j-2)^2}{2\sigma^2}\right); \quad 1 \leq i, j \leq 3 \tag{1}$$

Using the Sobel kernel in both horizontal and vertical directions on the image, we obtain the first derivative in the horizontal direction and the second derivative in the vertical direction [37]. We can determine the edge gradient

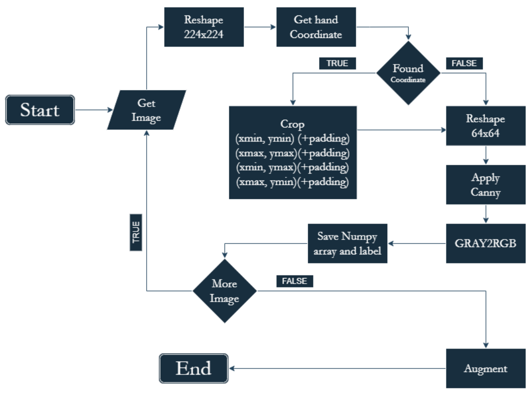


FIGURE 15. Flowchart preprocessing model 3.

and direction for each pixel using the following equations:

$$\text{Edge\_Gradient}(G) = \sqrt{G_x^2 + G_y^2} \tag{2}$$

$$\text{Angle}(\theta) = \tan^{-1}\left(\frac{G_y}{G_x}\right) \tag{3}$$

Using Canny changes the image to grayscale [38], which is then reverted back to RGB format to match the model input dimension. The images and labels are saved accordingly. Figure 14 visualizes the step-by-step changes in a sample as it goes through the preprocessing stage, while Figure 15 depicts the complete flowchart of the preprocessing technique for model 3. Figure 16 displays a collection of randomly selected samples from the preprocessed training set specifically designed for model 3.

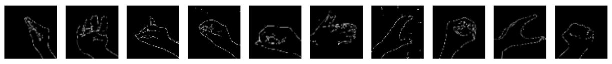


FIGURE 16. Random images from the pre-processed training set model 3.

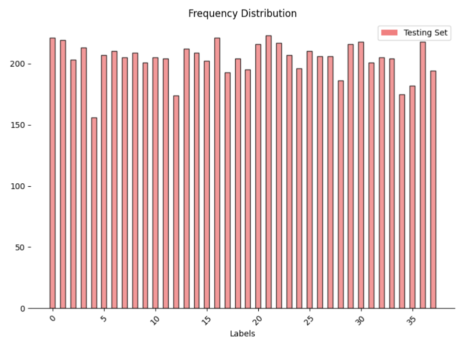


FIGURE 17. Validation set.

IV. METHODOLOGY

In this section, we describe the models proposed for Bangla Sign Language Recognition. The proposed models are based on convolutional neural network (CNN) architectures, which excel at performing image processing tasks such as image

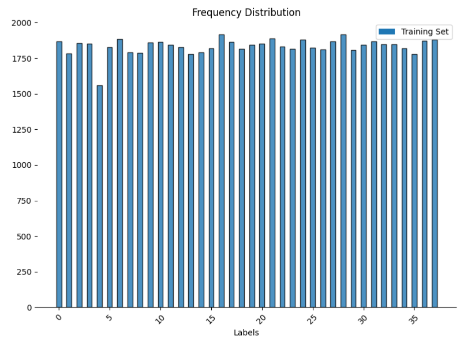


FIGURE 18. Training set.

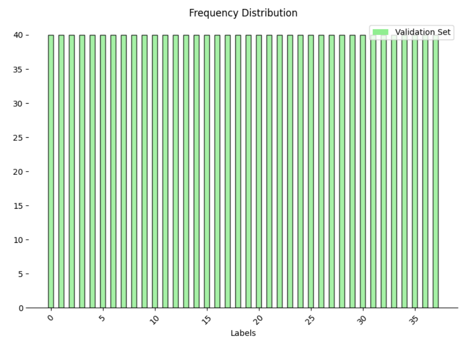


FIGURE 19. Testing set.

classification and pattern recognition. These CNN models are designed to capture and exploit the hierarchical spatial structure present in images.

A. CONVOLUTIONAL NEURAL NETWORK

The CNN is built on the same principle as the human brain. The brain's neuron and the convolutional neuron in CNN work similarly by sharing and communicating with local neurons. A Convolutional Neural Network (CNN) is a distinct neural network type engineered for tasks involving structured data like images or time series. The CNN architecture usually consists of three layers: convolutional, pooling, and fully connected [29]. Each layer of CNN works to detect different features from the input image. In order to complete our task of sign language recognition, layers have to learn the pattern of the fingers and the hand. A kernel is an element of the convolution layer that outputs edge features, and the edge features slowly get better after each convolution layer. The final layer, or the fully connected layer, provides the final output, which is the prediction of the hand sign for our case.

B. PROPOSED CNN MODELS FOR SIGN LANGUAGE DETECTION

In order to address the task of sign language recognition, we propose three CNN models, each with its own characteristics and trade-offs. These models have been designed to cater to different requirements in terms of prediction speed and accuracy, allowing flexibility for various real-world applications.

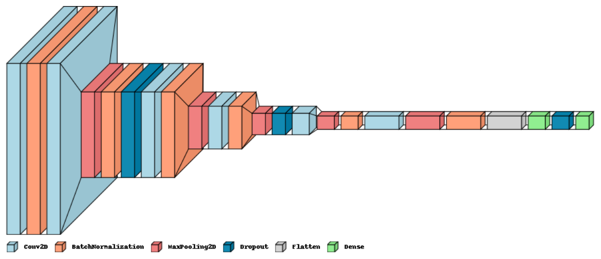


FIGURE 20. Illustration of the proposed cnn architecture for Model 1.

TABLE 1. Model-1 summary.

Layer (type)	Output Shape	Param #
conv2d	(None, 64, 64, 32)	896
batch_normalization	(None, 64, 64, 32)	128
conv2d_1	(None, 64, 64, 48)	13872
max_pooling2d	(None, 32, 32, 48)	0
batch_normalization_1	(None, 32, 32, 48)	192
dropout	(None, 32, 32, 48)	0
conv2d_2	(None, 32, 32, 64)	27712
batch_normalization_2	(None, 32, 32, 64)	256
max_pooling2d_1	(None, 16, 16, 64)	0
conv2d_3	(None, 16, 16, 128)	73856
batch_normalization_3	(None, 16, 16, 128)	512
max_pooling2d_2	(None, 8, 8, 128)	0
dropout_1	(None, 8, 8, 128)	0
conv2d_4	(None, 8, 8, 256)	295168
max_pooling2d_3	(None, 4, 4, 256)	0
batch_normalization_4	(None, 4, 4, 256)	1024
conv2d_5	(None, 2, 2, 512)	1180160
max_pooling2d_4	(None, 1, 1, 512)	0
batch_normalization_5	(None, 1, 1, 512)	2048
flatten	(None, 512)	0
dense	(None, 256)	131328
dropout_2	(None, 256)	0
dense_1	(None, 38)	9766
Total params		1736918
Trainable params		1734838
Non-trainable params		2080

1) MODEL-1, MODEL-2 AND MODEL-3

Model-1 is built on a sequence of Conv2D, Batch Normalization, MaxPool, Dropout, Flatten, and Dense layers. The model consists of a series of Conv2D layers with filter sizes of 32, 48, 64, 128, 256, and 512, consisting of (3,3) kernels. ReLU activation functions were used for all of the layers. A series of batch normalization layers and maxpool2D layers have been implemented between the Conv2D layers. Dropout layers are used to minimize overfitting. Finally, the network incorporates a Flatten layer to transform the multi-dimensional feature maps into a one-dimensional vector, followed by two densely connected layers to perform classification or regression tasks as required.

Model-2 and Model-3 use the same architecture with slight changes in the batch sizes during training. Similar to Model-1, Model-2, and Model- 3 combine Conv2D, Batch Normalization, MaxPool, Dropout, Flatten, and Dense layers. The filter sizes used for these models are 32, 6, 128, 256, and 512 with a (3,3) kernel shape. Several dropout layers with different

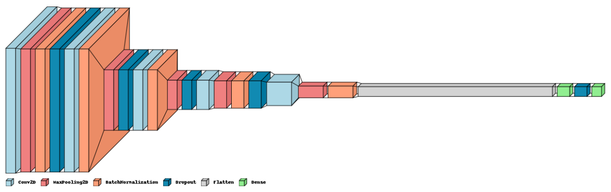


FIGURE 21. Illustration of the proposed CNN architecture for model 2 & model 3.

parameters are used in both of the models. However, two callback functions, ReduceLROnPlateau and EarlyStopping, are used during training to reduce the learning rate gradually and stop the training if no further improvements are monitored to minimize overfitting. Both models have several Conv2D layers. The equation for each convolutional layer can be written as follows:

$$Y_{i,j,k} = b_k + \sum_{m=0}^{M-1} \sum_{p=0}^{P-1} \sum_{q=0}^{Q-1} X_{i+p,j+q,m,k} \times W_{p,q,m,k}$$

Here $Y_{i,j,k}$ is the output location of the k-th feature map and $X_{i+p,j+q,m,k}$ is the input at location $(i + p, j + q, m)$ of m-th input depth. Each filter W acts as a feature detector, responding to specific patterns or features in the input image.

The Maxpool layer could be defined as follows

$$Y_{i,j,k} = \max_{p=0}^{P-1} \max_{q=0}^{Q-1} X_{i\times S+p, j\times S+q,k}$$

$Y_{i,j,k}$ is the output at location (i,j) of the k-th feature map, $X_{i\times S+p,j\times S+q,k}$ is the input at location (i×S+p,j×S+q) of the k-th feature map, with S being the stride.

Figure 22 contains the feature map visualization of some important layers of our model 1. After the Conv2D-(0) layer, applying Batch Normalization enhances the training process of the neural network by normalizing the activations of each layer. The visualization for the BN-(0) clearly shows how the model benefits from the normalization. A neuron with a BN filter could be defined as:

$$z = g(w, x) \tag{4}$$

$$z^N = \left(\frac{z - m_z}{s_z}\right) \cdot \gamma + \beta \tag{5}$$

$$a = f(z^N) \tag{6}$$

Here neuron's output mean and standard deviation is m_z & s_z, output of the BatchNormalization is z^N. Figure 23 and 24 also provide a very good visual of how BatchNormalization can normalize the activation of each layer.

TABLE 2. Model-2 & Model-3 summary.

Layer (type)	Output Shape	Param #
conv2d	(None, 64, 64, 32)	896
max_pooling2d	(None, 63, 63, 32)	0
batch_normalization	(None, 63, 63, 32)	128
dropout	(None, 63, 63, 32)	0
conv2d_1	(None, 63, 63, 64)	18496
batch_normalization_1	(None, 63, 63, 64)	256
max_pooling2d_1	(None, 31, 31, 64)	0
dropout_1	(None, 31, 31, 64)	0
conv2d_2	(None, 31, 31, 128)	73856
batch_normalization_2	(None, 31, 31, 128)	512
max_pooling2d_2	(None, 15, 15, 128)	0
dropout_2	(None, 15, 15, 128)	0
conv2d_3	(None, 15, 15, 256)	295168
max_pooling2d_3	(None, 14, 14, 256)	0
batch_normalization_3	(None, 14, 14, 256)	1024
dropout_3	(None, 14, 14, 256)	0
conv2d_4	(None, 12, 12, 512)	1180160
max_pooling2d_4	(None, 6, 6, 512)	0
batch_normalization_4	(None, 6, 6, 512)	2048
flatten	(None, 18432)	0
dense	(None, 256)	4718848
dropout_4	(None, 256)	0
dense_1	(None, 38)	9766
Total params		6,301,158
Trainable params		6,299,174
Non-trainable params		1,984

V. PRE-TRAINED NEURAL NETWORK MODELS

The architecture of VGG19 consists of 16 convolutional layers and 3 fully connected layers, for a total of 19 layers. The model has 144,000,000 parameters, a significantly large number of parameters.

The main idea behind Inception V3 is to incorporate different sizes of convolutions (1 × 1, 3 × 3, and 5 × 5) and pooling operations, enabling the network to extract features at various levels of abstraction. The Inception V3 model consists of about 25,000,000 parameters. The key idea behind DenseNet is that it connects each layer to every other layer in a feed-forward manner. The architecture of DenseNet consists of multiple dense blocks, where each thick block contains a series of densely connected layers.

This model has around 10,400,000 parameters. This hypothesis of Xception is built upon the idea behind the Inception architecture and takes it to an extreme level. Hence, it is called Xception, which stands for "Extreme Inception." The main feature extraction component of the model consists of 36 convolutional layers. These convolutional layers are organized into 14 modules. Around 22,800,000 parameters are present in this model. ResNet50 is a 50-layer deep convolutional neural network with an input size of 224 by 224. It can classify images into 1000 object categories. The model contains Bottleneck residual blocks that reduce the

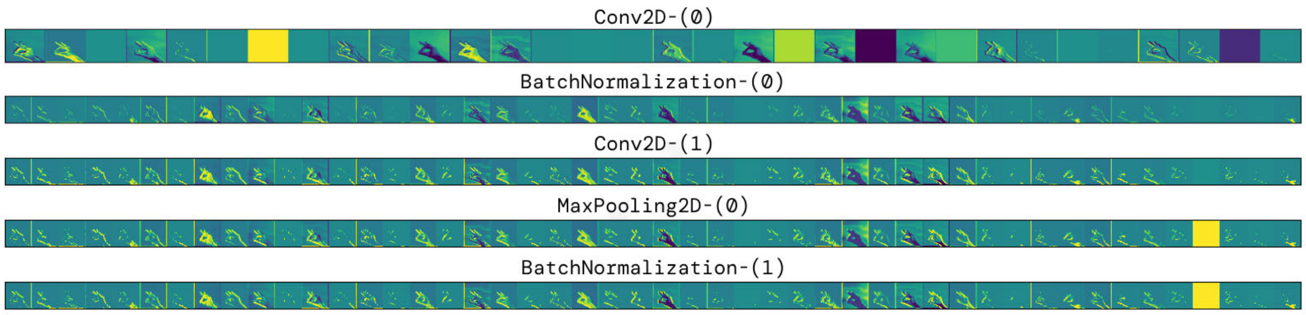


FIGURE 22. Feature Map Visualization on Model-1.

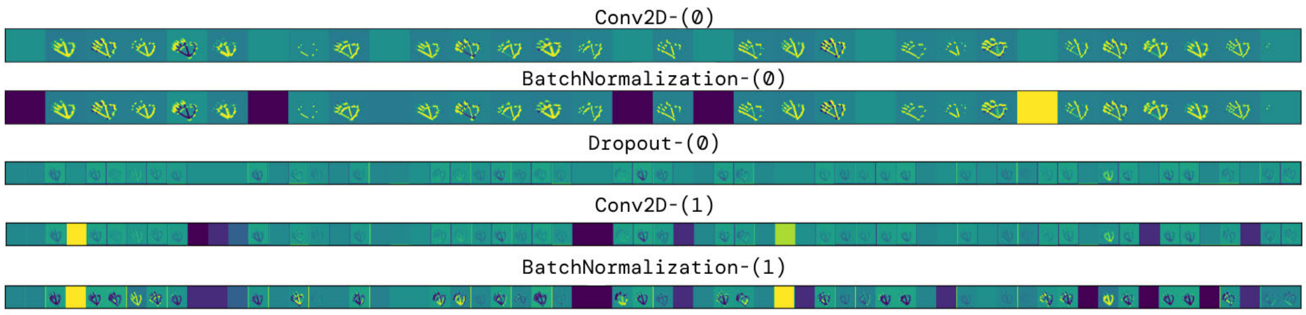


FIGURE 23. Feature map visualization on Model-2.

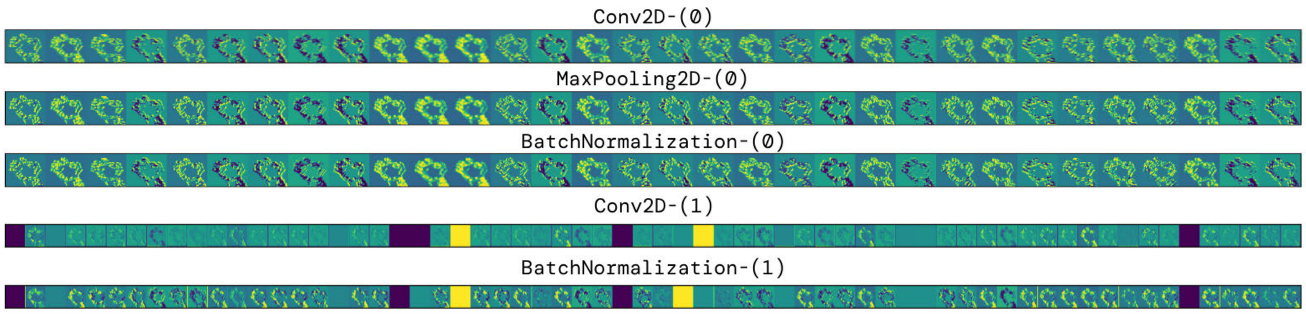


FIGURE 24. Feature map visualization on Model-3.

number of parameters and matrix multiplication, resulting in much faster training of individual layers.

The figures 22, 23, 24, 25 and 26 is a compilation of the training curves on the same dataset that we used in the study.

VI. EXPERIMENT RESULTS

A. RESULTS

A study comparing the proposed models with the five other pre-trained models was conducted, which included predefined state-of-the-art models such as VGG19, Inception V3, DenseNet, Xception, and ResNet50. The proposed models achieved high evaluation metrics scores compared to most predefined models, with the ResNet50 being the only exception by beating model 2. However, such an outcome was expected as model 2 works on a different modality than Resnet50 and has a significantly smaller architecture compared to ResNet50, with a lower dimensional input size. This experiment incorporates evaluation metrics such as accuracy, precision, F1 score, and loss to evaluate the models' performance [30]. The accuracy of a model measures how often the model correctly predicts the expected outcome. It is the fraction of accurate predictions made by the model relative to the total number of predictions. Precision is used to determine whether a percentage of the identifications made by the mode is accurate. To assess precision, we take the sum of all predicted positive outcomes known as True Positives (TP) and divide it by the sum of the total number of predicted positives (TP + FP). Hence, we use the following formula:

$$Precision = \frac{TP}{TP + FP}$$

The Recall metric measures how accurately a model can identify positive examples from a dataset. It is derived by dividing the count of true positives (TP) by the sum of true

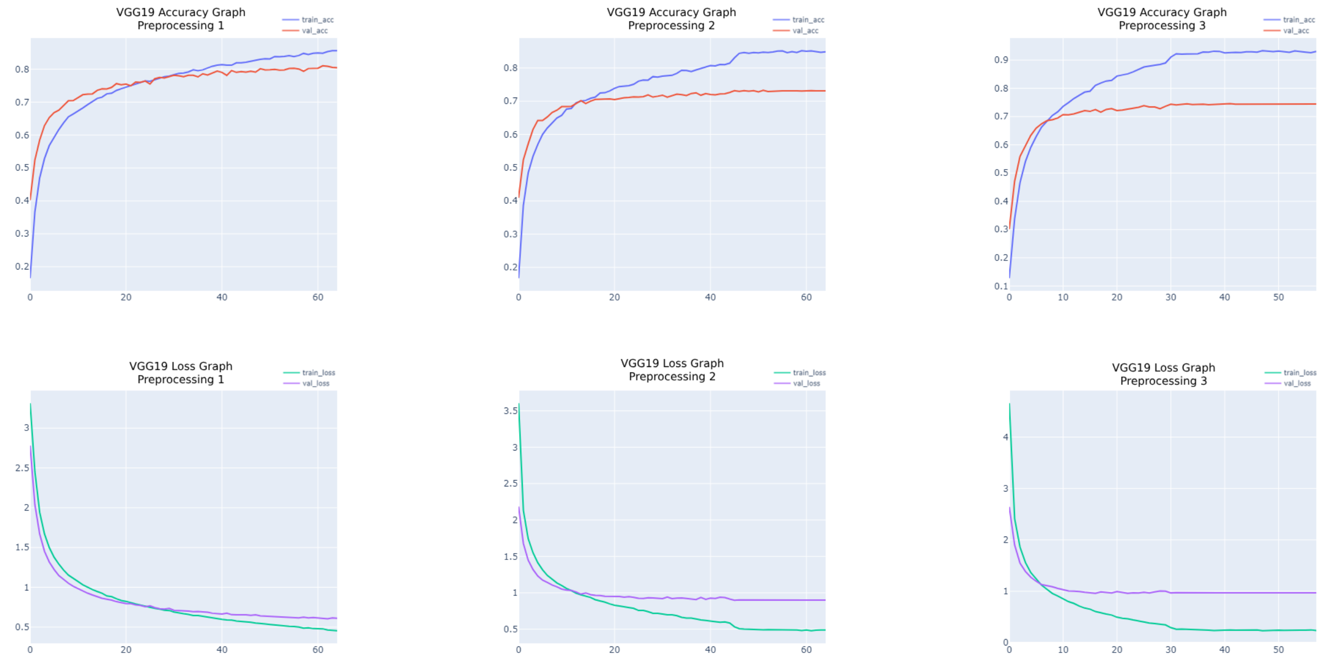


FIGURE 25. VGG19 Learning Curves for Different Preprocessing Methods.

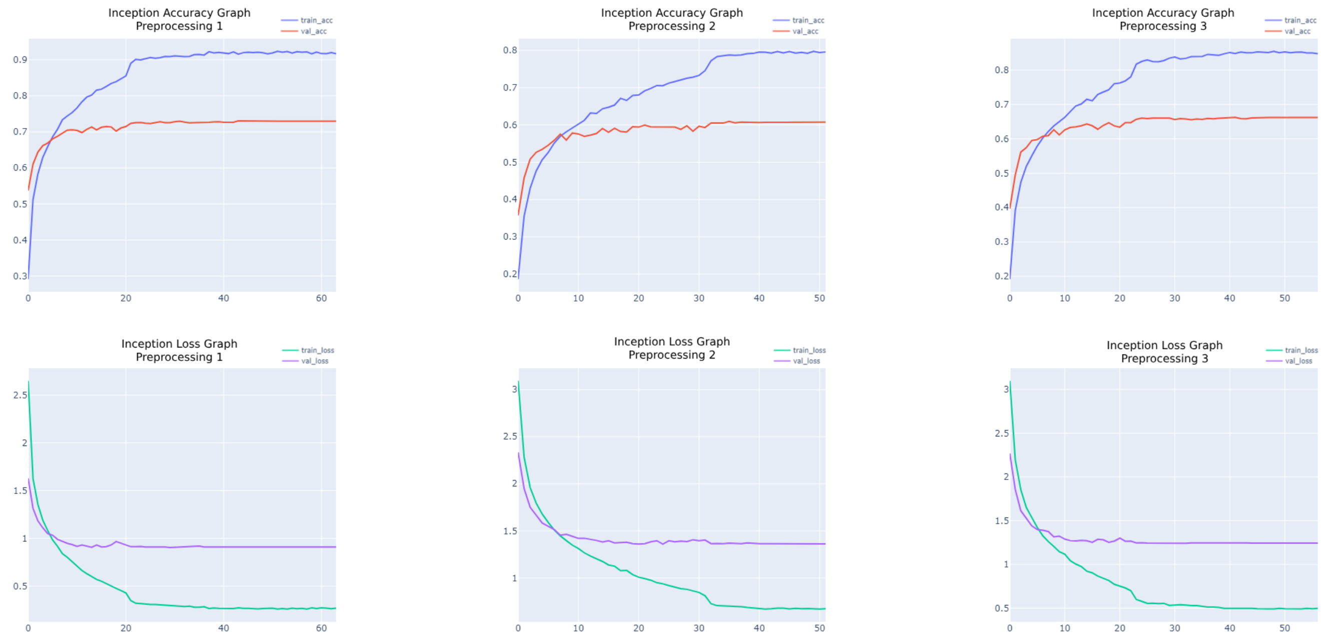


FIGURE 26. InceptionV3 learning curves for different preprocessing methods.

positives and false negatives (FN). The formula for the recall is:

$$Recall = \frac{TP}{TP + FN}$$

Finally, the F1 score is used to determine the overall performance of the models. It is calculated by taking the harmonic mean of the precision and recall scores. We use the following formula to select it:

$$F_1 = 2 \times \frac{precision \times recall}{precision + recall}$$

A comparative analysis with existing baseline or benchmark pre-trained models in the field was conducted. The evaluation metrics considered for this analysis included

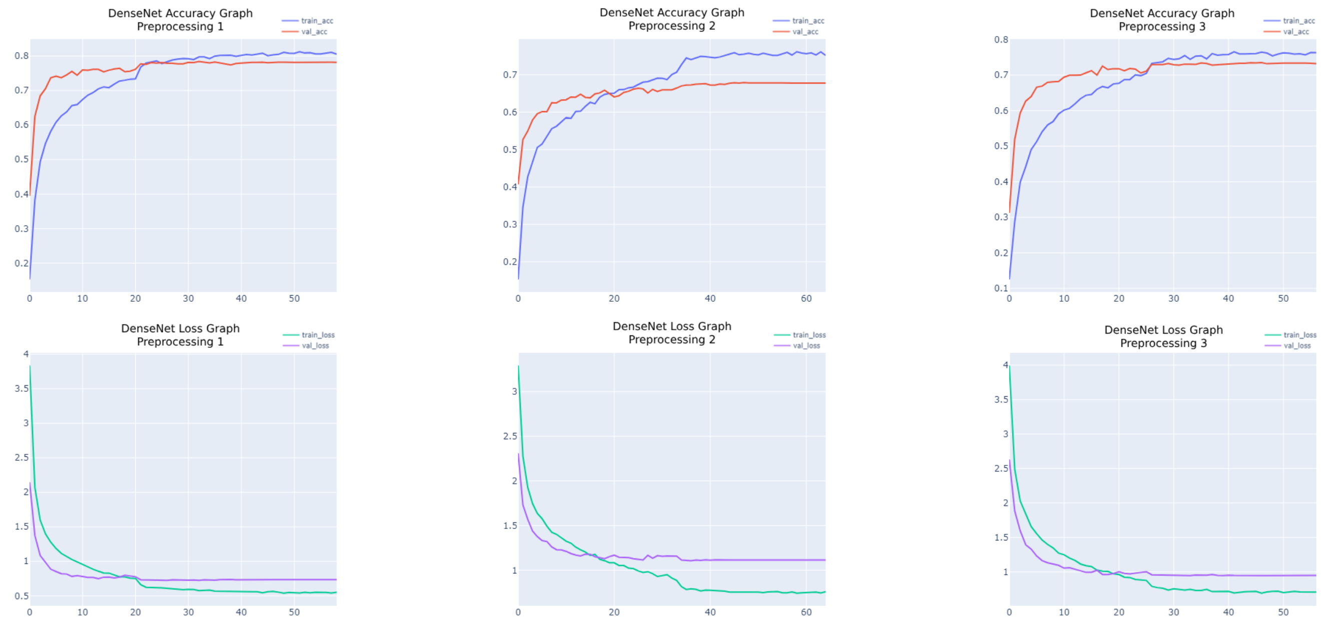


FIGURE 27. DenseNet learning curves for different preprocessing methods.

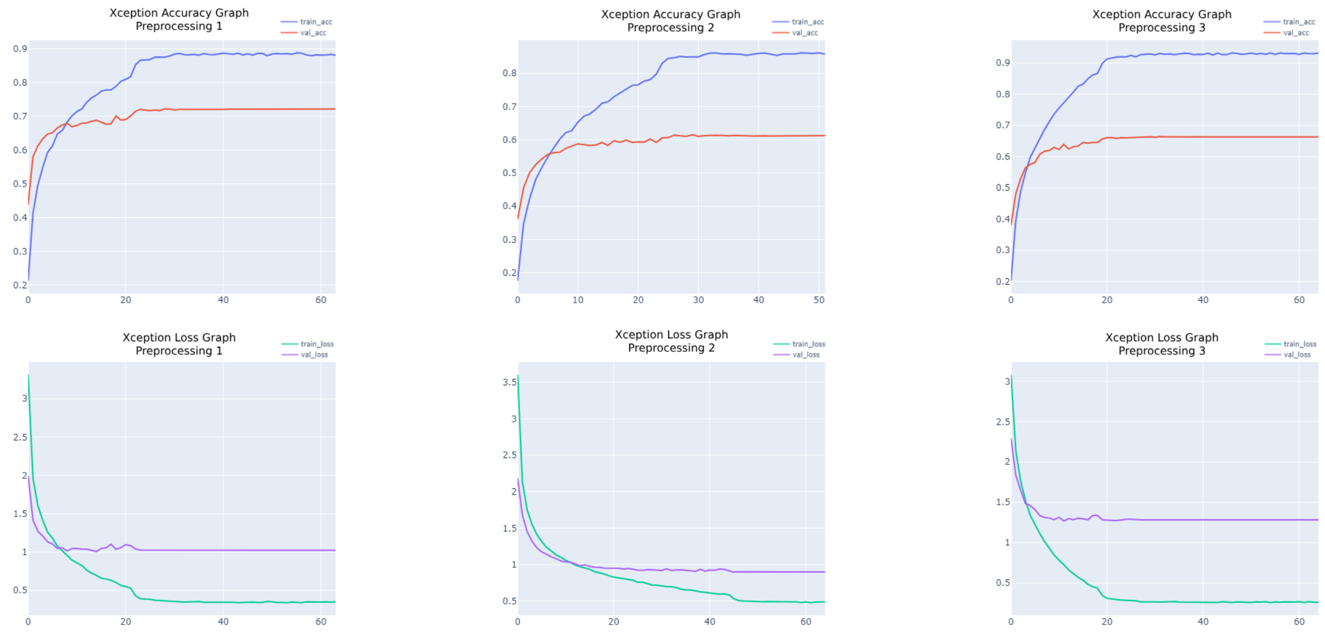


FIGURE 28. Xception learning curves for different preprocessing methods.

accuracy, precision, recall, and F1-score. The proposed models significantly improved all the evaluated metrics, outperforming the existing pre-trained models. Specifically, proposed model 1 achieved an accuracy of 97.65% on the testing set, surpassing the highest reported accuracy of the previous models. This substantial performance enhancement highlights the proposed model's effectiveness in addressing the problem.

The graphs illustrating the accuracy and loss of the proposed models are depicted in Figure 30.

B. CLASSIFICATION REPORT

The classification reports for all three models are provided below. This report helps to figure out the weaknesses of the models. It suggests that out of all the models, model 2 had the lowest precision with the lowest individual precision for

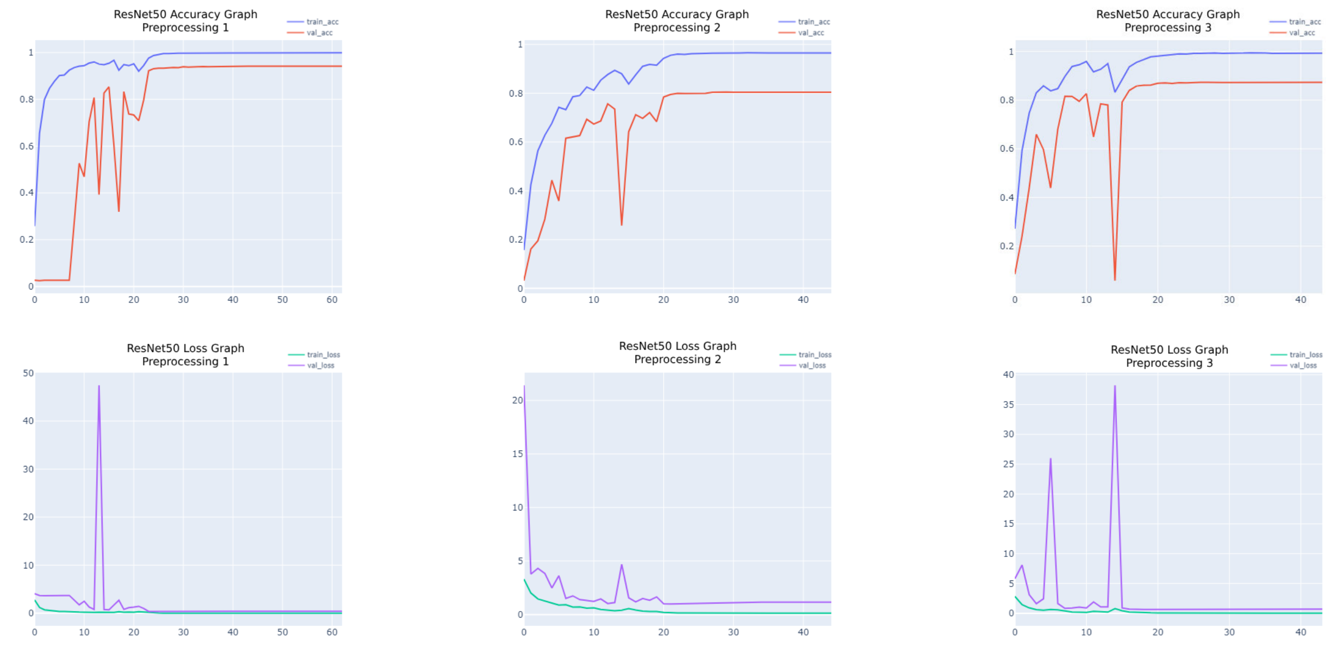
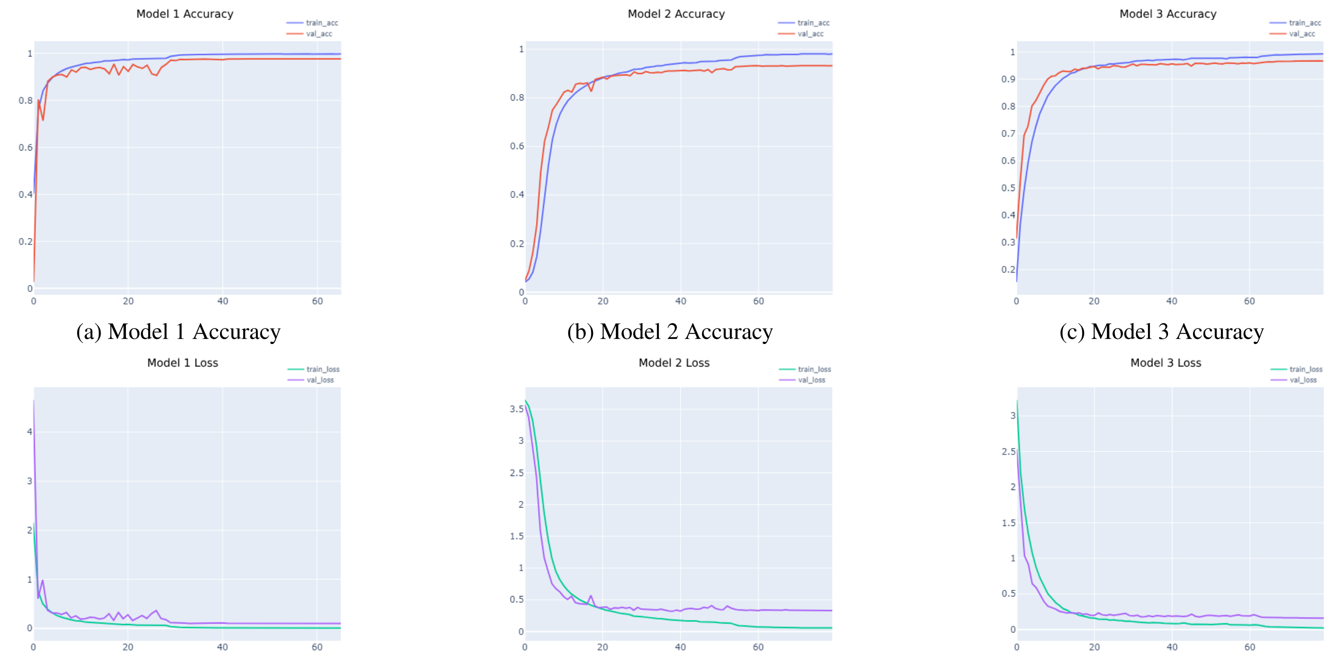


FIGURE 29. ResNet50 learning curves for different preprocessing methods.



(a) Model 1 Accuracy (b) Model 2 Accuracy (c) Model 3 Accuracy

(d) Model 1 Loss (e) Model 2 Loss (f) Model 3 Loss

FIGURE 30. Proposed models learning curves for different preprocessing methods.

a single class standing at 0.83 for classes 7 and 14. However, in some cases, model 2 has the best precision, beating both model 1 and model 3.

C. ENSEMBLE LEARNING

There are several ensemble learning techniques for combining different models [31]. The following segment describes some of these techniques and finds the best possible solution for our problem. A ground truth label, n models, and predictions from each model are needed for the ensemble model. The challenge is putting the best algorithm into practice so that the combination of predictions can give us matching ground truth. The proposed model 1, model 2, and model 3 are the models used for the ensemble's base learner.

TABLE 3. Evaluation metrics for training set.

Deep Learning Architecture	Pre Processing Technique	Acc.	Ep.	Recall	Prec.	Loss	F1 Score
VGG19	1 (128x128)	0.8561	65	0.9022	0.9041	0.4543	0.9018
VGG19	2 (128x128)	0.8489	65	0.9067	0.9233	0.4888	0.9116
VGG19	3 (128x128)	0.9307	58	0.9870	0.9870	0.2370	0.9870
Inception V3	1 (128x128)	0.9170	65	0.9876	0.9876	0.2703	0.9876
Inception V3	2 (128x128)	0.7959	52	0.8916	0.9070	0.6729	0.8961
Inception V3	3 (128x128)	0.8476	57	0.9453	0.9453	0.4973	0.9452
DenseNet	1 (128x128)	0.8052	65	0.9728	0.9729	0.5509	0.9727
DenseNet	2 (128x128)	0.7520	65	0.8881	0.9035	0.7594	0.8921
DenseNet	3 (128x128)	0.7630	57	0.9597	0.9599	0.7063	0.9597
ResNet50	1 (128x128)	0.9987	65	0.9999	0.9999	0.0700	0.9999
ResNet50	2 (128x128)	0.9652	45	0.9680	0.9857	0.1297	0.9737
ResNet50	3 (128x128)	0.9926	44	0.9998	0.9998	0.0269	0.9998
Xception	1 (128x128)	0.8808	65	0.9881	0.9882	0.3529	0.9881
Xception	2 (128x128)	0.8578	52	0.9446	0.9616	0.4861	0.9499
Xception	3 (128x128)	0.9317	65	0.9887	0.9888	0.2624	0.9887
Model 1	1 (128x128)	0.9965	70	0.9993	0.9992	0.0134	0.9992
Model 2-3	1 (128x128)	0.9966	51	0.9999	0.9999	0.0121	0.9999
Model 1	1 (64x64)	0.9977	66	0.9999	0.9999	0.0075	0.9999
Model 2	2 (64x64)	0.9811	80	0.9995	0.9995	0.0567	0.9995
Model 3	3 (64x64)	0.9930	80	0.9994	0.9994	0.0201	0.9994

TABLE 4. Evaluation metrics for validation set.

Deep Learning Architecture	Pre Processing Technique	Acc.	Ep.	Recall	Prec.	Loss	F1 Score
VGG19	1 (128x128)	0.8045	65	0.8045	0.8083	0.6119	0.8037
VGG19	2 (128x128)	0.7313	65	0.7313	0.7437	0.8994	0.7342
VGG19	3 (128x128)	0.7445	58	0.7444	0.7453	0.9704	0.7437
Inception V3	1 (128x128)	0.7297	65	0.7300	0.7307	0.9105	0.7293
Inception V3	2 (128x128)	0.6079	52	0.6081	0.6189	1.3628	0.6099
Inception V3	3 (128x128)	0.6621	57	0.6621	0.6643	1.2440	0.6620
DenseNet	1 (128x128)	0.7816	65	0.7816	0.7846	0.7342	0.7817
DenseNet	2 (128x128)	0.6779	65	0.6779	0.6890	1.1143	0.6799
DenseNet	3 (128x128)	0.7321	57	0.7321	0.7365	0.9484	0.7327
ResNet50	1 (128x128)	0.9416	65	0.9416	0.9421	0.3763	0.9417
ResNet50	2 (128x128)	0.8039	45	0.8039	0.8196	1.1585	0.8088
ResNet50	3 (128x128)	0.8729	44	0.8729	0.8739	0.7086	0.8728
Xception	1 (128x128)	0.7213	65	0.7213	0.7245	1.0248	0.7219
Xception	2 (128x128)	0.6126	52	0.6126	0.6262	1.3850	0.6164
Xception	3 (128x128)	0.6629	65	0.6629	0.6643	1.2841	0.6618
Model 1	1 (128x128)	0.9002	70	0.9002	0.9023	0.5285	0.9005
Model 2-3	1 (128x128)	0.8756	51	0.8756	0.8765	0.6012	0.8754
Model 1	1 (64x64)	0.9765	66	0.9766	0.9766	0.1005	0.9765
Model 2	2 (64x64)	0.9330	80	0.9330	0.9334	0.3309	0.9329
Model 3	3 (64x64)	0.9666	80	0.9666	0.9667	0.1591	0.9666

For the ensemble testing, the remaining 1520 testing data are used.

1) UNWEIGHTED AVERAGING ENSEMBLE TECHNIQUE

The ensemble unweighted averaging technique is one of the easiest yet effective ensemble techniques. We use the following equation to get the averaging probability of each class.

$$e.p(Class\ X) = \frac{1}{N}\sum_{i=1}^{N} \text{Model}_i(Probability\ of\ \ Class\ X)$$

$Model_i$(Probability of Class c) represents the predicted probability of class c for the i-th base model, and N represents the count of models used in the ensemble technique. The prediction corresponding to the class with the highest probability is considered final.

TABLE 5. Classification report Model 1.

Index	precision	recall	f1-score	support
0	1.00	0.97	0.98	202
1	0.98	0.98	0.98	200
2	0.99	0.99	0.99	200
3	0.93	0.90	0.92	216
4	0.99	0.99	0.99	165
5	0.98	1.00	0.99	206
6	0.98	0.99	0.99	201
7	0.97	0.99	0.98	204
8	0.98	0.98	0.98	214
9	0.98	0.98	0.98	191
10	0.99	0.99	0.99	220
11	0.99	0.96	0.98	196
12	0.99	0.98	0.99	178
13	0.99	0.96	0.97	194
14	0.98	0.97	0.98	200
15	0.99	0.97	0.98	180
16	0.99	1.00	0.99	246
17	0.99	1.00	1.00	221
18	0.95	0.97	0.96	193
19	0.98	0.97	0.98	214
20	0.98	1.00	0.99	205
21	1.00	0.98	0.99	206
22	0.94	0.94	0.94	232
23	0.95	0.97	0.96	197
24	0.97	0.95	0.96	217
25	0.98	0.99	0.98	192
26	0.98	0.97	0.97	206
27	0.96	0.98	0.97	192
28	0.99	0.98	0.99	194
29	0.99	0.99	0.99	201
30	0.97	1.00	0.99	212
31	0.98	0.96	0.97	181
32	0.95	0.99	0.97	201
33	0.96	0.97	0.96	210
34	0.99	0.99	0.99	213
35	0.96	0.96	0.96	221
36	0.98	0.98	0.98	200
37	1.00	1.00	1.00	222
accuracy	None	None	0.98	7743
macro avg	0.98	0.98	0.98	7743
weighted avg	0.98	0.98	0.98	7743

TABLE 6. Classification report Model 2.

Index	precision	recall	f1-score	support
0	1.00	0.97	0.98	202
1	0.98	0.98	0.98	200
2	0.99	0.99	0.99	200
3	0.93	0.90	0.92	216
4	0.99	0.99	0.99	165
5	0.98	1.00	0.99	206
6	0.98	0.99	0.99	201
7	0.97	0.99	0.98	204
8	0.98	0.98	0.98	214
9	0.98	0.98	0.98	191
10	0.99	0.99	0.99	220
11	0.99	0.96	0.98	196
12	0.99	0.98	0.99	178
13	0.99	0.96	0.97	194
14	0.98	0.97	0.98	200
15	0.99	0.97	0.98	180
16	0.99	1.00	0.99	246
17	0.99	1.00	1.00	221
18	0.95	0.97	0.96	193
19	0.98	0.97	0.98	214
20	0.98	1.00	0.99	205
21	1.00	0.98	0.99	206
22	0.94	0.94	0.94	232
23	0.95	0.97	0.96	197
24	0.97	0.95	0.96	217
25	0.98	0.99	0.98	192
26	0.98	0.97	0.97	206
27	0.96	0.98	0.97	192
28	0.99	0.98	0.99	194
29	0.99	0.99	0.99	201
30	0.97	1.00	0.99	212
31	0.98	0.96	0.97	181
32	0.95	0.99	0.97	201
33	0.96	0.97	0.96	210
34	0.99	0.99	0.99	213
35	0.96	0.96	0.96	221
36	0.98	0.98	0.98	200
37	1.00	1.00	1.00	222
accuracy	None	None	0.98	7743
macro avg	0.98	0.98	0.98	7743
weighted avg	0.98	0.98	0.98	7743

TABLE 7. Classification report Model 3.

Index	precision	recall	f1-score	support
0	0.98	0.97	0.97	221
1	0.96	0.97	0.96	219
2	0.98	0.98	0.98	203
3	0.96	0.95	0.95	213
4	0.95	0.97	0.96	156
5	0.96	0.96	0.96	207
6	0.96	0.96	0.96	210
7	0.96	0.95	0.96	205
8	0.98	0.97	0.97	209
9	0.97	0.98	0.98	201
10	0.96	0.97	0.97	205
11	0.99	0.96	0.98	204
12	0.94	0.97	0.95	174
13	0.98	0.95	0.97	212
14	0.97	0.97	0.97	209
15	0.98	0.95	0.96	202
16	0.97	0.96	0.96	221
17	0.95	0.94	0.95	193
18	0.95	0.98	0.96	204
19	0.96	0.93	0.95	195
20	0.98	0.99	0.98	216
21	0.97	0.96	0.97	223
22	0.94	0.94	0.94	217
23	0.93	0.96	0.95	207
24	0.95	0.96	0.96	196
25	0.99	0.99	0.99	210
26	0.96	0.94	0.95	206
27	1.00	0.99	0.99	206
28	0.99	0.98	0.99	186
29	0.99	0.99	0.99	216
30	0.96	0.98	0.97	218
31	0.99	0.97	0.98	201
32	0.98	0.99	0.98	205
33	0.96	0.95	0.95	204
34	0.98	0.98	0.98	175
35	0.97	0.97	0.97	182
36	0.98	0.98	0.98	218
37	0.93	0.97	0.95	194
accuracy	None	None	0.97	7743
macro avg	0.97	0.97	0.97	7743
weighted avg	0.97	0.97	0.97	7743

2) UNWEIGHTED VOTING ENSEMBLE TECHNIQUE

In the case of the unweighted voting ensemble, the final prediction is based on the number of votes in each class. We used the following formula for the final prediction.

$$\begin{aligned}&Ensemble\ Prediction\\&\quad = mode(Class\ Predictions\ from\ Base\ Models)\end{aligned}$$

Here, mode(Class Predictions from Base Models) represents the most frequently occurring class among the predictions of the base models. This process comes with the drawback of taking a random prediction if no one highest voted class is found.

3) STATIC-WEIGHT AVERAGING ENSEMBLE TECHNIQUE

This ensemble technique combines predictions from all base models and considers the weight of each base model. We used the average precisions from the training report as the weight of the models. We calculated the probability of each class using the following formula:

$$\begin{aligned}&e.p(Class\ X)\\&= \frac{\sum_{i=1}^{N} Model_i(Probability\ of\ \ Class\ X) * AveragePrecision_i}{\sum_{i=1}^{N} AveragePrecision_i}\end{aligned}$$

N is the total number of base models in the ensemble. $Model_i$(Probability of Class X) represents the predicted probability of class X for the i-th base model. Average $Precision_i$ represents the average precision of the i-th base model.

4) STATIC-WEIGHT VOTING ENSEMBLE TECHNIQUE

The static weight voting technique tries to solve the drawback of the unweighted voting technique. The formula is the same as the unweighted voting technique, but when there is a tie in the voting process, the prediction from the highest-weighted model is chosen as the final prediction. Similarly, the weight is considered as the average precision value from the base model's training report.

5) DYNAMIC-WEIGHT AVERAGING ENSEMBLE TECHNIQUE

The process of the Dynamic-weight averaging technique is similar to the static-weight averaging ensemble technique, but as the name suggests, the weight is dynamic. For Model-1, Model-2 and Model-3 we have the classification report table 5, 6 and 7. After getting the prediction from each model, the precision value of that specific class from the specific model's validation report is taken and used as a weight. We used the following formula to calculate the ensemble probability of each class.

$$e.p(Class\ X) = \frac{\sum_{i=1}^{N} Model_i(Probability\ of\ \ Class\ X) * Precision_{\ i,c}}{\sum_{i=1}^{N} Precision_{\ i,c}}$$

Here, C is the total number of classes, $Precision_{\ i,c}$ represents the precision of class c from the i-th base model.

6) ENSEMBLE STACKING TECHNIQUE

To achieve ensemble stacking, the testing subsets are split into training and testing sets. Predictions are made on the training sets using the base models. These predictions are multiplied by respective weights assigned to each model. The weighted predictions are combined to create the training dataset for the meta-learner. A meta-learner is trained on this new training dataset. The new testing subsets are used to evaluate the model. Accuracy scores are calculated for each meta-learner. Finally, the accuracy scores of the meta-learner are printed to assess the performance. For the meta-learner algorithms, we used classifiers such as Random Forest Classifier, Support vector machine, K-Nearest Neighbors, and Logistic Regression.

D. ENSEMBLE MODEL CLASSIFICATION REPORT

The following report is based on the 1,520 testing samples we split earlier. Different ensemble techniques [32] are compared alongside the regular independent models. It is noticed that Resnet50 outperforms all the predefined models by a significant and convincing margin. On the other hand, a comparison between multiple ensemble techniques is studied, where static-weight averaging ensemble network outperformed everyone. We hypothesized that dynamic-weight-based averaging would provide the highest accuracy. How- ever, in the empirical analysis, it is tested to be false as it came in third place only after the static-weight and unweighted averaging techniques. The voting and averaging models are similar in principle, but taking confidence into account were able to make a marginal difference in the total outcome. The ensemble stacking models worked well too, but they could not outperform the simpler decision-making algorithms.

TABLE 8. Model accuracy.

Model	Accuracy
Inception (Pre-processing 1)	0.7118
Inception (Pre-processing 2)	0.5967
Inception (Pre-processing 3)	0.6447
DenseNet (Pre-processing 1)	0.7770
DenseNet (Pre-processing 2)	0.6684
DenseNet (Pre-processing 3)	0.7382
Xception (Pre-processing 1)	0.6855
Xception (Pre-processing 2)	0.6053
Xception (Pre-processing 3)	0.6309
ResNet50 (Pre-processing 1)	0.9349
ResNet50 (Pre-processing 2)	0.7934
ResNet50 (Pre-processing 3)	0.8803
VGG19 (Pre-processing 1)	0.7822
VGG19 (Pre-processing 2)	0.7013
VGG19 (Pre-processing 3)	0.7230
Model-1 (128x128)	0.9118
Model-(2-3) (128x128)	0.8862
Unweighted Averaging Ensemble Model (64x64)	0.9500
Unweighted Voting Ensemble Model (64x64)	0.9329
Static-Weight Averaging Ensemble Model (64x64)	0.9513
Static-Weight Voting Ensemble Model (64x64)	0.9454
Dynamic-Weight Averaging Ensemble Model (64x64)	0.9480
Ensemble Stack RandomForest Meta-learner (64x64)	0.9137
Ensemble Stack SVM Meta-learner (64x64)	0.9194
Ensemble Stack KNN Meta-learner (64x64)	0.9235
Ensemble Stack Logistic Regression Meta-learner (64x64)	0.9367

E. COMPARATIVE ANALYSIS AND DISCUSSION

In the following Table 9 we compared our results with the existing study on BdSL. Hossen et al. [34] in their study augmented the dataset with flipping. Rafi et al. [24] used several augmentation techniques including horizontal flip, and vertical flip. However, Horizontal or Vertical flip as a preprocessing step is not always suitable for hand sign recognition as it may alter the meaning of the hand sign which will lead to incorrect predictions. On the other hand, Islam et al. manually cropped the samples to reduce the noise from the images, followed by the samples being resized and converted to grayscale. Manually cropping the images helped the model train better, but it is not suitable for real-life implementation, so a technique to find the ROI needed to be developed. Tasmere et al. [39] preprocessed their images with Gaussian mixture-based background algorithm, Gaussian blur, Binary image conversion, and normalization. The preprocessing techniques used in their study provided promising results but a significant amount of data loss could occur due to binary image conversion (e.g. no way to trace finger position when it is inside the palm region). Abedin et al. [33] proposed an image network and pose estimation network approach where for the preprocessing technique, sample images were normalized and resized. Instead of providing separate predictions for each network, concatenated BdSL provided a single result for such reason failure to find the hand coordinate for pose estimations can create further complexity.

TABLE 9. Comparison between the existing BSL system and proposed approach.

Reference	Year	Sample Size and Dataset	Model	Testing Accuracy
Hossen et al. [34]	2017	1,147 [35]	Deep CNN	84.68%
Islam et al. [40]	2018	1,800 [40]	Custom CNN	94.74 %
Rafi et al. [24]	2019	12,581 [24]	Custom VGG19	89.6%
Tasmere et al. [39]	2020	1,674 [39]	Custom CNN	97.66%
Abedin et al. [33]	2021	12,581 [24]	Concatenated BdSL Network	91.51%
Nihal et al. [7]	2021	12,581 [24]	DenseNet-201	93.8%
Basnin et al. [36]	2021	12,960 [36]	Integrated CNN–LSTM	88.5%
Das, Sunanda, et al. [41]	2022	1,075 [41]	Hybrid (VGG16 + RF)	91.67%
Proposed Model	**2024**	**12,581** [24]	**Static-Weight Averaging Ensemble**	**95.13%**

In our proposed method, we were very careful in the preprocessing steps to mitigate the discussed concerns from previous studies. For the starter we did not apply any flipping during the augmentation of our dataset, we kept the rotation to a minimum (±5°). Cropping the images can significantly boost the prediction accuracy, so we used MediaPipe's hand-landmark model to find the ROI and clip the image with a certain level of padding. It is now possible to feed our model with cropped images during the real-life implementation. To avoid data loss during binary image conversion, we proposed a canny edge detection algorithm, which helped us retain most information, without losing data near the palm region. Instead of using a pose estimation network to feed a single model, we propose an individual model to provide prediction with pose-estimated data. In any case, if the pose estimation fails the other two models still provide a prediction rather than feeding a single model with incorrect data. Furthermore, we proposed an ensemble comprising multiple models rather than a solitary model, each of the models is based on different data, and provides more credibility in the prediction. Thus, the results presented in Table 9 demonstrate that the proposed model surpasses the performance of all previous approaches, except for the work proposed by Tasmere et al. [39] that ultimately uses a much smaller dataset. The performance is also better compared to the works that use the dataset from [24]; therefore, the empirical analysis proves the approach's effectiveness.

VII. CONCLUSION AND FUTURE WORK

This paper presented a multimodal approach to Bangla sign language recognition. The extensive preprocessing techniques convert the original image to different modalities to train CNN models. The first preprocessing technique normalizes the base images for consistent scaling, reducing the impact of pixel intensity variations so that the model becomes less sensitive to illumination changes. The second preprocessing technique provides a different modality of the base images by completely removing all the background information and only considering the hand landmarks coordinates drawn using MediaPipe. The approach helps the model learn more relevant information by isolating obstacles. Similarly, the final preprocessing technique provides a different modality of the original images using only the edge information. Using the hand coordinates from Mediapipe. It crops out the unnecessary noise from the background and finds the best possible case for edge detection on the ROI. Then, nine different ensemble methods were tested to find the best possible algorithm suitable for this task. All these aspects help classify hand patterns while learning their meaningful reasoning. Furthermore, beyond BdSLR, our methodology for the preprocessing step can be applied to industries such as human-computer interaction, virtual reality or robotics to enhance the user experience in gesture recognition and enable intuitive interactions between humans and machines. The approach is based on static hand signs, but the preprocessing techniques can be used for a continuous sign language recognition system. However, it is still uncertain how it will react to temporal information. While our study provides promising results, there are certain limitations. The approach depends highly on the preprocessing techniques and may require more time on preprocessing steps in real-life implementation. Model 2 depends entirely on the MediaPipe library, so it may create a single point of failure for this model, even though accurate prediction from models 1 and 3 can overcome the problem entirely. We hypothesized that dynamic-weight averaging would provide the best result between the ensemble algorithms, but it came with the third-highest accuracy but very close to the best accuracy. Direction for future work includes following our approach on different datasets to accept or reject some of the hypotheses, i.e., testing the results for the dynamic-weight averaging technique, hyperparameter tuning our proposed CNN model to determine the overall accuracy of our network further, Color- coding each finger differently in the second preprocessing technique so it can provide more meaningful images for the model to learn and test if the ac- curacy can improve, and test the proposed preprocessing techniques on a continuous sign language recognition system. Such studies are also expected to highlight the outcome of our hypotheses and evaluate the approach further.

REFERENCES

[1] WHO. (Feb. 2023). *Deafness and Hearing Loss*. [Online]. Available: https://www.who.int/news-room/fact-sheets/detail/deafness-and-hearing

[2] U. Nations. (Jun. 2017). *World Population Projected to Reach 9.8 Billion in 2050, and 11.2 Billion in 2100—UN DESA—United Nations Department of Economic and Social Affairs*. [Online]. Available: https://www.un.org/development/desa/en/news/population/world-population-prospects-2017.html

[3] K. H. Tarafder, N. Akhtar, M. M. Zaman, M. A. Rasel, M. R. Bhuiyan, and P. G. Datta, "Disabling hearing impairment in the Bangladeshi population," *J. Laryngol. Otol.*, vol. 129, no. 2, pp. 126–135, Feb. 2015. [Online]. Available: https://api.semanticscholar.org/CorpusID:25333292

[4] R. J. Johnson and J. E. Johnson, "Distinction between west Bengal sign language and Indian sign language based on statistical assessment," *Sign Lang. Stud.*, vol. 16, no. 4, pp. 473–499, 2016.

[5] A. Palshikar, "What distinguishes binary from multi-class intrusion detection systems: Observations from experiments," *Int. J. Inf. Manag. Data Insights*, vol. 2, no. 2, Nov. 2022, Art. no. 100125, doi: 10.1016/j.jjimei.2022.100125.

[6] P. D. Moral, S. Nowaczyk, and S. Pashami, "Why is multiclass classification hard?" *IEEE Access*, vol. 10, pp. 80448–80462, 2022, doi: 10.1109/ACCESS.2022.3192514.

[7] R. A. Nihal, S. Rahman, N. M. Broti, and S. A. Deowan, "Bangla sign alphabet recognition with zero-shot and transfer learning," *Pattern Recognit. Lett.*, vol. 150, pp. 84–93, Oct. 2021, doi: 10.1016/j.patrec.2021.06.020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167865521002269

[8] M. S. Islam, A. Joha, M. N. Hossain, S. Abdullah, I. Elwarfalli, and M. M. Hasan, "Word level Bangla sign language dataset for continuous BSL recognition," 2023, *arXiv:2302.11559*.

[9] A. Hasib, J. F. Eva, S. S. Khan, M. N. Khatun, A. Haque, N. Shahrin, R. Rahman, H. Murad, M. R. Islam, and M. R. Hussein, "BDSL 49: A comprehensive dataset of Bangla sign language," *Data Brief*, vol. 49, Aug. 2023, Art. no. 109329, doi: 10.1016/j.dib.2023.109329.

[10] S. Wang, K. Wang, T. Yang, Y. Li, and D. Fan, "Improved 3D-ResNet sign language recognition algorithm with enhanced hand features," *Sci. Rep.*, vol. 12, no. 1, p. 17812, Oct. 2022.

[11] W. C. Sleeman, R. Kapoor, and P. Ghosh, "Multimodal classification: Current landscape, taxonomy and future directions," *ACM Comput. Surv.*, vol. 55, no. 7, pp. 1–31, Dec. 2022, doi: 10.1145/3543848.

[12] A. Sen, T. K. Mishra, and R. Dash, "A novel hand gesture detection and recognition system based on ensemble-based convolutional neural network," *Multimedia Tools Appl.*, vol. 81, no. 28, pp. 40043–40066, 2022.

[13] F. B. Slimane and M. Bouguessa, "Context matters: Self-attention for sign language recognition," in *Proc. 25th Int. Conf. Pattern Recognit. (ICPR)*, Jan. 2021, pp. 7884–7891.

[14] P. Xie, Z. Cui, Y. Du, M. Zhao, J. Cui, B. Wang, and X. Hu, "Multi-scale local-temporal similarity fusion for continuous sign language recognition," *Pattern Recognit.*, vol. 136, Apr. 2023, Art. no. 109233, doi: 10.1016/j.patcog.2022.109233.

[15] S. Jiang, B. Sun, L. Wang, Y. Bai, K. Li, and Y. Fu, "Skeleton aware multi-modal sign language recognition," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, Jun. 2021, pp. 3408–3418, doi: 10.1109/CVPRW53098.2021.00380.

[16] M. S. Islalm, M. M. Rahman, M. H. Rahman, M. Arifuzzaman, R. Sassi, and M. Aktaruzzaman, "Recognition Bangla sign language using convolutional neural network," in *Proc. Int. Conf. Innov. Intell. Informat., Comput., Technol. (ICT)*, Sep. 2019, pp. 1–6.

[17] S. Ahmed, M. R. Islam, J. Hassan, M. U. Ahmed, B. J. Ferdosi, S. Saha, M. Shopon, "Hand sign to bangla speech: A deep learning in vision based system for recognizing hand sign digits and generating bangla speech," in *Proc. Int. Conf. Sustain. Comput. Sci., Technol. Manag. (SUSCOM)*. Jaipur, India: Amity Univ. Rajasthan, Feb. 2019. [Online]. Available: https://ssrn.com/abstract=3358187

[18] A. Wadhawan and P. Kumar, "Deep learning-based sign language recognition system for static signs," *Neural Comput. Appl.*, vol. 32, no. 12, pp. 7957–7968, Jan. 2020.

[19] S. Yang and Q. Zhu, "Video-based Chinese sign language recognition using convolutional neural network," in *Proc. IEEE 9th Int. Conf. Commun. Softw. Netw. (ICCSN)*, May 2017, pp. 929–934.

[20] B. Sundar and T. Bagyammal, "American sign language recognition for alphabets using MediaPipe and LSTM," *Proc. Comput. Sci.*, vol. 215, pp. 642–651, Jan. 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1877050922021378

[21] S. Ameen and S. Vadera, "A convolutional neural network to classify American sign language fingerspelling from depth and colour images," *Exp. Syst.*, vol. 34, no. 3, Jun. 2017, Art. no. e12197.

[22] B. Shi, A. M. D. Rio, J. Keane, D. Brentari, G. Shakhnarovich, and K. Livescu, "Fingerspelling recognition in the wild with iterative visual attention," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, Oct. 2019, pp. 5400–5409.

[23] M. A. Ganaie, M. Hu, A. K. Malik, M. Tanveer, and P. N. Suganthan, "Ensemble deep learning: A review," *Eng. Appl. Artif. Intell.*, vol. 115, Oct. 2022, Art. no. 105151, doi: 10.1016/j.engappai.2022.105151.

[24] A. M. Rafi, N. Nawal, N. S. N. Bayev, L. Nima, C. Shahnaz, and S. A. Fattah, "Image-based Bengali sign language alphabet recognition for deaf and dumb community," in *Proc. IEEE Global Humanitarian Technol. Conf. (GHTC)*, Oct. 2019, pp. 1–7.

[25] A. Luque, A. Carrasco, A. Martín, and A. de las Heras, "The impact of class imbalance in classification performance metrics based on the binary confusion matrix," *Pattern Recognit.*, vol. 91, pp. 216–231, Jul. 2019, doi: 10.1016/j.patcog.2019.02.023.

[26] X. Pei, Y. H. Zhao, L. Chen, Q. Guo, Z. Duan, Y. Pan, and H. Hou, "Robustness of machine learning to color, size change, normalization, and image enhancement on micrograph datasets with large sample differences," *Mater. Des.*, vol. 232, Aug. 2023, Art. no. 112086. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0264127523005014

[27] G. Sánchez-Brizuela, A. Cisnal, E. de la Fuente-López, J.-C. Fraile, and J. Pérez-Turiel, "Lightweight real-time hand segmentation leveraging MediaPipe landmark detection," *Virtual Reality*, vol. 27, no. 4, pp. 3125–3132, Dec. 2023.

[28] O. Elharrouss, Y. Hmamouche, A. K. Idrissi, B. El Khamlichi, and A. El Fallah-Seghrouchni, "Refined edge detection with cascaded and high-resolution convolutional network," *Pattern Recognit.*, vol. 138, Jun. 2023, Art. no. 109361, doi: 10.1016/j.patcog.2023.109361.

[29] J. Kunhoth, S. Al Maadeed, M. Saleh, and Y. Akbari, "CNN feature and classifier fusion on novel transformed image dataset for dysgraphia diagnosis in children," *Expert Syst. Appl.*, vol. 231, Nov. 2023, Art. no. 120740, doi: 10.1016/j.eswa.2023.120740.

[30] B. Wu, Z. Gu, W. Zhang, Q. Fu, M. Zeng, and A. Li, "Investigator accuracy: A center-weighted metric for evaluating the location accuracy of image segments in land cover classification," *Int. J. Appl. Earth Observ. Geoinf.*, vol. 122, Aug. 2023, Art. no. 103402. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1569843223002261

[31] A. Mohammed and R. Kora, "A comprehensive review on ensemble deep learning: Opportunities and challenges," *J. King Saud Univ.-Comput. Inf. Sci.*, vol. 35, no. 2, pp. 757–774, Feb. 2023, doi: 10.1016/j.jksuci.2023.01.014.

[32] S. Lin, H. Zheng, B. Han, Y. Li, C. Han, and W. Li, "Comparative performance of eight ensemble learning approaches for the development of models of slope stability prediction," *Acta Geotechnica*, vol. 17, no. 4, pp. 1477–1502, Apr. 2022.

[33] T. Abedin, K. S. S. Prottoy, A. Moshruba, and S. Bin Hakim, "Bangla sign language recognition using concatenated BdSL network," 2021, *arXiv:2107.11818*.

[34] M. A. Hossen, A. Govindaiah, S. Sultana, and A. Bhuiyan, "Bengali sign language recognition using deep convolutional neural network," in *Proc. Joint 7th Int. Conf. Informat., Electron. Vis. (ICIEV) 2nd Int. Conf. Imag., Vis. Pattern Recognit. (icIVPR)*, Jun. 2018, pp. 369–373.

[35] K. E. Aziz, A. Wadud, S. Sultana, M. A. Hussain, and A. Bhuiyan, "Bengali sign language recognition using dynamic skin calibration and geometric hashing," in *Proc. 6th Int. Conf. Informat., Electron. Vis. 7th Int. Symp. Comput. Med. Health Technol. (ICIEV-ISCMHT)*, Sep. 2017, pp. 1–5.

[36] N. Basnin, L. Nahar, and M. S. Hossain, "An integrated CNN-LSTM model for Bangla lexical sign language recognition," in *Proc. Int. Conf. Trends Comput. Cognitive Eng.* Singapore: Springer, 2021, pp. 695–707.

[37] L. Xuan and Z. Hong, "An improved Canny edge detection algorithm," in *Proc. 8th IEEE Int. Conf. Softw. Eng. Service Sci. (ICSESS)*, Nov. 2017, pp. 275–278.

[38] F. Alkhalid, "Online preprocessing of gesture signs using background substructure and edge detection algorithms," *Int. J. Simul. Syst. Sci. Technol.*, vol. 21, no. 2, pp. 1–6, Mar. 2020.

[39] D. Tasmere, B. Ahmed, and M. M. Hasan, "Bangla sign digits: A dataset for real time hand gesture recognition," in *Proc. 11th Int. Conf. Electr. Comput. Eng. (ICECE)*, Dec. 2020, pp. 186–189.

[40] M. S. Islam, S. S. S. Mousumi, N. A. Jessan, A. S. A. Rabby, and S. A. Hossain, "Ishara-lipi: The first complete MultipurposeOpen access dataset of isolated characters for Bangla sign language," in *Proc. Int. Conf. Bangla Speech Lang. Process. (ICBSLP)*, Sep. 2018, pp. 1–4.

[41] S. Das, M. S. Imtiaz, N. H. Neom, N. Siddique, and H. Wang, "A hybrid approach for Bangla sign language recognition using deep transfer learning model with random forest classifier," *Exp. Syst. Appl.*, vol. 213, Mar. 2023, Art. no. 118914.

KHAN ABRAR SHAMS received the B.Sc. degree in computer science and engineering from Brac University. He is currently a Software Engineer, specializing in Django, Laravel, Next.js, Nuxt.js, Vue.js, and React.js. His research interests include artificial intelligence, image processing, and computer vision.

MD. RAFID REAZ is currently pursuing the B.Sc. degree in computer science and engineering with Brac University. In addition to his academic pursuits, he also has experience as a Data Analyst, a Web Developer, and a Content Writer. His research interests include image processing, machine learning, computer vision, and geospatial analysis.

MOHAMMAD RYAN UR RAFI is currently pursuing the B.Sc. degree in computer science with Brac University. His research interests include artificial intelligence, image processing, and computer vision. In addition to that, his interests lie in full stack development using MERN stack and cyber security (CTFs).

SANJIDA ISLAM is currently pursuing the B.Sc. degree in computer science and engineering with Brac University. Her research interests include computer vision, artificial intelligence, image processing, and natural language processing.

MD. SHAHRIAR RAHMAN received the bachelor's degree in computer science and engineering from Brac University. After graduation, he spent about a year in the software industry to gain some hands-on experience. He is currently a Lecturer with the Department of Computer Science and Engineering, Brac University. As an Instructor, he always tries to guide the students in the right direction.

RAFEED RAHMAN received the B.Sc. degree in computer science and engineering and the M.Sc. degree in computer science from Brac University. He is currently a Lecturer with the Department of Computer Science and Engineering, Brac University. His research interests include artificial intelligence, machine learning, natural language processing, and computer vision.

MD. TANZIM REZA received the B.Sc. degree in computer science and engineering from Brac University, in 2018. Afterward, he joined as a contractual Lecturer, in 2019, and eventually became a full-time Lecturer, in January 2020. His research interests include artificial intelligence, machine learning, natural language processing, and computer vision.

MOHAMMAD ZAVID PARVEZ received the B.Sc. degree in computer science and engineering from the Asian University of Bangladesh, the M.Sc. degree in electrical engineering from Blekinge Institute of Technology, Sweden, and the Ph.D. degree in computer science from Charles Sturt University, in 2016. He is currently a Research Fellow with the Cyber Security Cooperative Research Centre (CSCRC) Project, Charles Sturt University. He has published more than 63 refereed publications, including IEEE Transactions on Biomedical Engineering, IEEE Transactions on Neural Systems and Rehabilitation Engineering, *Neurocomputing*, and *Neural Processing Letters*. He has supervised more than 60 research topics and more than 40 articles were published under his supervision. He has over 16 years of research experience. His research interests include cyber security, machine learning, data science, and biomedical signal/image analysis.

SUBRATA CHAKRABORTY (Senior Member, IEEE) received the Ph.D. degree in decision support systems from Monash University, Australia. He is currently a Senior Lecturer with the School of Science and Technology, Faculty of Science, Agriculture, Business, and Law, University of New England, Australia. Previously, he was an Academician with the University of Technology Sydney, (UTS), Australia, University of Southern Queensland, Charles Sturt University, Queensland University of Technology, and Monash University. He is also a Visiting Fellow with the Centre for Advanced Modeling and Geospatial Information Systems (CAMGIS), Faculty of Engineering and IT, UTS, and a Visiting Fellow with the Griffith Business School, Queensland. His current research interests include data analytics, machine learning, image processing, and optimization with decision support applications in diverse domains, including health, agriculture, business, and education.

BISWAJEET PRADHAN (Senior Member, IEEE) received the B.Sc. degree (Hons.) from Berhampur University, India, the M.Sc. degree from Indian Institute of Technology (IIT) Bombay, India, the joint M.Tech. degree in civil engineering from IIT Kanpur, Kanpur, India, and Dresden University of Technology, Dresden, Germany, and the Ph.D. degree in GIS and geomatics engineering from Universiti Putra Malaysia, Serdang, Malaysia. He is currently a Distinguished Professor with the School of Information, Systems, and Modeling, and the Director of the Centre for Advanced Modeling and Geospatial Information Systems, University of Technology Sydney. Before joining the School of Information, Systems, and Modeling, Faculty of Engineering and IT, he was a Full Professor of GIS and remote sensing with the Faculty of Engineering, Universiti Putra Malaysia. He has more than 19 years of teaching, research, and industrial experience in the field of geospatial information systems. He is listed as the World's most Highly Cited Researcher by Clarivate Analytics Report, in 2016 and 2017, as one of the world's most influential mind. He was a recipient of the Alexander von Humboldt Research Fellowship from Germany. In 2011, he received the habituation in remote sensing from Dresden University of Technology. He was a recipient of German Academic Exchange Research (DAAD) Fellowship Award, the Saxony State Fellowship, from 1999 to 2002, the Keith Aurtherton Research Award, and the Georg Forster Research Award from German Government. Since February 2015, he has been serving as the Ambassador Scientist for Alexander Humboldt Foundation, Germany. He has widely travelled abroad visiting more than 55 countries to present his research findings.

ABDULLAH ALAMRI received the B.S. degree in geology from King Saud University (KSU), Riyadh, Saudi Arabia, in 1981, the M.Sc. degree in applied geophysics from the University of South Florida, Tampa, FL, USA, in 1985, and the Ph.D. degree in earthquake seismology from the University of Minnesota, Minneapolis, MN, USA, in 1990. He is currently a Professor of earthquake seismology and the Director of the Seismic Studies Center, KSU. He has published more than 150 research articles, achieved more than 45 research projects, and authored several books and technical reports. His recent projects also involve applications of electromagnetic (EM) and magnetotelluric (MT) in deep groundwater exploration of the empty quarter and geothermal prospecting of volcanic harrats in the Arabian shield. He is also a Principal and a Co-Investigator in several national and international projects (KSU, King Abdulaziz City for Science and Technology (KACST), National Plan for Science and Technology (NPST), Institute for Research and Investment Services (IRIS), Comprehensive Nuclear Test Ban Treaty Organization (CTBTO), U.S. Air Force, National Science Foundation (NSF), UC San Diego (UCSD), Lawrence Livermore National Laboratory (LLNL), Ohio State University (OSU), Prince Sultan University (PSU), and Max Planck). His research interests include crustal structures and seismic micro zoning of Arabian Peninsula. He is also a member of the Seismological Society of America, American Geophysical Union, European Association for Environmental and Engineering Geophysics, the Earthquakes Mitigation in the Eastern Mediterranean Region, the National Committee for Assessment and Mitigation of Earthquake Hazards in Saudi Arabia, and the Mitigation of Natural Hazards Commission at Civil Defense. He obtained several worldwide prizes and awards for his scientific excellence and innovation. He has chaired and co-chaired several Standing Scientific Group (SSG), Geoscience Frontiers (GSF), and Reducing Earthquake Losses in the Extended Mediterranean Region (RELEMR) workshops and forums in Middle East. He is also the President of Saudi Society of Geosciences and the Editor-in-Chief of the *Arabian Journal of Geosciences* (AJGS).