Computer Vision-based Advanced Hybrid Convolutional Neural Network to Recognize Sign Language for both Numeral and Alphabet Signs

Kabiratun Ummi Oyshe[1], Md Ikramul Haque Nirjon[1], Muhammad Aminur Rahaman[1*], Tanoy Debnath[2], Utpol Kanti Das[3], Anichur Rahman[3*]

[1]Computer Science and Engineering, Green University of Bangladesh, Purbachal American City, Narayanganj, 1461, Dhaka, Bangladesh.
[2]Department of Computer Science, Stony Brook University, 100 Nicolls Road, Stony Brook, 11794, NY, USA.
[3]Computer Science and Engineering, National Institute of Textile Engineering and Research (NITER), Kohinoor Gate, Savar, 1350, Dhaka, Bangladesh.

*Corresponding author(s). E-mail(s): aminur@cse.green.edu.bd; anis_cse@niter.edu.bd;
Contributing authors: kabiratunummi@gmail.com; ikramnirjon@gmail.com; tadebnath@cs.stonybrook.edu; ukdas@niter.edu.bd;

Abstract

People with verbal communication challenges often use sign language, but many find it difficult to understand their messages. Researchers have worked together to create a system that helps people who cannot speak communicate with those who use sign language. Even with some flaws, the system achieved its goal successfully. Our main goal in this research is to create Sign Language Recognition (SLR) technology that goes beyond the limitations of the past. This study presents the Advanced Hybrid Convolutional Neural Network (AdH-CNN), which is an ensemble model that merges the VGG16 and ResNet50 neural networks. Combining VGG16's detail capture with ResNet50's deep representation improves feature variety, resulting in better recognition. This combination improves accuracy on the datasets "BdSL_OPSA22_STATIC1" and "BdSL_OPSA22_STATIC2". Moreover, the simplicity of VGG16 balances the

complexity of ResNet50, resulting in a strong but manageable computational need. The datasets "BdSL_OPSA22_STATIC1" and "BdSL_OPSA22_STATIC2" are our own and include backgrounds with unclear features. Each dataset contains 24,615 images of Bangla characters and numbers. Finally, the "AdH-CNN" model outperforms earlier models, achieving an accuracy of 96.51 percent on "BdSL_OPSA22_STATIC1" and 97.24 percent on "BdSL_OPSA22_STATIC2".

Keywords: Convolutional Neural Network, Bangla Sign Language, VGG16, ResNet, BdSL_OPSA22_STATIC1, BdSL_OPSA22_STATIC2.

1 Introduction

Sign language is an essential mode of communication for individuals with speech or hearing impairments. Their heads, features, and hands will all assist them in expressing their emotions and the message they wish to convey [1]. Posture, contour, and hand movement technique [2] are the three primary factors that determine it. Deaf individuals worldwide have access to a diverse array of languages, as various regions have adopted their own sign languages [3]. Despite the fact that hearing loss is a substantial issue, approximately 5% of the global population experienced difficulties at work, in the classroom, and during social events. Long-term exposure to harsh noise, childhood diseases, or genetic disorders can all lead to hearing loss. This underscores the importance of identifying technical solutions that facilitate more efficient communication among individuals.

In education, healthcare, customer service, and emergency response, Sign Language Recognition (SLR) technology improves usability for hard-of-hearing individuals. Yet the development of robust SLR systems is a difficult task because of the complexity of sign movements, the ambiguity in defining continuous vs. stationary recognition, variable backgrounds, and the necessity of real-time processing. These challenges are especially evident in under-resourced sign languages, such as Bengali Sign Language (BdSL), which has recently garnered research attention. Although previous research has made strides in the recognition of static BdSLs, it frequently concentrates on controlled environments, disregarding real-world obstacles such as user diversity, varying illumination, and locations [4, 5]. The complexity of sign language data presents a challenge for traditional machine learning methods such as SVM and HMM [6, 7]. However, CNNs have demonstrated superior accuracy and feature extraction capabilities [8]. But the majority of research relies on tiny datasets with uniform backgrounds, ignoring the necessity of diverse datasets that capture variations in skin tone, lighting, and environment. There are a number of difficulties in creating a hybrid convolutional neural network based on computer vision that can recognize both alphabet and numeral sign language. Accuracy of recognition can be affected by individual differences in hand gestures, signs that look alike, and poor lighting or backgrounds. The model design is further complicated by the need for real-time processing, and small and unbalanced datasets. Significant challenges that must be overcome include ensuring generalization across users and adjusting to regional variations in sign language.

This constraint impedes the development of dependable SLR systems for practical applications, particularly in real-world BdSL recognition scenarios. The recognition of BdSL in real-world scenarios is a complex task that existing systems are unable to effectively address. These consist of:

- Changing lighting conditions can make hand motions far more difficult to observe and comprehend, therefore affecting the accuracy of recognition.
- Using identification models for everyone is challenging since individuals's hands move and position themselves differently.
- The majority of the datasets that are currently available are too small to include all of the diverse backgrounds, skin tones, and groups that are representative of the real world.
- Real-world applications necessitate low-latency processing, which can be difficult for models that are significantly dependent on work.
- These issues highlight the need to develop fresh approaches to handle the complexity and changeability of BdSL in different environments.

In spite of the promising advancements in Sign Language Recognition (SLR) through deep learning, the current systems are significantly constrained when applied to real-world scenarios, particularly for under-resourced languages such as Bengali Sign Language (BdSL). The majority of current research is concentrated on controlled environments, which are characterized by small datasets with limited background diversity. This constrains their generalizability across users and contexts. The aim of this study is to create a more efficient and robust SLR system that can operate effectively in a variety of environments, including varied lighting conditions, hand postures, and backgrounds, in response to these deficiencies. By incorporating the capabilities of ResNet50 and VGG16 into a hybrid CNN architecture (AdH-CNN), this research endeavors to enhance the accuracy of recognition and the efficiency of computation, thereby facilitating real-time, real-world BdSL recognition. In addition, the model's suitability for practical, inclusive SLR applications in education, healthcare, and emergency communication is bolstered by the development of a vast, demographically diverse dataset.

In real-world scenarios, modern algorithms frequently fail to meet expectations due to constraints in model adaptability, dataset diversity, and processing efficiency, despite recent advancements. This study aims to assess the potential of hybrid CNN architectures to be optimized for robust and scalable BdSL recognition in a variety of real-world conditions, thereby addressing these gaps.

- How can deep learning models be enhanced to precisely recognize Bangla Sign Language in unsupervised, real-world situations with diverse hand postures, lighting conditions, and backgrounds?
- Is it possible to improve the spatial feature extraction needed to process complex BdSL postures while maintaining real-time performance by incorporating VGG16 and ResNet50 architectures into a hybrid CNN model?
- What are the effects of dataset diversity on the applicability and robustness of SLR systems?

- To what extent does the AdH-CNN model that has been proposed exceed the performance of existing models in terms of computational efficiency and recognition accuracy on large-scale BdSL datasets?

While CNN-based methods achieve high accuracy in recognizing stationary BdSL, they struggle with spatial and temporal variations and are often trained on small, homogeneous datasets, limiting their adaptability to diverse environments. Computational constraints further hinder real-time implementation, underscoring the need for innovative solutions to improve recognition in real-world scenarios. Although CNNs excel in spatial feature extraction compared to Transformers and LSTMs, hybrid models like ResNet50 and VGG16 offer a practical balance between accuracy and efficiency. These hybrid approaches address the limitations of traditional methods like SVM and HMM, which fail to handle the complexity of sign language variation. By integrating ResNet50 and VGG16, hybrid CNNs provide a robust framework for real-time BdSL recognition, making them suitable for practical applications. Recent advancements in deep learning, such as CorrNet+ for spatial-temporal correlation [9] and EvSign for mitigating motion distortion and background noise [10], have significantly improved SLR capabilities. Additionally, real-time SLR systems enable live translation, enhancing practical usability [11]. To address existing challenges, this study introduces AdH-CNN, a hybrid CNN model integrating VGG16 and ResNet50 architectures. VGG16 identifies basic gesture structures, while ResNet50 captures complex sign variations, enabling efficient extraction of spatial features. Evaluated on two BdSL datasets, AdH-CNN demonstrates superior accuracy and performance, offering a robust solution for real-world SLR applications. The main contributions of the article are as follows:

- To develop a CNN-based model that is capable of accurately identifying Bangla Sign Language (BdSL) in a variety of real-world environments, including those with varying hand posture, background, and illumination, that is more robust.
- To enable the model to be more ubiquitous, a comprehensive BdSL dataset with 24,615 images that showcase a variety of demographics and environmental conditions will be generated.
- To combine the VGG16 and ResNet50 structures will result in the development of a more sophisticated CNN-based model (AdH-CNN), which will increase the number of features and maintain a high level of computational efficacy for the recognition of sign languages.
- To achieve superior recognition accuracy the proposed model's 96.51 percent success rate on "BdSL_OPSA22_STATIC1" and 97.24 percent success rate on "BdSL_OPSA22_STATIC2" suggest that it is possible to achieve a higher level of recognition accuracy than current models.
- To enhance our understanding and recognition of sign language, we implemented the most effective components of VGG16 for feature extraction and ResNet50 for deep representation.
- To ensuring that the AdH-CNN model is suitable for real-world applications, such as improving the usability of communication tools for hearing-impaired individuals,

and that it processes quickly and accurately detects BdSL in a variety of lighting and background conditions will enable the model to function in real time for significant purposes.

The rest of the paper is structured as follows: Section 2 reviews related work on Sign Language Recognition. Section 3 details the proposed methodology, including feature extraction and classification techniques. Section 4 presents the performance evaluation. Section 5 discusses Conclusion, limitations and future research directions.

2 Literature Review

Assisting individuals with disabilities has been the focus of extensive research in recent years, particularly in developing effective techniques for sign language recognition. While Convolutional Neural Networks (CNNs) [12, 13] have been widely utilized, other methods have also been explored and implemented.

In 2023, Das et al. [14] proposed a method for recognizing British Sign Language (BdSL) using a hybrid transfer learning system with the RF classifier [15] but struggled with the absence of a publicly available dataset, which hindered model reproducibility. An algorithm was developed to remove background elements from sign images, enhancing recognition performance. Their work faced a significant limitation: the absence of a comprehensive, high-quality, publicly available dataset for British Sign Language (BSL), which restricted the scalability and reproducibility of their model. Khatawate et al. [16] this study examines a CNN-based Sign Language Recognition (SLR) system using pre-trained models like VGG16 and ResNet50 on a 26 ASL alphabet and number dataset. VGG16 outperformed ResNet50 with 99.92% accuracy. However, challenges like individual signature skills variability, hand shape changes, and dynamic and continuous signing complications remain. Despite these challenges, establishing generalizable SLR systems remains a challenge.

Islam et al. [17] developed an automated sign language recognition (SLR) system for Bengali Sign Language (BdSL) in 2022. They used four established transfer learning models VGG16 [18], VGG19 [19], AlexNet [20], and InceptionV3 [21] using pre-trained weights to achieve remarkable accuracy rates of 99. 92%, 99. 58%, 97. 86%, and 98. 70% to recognize 11 commonly used Bengali sign words. However, their approach was limited by a relatively small vocabulary, which restricts its applicability in real-world scenarios requiring broader sign language coverage. Rao et al. [22] introduced a complex CNN architecture for recognizing Indian Sign Language (ISL) and curated a unique dataset of 200 signs performed against five different backgrounds. While achieving a recognition accuracy of 92.88%, their model's reliance on a limited dataset and their study was conducted under controlled and lighting conditions remain challenges for real-world applicability.

Yasir et al. [23] used a CNN approach to recognize Bengali Sign Language (BdSL), employing the Leap Motion Controller (LMC) [24] and the Hidden Markov Model (HMM)[25] for real-time hand tracking. However, their work was restricted by the dependency of the device on specific hardware, which limits accessibility and adoption

in cost-sensitive environments. Zaki et al. [26] proposed an innovative vision-based sign language recognition system, while Kasukurthi et al.[27] developed a neural network optimized for mobile devices to recognize the American Sign Language (ASL) alphabet. They adopted the efficient SqueezeNet architecture, achieving an accuracy of 83.29%. However, the lower accuracy and limited vocabulary demonstrate the need for further optimization and expanded datasets. Although computationally efficient, reduced accuracy demonstrated trade-offs between model size and recognition performance.

Shamsaldin et al. [28] emphasizes how CNNs play a key role in resolving challenging issues in the linguistic and visual domains. An overview of the two main areas in which Convolutional Neural Networks (CNNs) are used. First, CNNs are used in computer vision for tasks like image classification, face recognition, action recognition, and scene labeling. Second, CNNs are used in natural language processing (NLP) for tasks like text classification and speech recognition. In order to improve facial emotion recognition in unrestricted settings, a CNNEELM model that combines Convolutional Neural Networks with Extreme Learning Machines was recently proposed [29]. Through the use of a modified SGD optimizer and optical flow for peak frame extraction, the model was able to decrease processing time from 113 ms to 65 ms and improve accuracy by 2%. The model enables real-time video analysis and successfully recognizes six emotions after being trained on the JAFFE, CK+, and FER2013 datasets using a pre-trained InceptionV3. Moreover, A technique for facial expression recognition using imbalanced facial expression datasets was introduced [30], involving key preprocessing steps such as data balancing and significant feature extraction. These features were then fed into various classifiers, including Decision Tree (DT), Multi-Layer Perceptron (MLP), and Convolutional Neural Network (CNN). Among these, CNN achieved the highest recognition accuracy, demonstrating its effectiveness in handling imbalanced data for facial emotion recognition.Meena et al [31] introduces a Few-Shot Transfer Learning framework (FSTL-SA) for sentiment analysis from facial expressions, specifically designed for situations with limited labeled data. Initially, a deep convolutional neural network (DCNN) is trained on large datasets (CK+ and FER2013) for anger, fear, and surprise. A few-shot learning (N-way-k-shot) approach is employed to fine-tune the DCNN for happy, sad, and neutral expressions. A two-stage semi-supervised approach is also employed to address data scarcity. The proposed method achieving an accuracy of 75.33% in the source domain and up to 82% in few-shot scenarios with semi-supervised learning.

Hasan et al.[32] presented a deep CNN-based model for classifying Bangla sign language characters with an impressive accuracy of 99.22%. However, their approach did not address variations in hand size, skin tone, or background conditions, limiting its robustness in diverse real-world settings. Similarly, Bird et al. [33] used transfer learning models to compare British and American sign languages, achieving accuracy rates of 94.44% for BSL and 82.55% for ASL. However, their reliance on Kaggle-sourced datasets introduced variability in data quality and limited their study's generalizability. However, dataset inconsistencies affected overall model performance.

Rahman et al. [34] introduced a real-time computer vision-based system for recognizing Bengali Sign Language in 2014. Their system utilized the K-Nearest Neighbor

(KNN) [35] classifier to detect BdSL, achieving accuracy rates of 98.17% for vowels and 94.75% for consonants. Despite these impressive results, their method was limited to static signs and required substantial manual effort for feature extraction, making it less suitable for dynamic, real-world applications. Table 1 highlight the gaps in existing work for sign language recognition (SLR) studies.

Table 1 Gaps in current sign language recognition research

References	Gap
Das et al.[14], Rao et al.[22]	Lack of Public Datasets
Islam et al. [17] , Kasukurthi et al.[27]	Limited Vocabulary
Yasir et al.[23]	Hardware Dependency
Rahman et al.[34]	Real-Time Deployment Challenges
Hasan et al.[32]	Lack of Robustness
Kasukurthi et al.[27]	Accuracy vs. Efficiency Trade-offs
Bird et al.[33]	Dataset Inconsistencies
Rao et al.[22]	Controlled Environment Bias
Bird et al.[33], Islam et al.[17]	Limited Generalizability

Hybrid CNN models are designed to enhance the precision of feature extraction and recognition by incorporating a variety of architectures. However, a number of the current methods are unable to effectively integrate deep features or prioritize comprehensive architectural integration. For instance, Islam et al. [17] conducted independent tests on numerous CNN architectures, such as VGG16 and InceptionV3, while Das et al. [14] integrated transfer learning with RF classifiers, choosing not to employ deep feature fusion. In addition to CNNs, Transformer-based models, Graph Neural Networks (GNNs), and Recurrent Neural Networks (RNNs) demonstrate promising advancements in deep learning. Transformers utilize self-attention mechanisms to identify dynamic gestures; nevertheless, they encounter substantial computational obstacles. Graph neural networks are capable of modeling hand-joint relationships to enhance gesture recognition, whereas recurrent neural networks and long short-term memory networks excel in sequence recognition but encounter real-time performance challenges due to their substantial computational demands.

By addressing the shortcomings of these previous methods, such as limited datasets, restricted vocabularies, hardware dependencies, and challenges with variability in real-world conditions our proposed approach seeks to deliver a robust, scalable, and accurate solution for sign language recognition. Our model incorporates diverse datasets, advanced feature extraction techniques, and a generalized architecture designed to handle real-world complexities.

3 Proposed Methodology for Sign Language Recognition

In this study, we used images from our own collection that included 46 distinct Bangla symbols (36 Bengali characters and 10 Numerical Signs). We developed a model named "Advanced Hybrid Convolutional Neural Network" (AdH-CNN), which is an amalgam

of ResNet50 and VGG16. The input size for this model is 224*224 images, which we utilize as our inputs. The pre-processed pictures are sent into the suggested CNN model for the extraction of features, classification, and training.

Fig. 1 represents the whole architectural view of the system of 46 Bangla Signs and Characters recognition system.

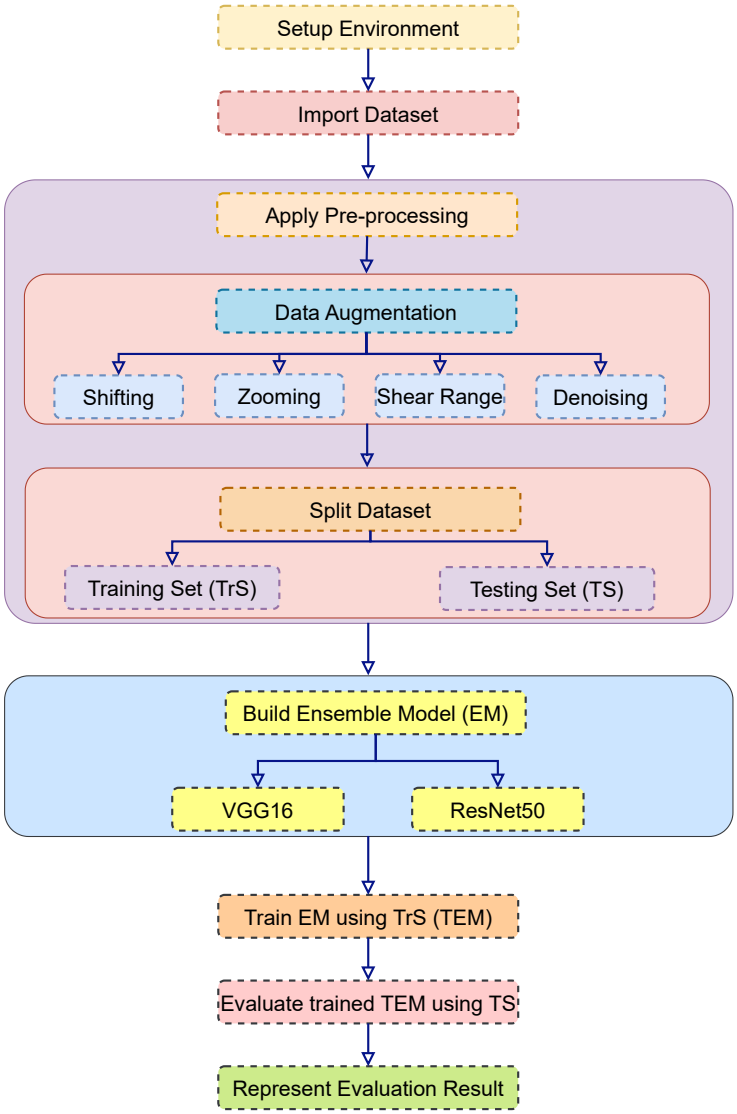

Fig. 1 Proposed Methodology for Posture Recognition Using an Ensemble Model

3.1 Import Dataset

This step involves loading the datasets named "BdSL_OPSA22_STATIC1" and "BdSL_OPSA22_STATIC2", which contain static images of 46 Bangla Sign Language (BdSL) characters collected in various lighting conditions and backgrounds. The datasets are typically loaded into the Python environment using libraries like 'Pandas', 'NumPy', or image processing tools such as 'OpenCV' or 'PIL'. The detailed description of the used datasets is illustrated in Dataset Section 4.1.

3.2 Apply Preprocessing

This step involves preparing the images from the datasets for optimal model performance. First, each image is resized to a standard dimension of 224x224 pixels with 3 color channels (RGB) to maintain consistency across the dataset and match the input size required by many deep learning models. Next, denoising techniques are applied to reduce any unwanted noise that may obscure important features, ensuring clearer image quality. Finally, partial contrast stretching is performed to enhance the contrast in the images, making subtle features more distinguishable. These preprocessing steps help improve the model's ability to accurately recognize and classify the Bangla Sign Language characters under varying lighting conditions. Fig. 2 demonstrates the preprocessing steps.

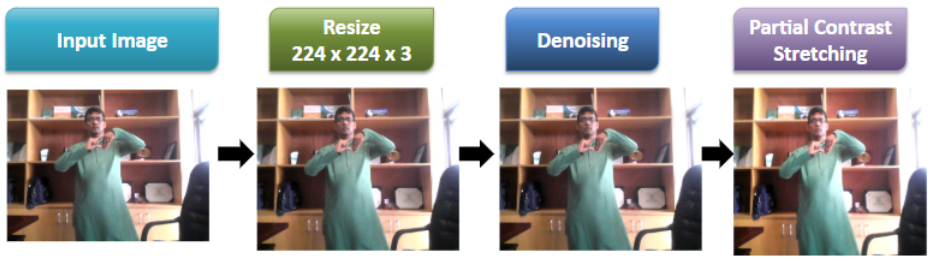

Fig. 2 Demonstrates data pre-processing techniques used in our model

We improved sign recognition accuracy and efficiency by tuning the algorithms. Algorithm 1 illustrates the procedure for preparing image files and labeling. This algorithm is designed for image data pre-processing and labeling, ensuring that images are properly formatted, resized, normalized, and labeled for deep learning applications [14]. It begins by initializing two empty lists, one for storing pre-processed images and another for storing corresponding labels. The algorithm then iterates through all files in the specified image directory, constructing the full file path for each image. Before proceeding with processing, it checks whether the file is a valid image to prevent errors. Once verified, the image is loaded and resized to 224*224 pixels, a common input size for deep learning models such as VGG16 and ResNet50. Next, the image is converted to RGB format to ensure compatibility, especially if it was originally in grayscale. After conversion, the algorithm expands the image dimensions to match the expected input format of deep learning frameworks, where batch size is included

Algorithm 1 Algorithm for Image Data Pre-processing and Labeling

Require: Images
Ensure: Pre-processed Images and Labels
Require: α, δ
$\upsilon \leftarrow \zeta$ *Initialize an empty list for pre-processed image data.
$\chi \leftarrow \zeta$ *Initialize an empty list for image labels.
for all f in os.$\theta(\alpha$ **do**
    $\kappa \leftarrow \xi$.join(α, f)
    if ξ.isfile(κ) **then**
        $\psi \leftarrow \lambda.\rho($ κ, ε=(224, 224))
        $\mu \leftarrow \lambda.\tau(\psi)$ Convert the image to the appropriate color format (e.g., RGB).
        $\mu \leftarrow$ np.$\Gamma($ μ, axis=0)
        $\mu \leftarrow \mu$ / 255.0 *Rescaled Pixel Value
        $\upsilon.\phi($ μ)
        $\chi.\phi(\delta)$
        print(" EPF " + f + " : " + str(e))
    end if
end for

as an additional dimension. To improve training stability, the image's pixel values are normalized to a range between 0 and 1 by dividing by 255. The pre-processed image is then appended to a list, along with its corresponding label, ensuring that both datasets remain aligned. Finally, the algorithm includes an error-handling mechanism, which prints an error message if any image fails to process, allowing the execution to continue without interruption. This approach ensures that all images are uniformly processed and labeled, creating a clean and structured dataset for training deep learning models effectively.

3.2.1 Data Augmentation

This step applies various transformations to the BdSL_OPSA22_STATIC1 and STATIC2 datasets to improve model generalization. Key augmentations include shifting, which translates images slightly to make the model robust to positional changes, and zooming, which enlarges or shrinks parts of the image to introduce scale variations. Shear ranging is used to skew images along an axis, helping the model handle angular distortions. At the same time, denoising improves image clarity by removing noise, allowing the model to focus on essential features. Importantly, flipping is avoided, as it could alter the structure of the Bangla Sign Language gestures, changing the meaning and leading to misclassification. These augmentations introduce variability while maintaining the integrity of the sign language characters.We also applied the SMOTE (Synthetic Minority Oversampling Technique) framework to balance the imbalanced classes and categories when required [36]. SMOTE is an effective method for balancing datasets. It improves model performance by balancing the dataset as well as reducing the risk of overfitting. SMOTE ensures that the model pays adequate

attention to the minority class, improving metrics like recall, precision, and F1-score for under-represented classes.

3.2.2 Split Dataset

In this section, the BdSL_OPSA22_STATIC1 and STATIC2 datasets are divided into three groups with different training and testing ratios to evaluate model performance. The first split is 70% for training and 30% for testing (7:3), the second is 60% for training and 40% for testing (3:2), and the third is 80% for training and 20% for testing (4:1). After training the model on each of these splits, the results are evaluated, and it is observed that the 7:3 split consistently outperforms the others. This split provides a balanced amount of data for both training and testing, leading to better generalization of the model while maintaining enough data to accurately evaluate performance.

3.3 Build Ensemble Model

We developed a hybrid CNN model, "AdH-CNN" (Advanced Hybrid Convolutional Neural Network) which is an enesmble of VGG16 and ResNet50 with few added parameters. Utilizing the advantages of both of these well-known convolutional neural network designs, VGG16 and ResNet50, an ensemble model is produced that enhances the predictive performance. The input layer as well as the output layer of the ensemble model are created using Keras Functional API. It is put together using the Adam optimizer with a learning rate of 0.001 and a loss function of categorical cross-entropy loss. Fig. 3 demonstrates the detailed architecture of our proposed model "AdH-CNN". By combining VGG16 and ResNet50 in the AdH-CNN model, we leverage the unique strengths of each architecture to achieve superior performance in sign language recognition , like ResNet50 is much deeper than VGG16, allowing it to learn more sophisticated representations of the data. Simplicity and Uniformity, Deep Feature Extraction of VGG16 is another sophisticated representation of the data. Using this method, the model has both basic spatial features and deep hierarchical representations. It might get better and more useful. This is the only ensemble method that doesn't use model averaging, stacking, or boosting like other methods do. AdH-CNNs outperform traditional CNNs due to their adaptive feature learning, hierarchical representation, regularization techniques, context-aware adaptation, efficient parameter sharing, resistance to noise and variability, and transfer learning advantages. These qualities make it an excellent choice for transfer learning since they reduce overfitting, improve performance on previously unknown data, and capture important patterns.

3.3.1 Fine Tuning Process of Proposed CNN

For our "AdH-CNN" (Advanced Hybrid Convolutional Neural Network) framework, we adjusted several parameters and hyper-parameters. In addition to adding a fully linked layer, a refinement technique adjusts several hyper-parameters, including padding, stride, batch size, and filters. "AdH-CNN" model comprises of previously trained VGG16 consisting 16 layers and the pre-trained ResNet50 consisting 50 layers. The output from the VGG16 and output from ResNet50 are then concatenated

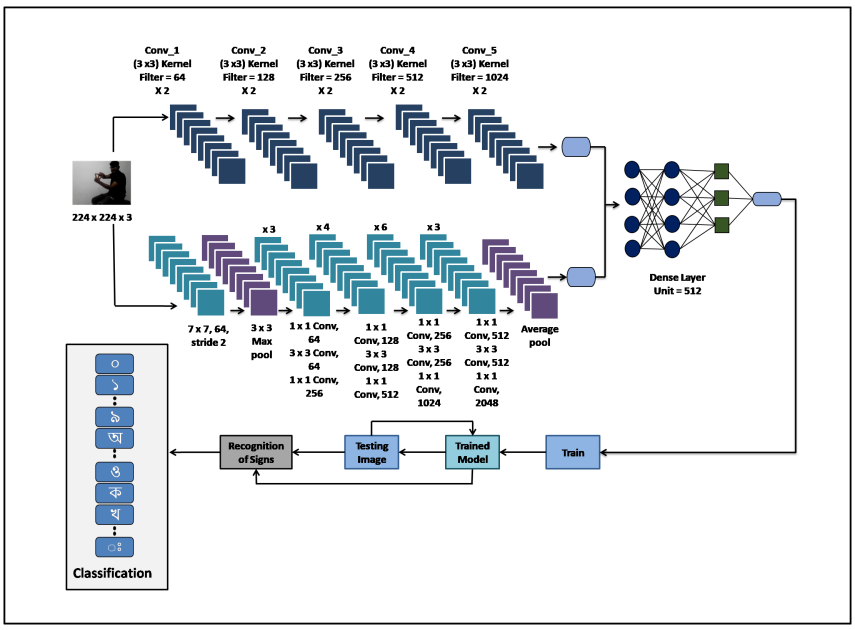

Fig. 3 AdH-CNN Hybrid Model combining VGG16 and ResNet50 for enhanced data representation

to get a final output. The Equation (1) calculates parameter count

$$P = O * (I + 1) \tag{1}$$

Where, I = Input channel number, P = Parameter number, O = Output channel.

In our model, "AdH-CNN", we added 1 fully connected layers which consists 512 neurons.The output of the final fully connected layer is a vector whose length is proportional to the problem's class count, with each component representing the input picture's likelihood of belonging to a specific class. Formula (2) for fully connected layer output:

$$V = a(WmF + b) \tag{2}$$

Where F is the input feature vector, Wm is the weight matrix, A is the activation function, and V is the output vector. The following layer, which may be another completely linked layer or the network's ultimate output layer, receives the output vector V. Fig. 4 shows the layered architecture of "AdH-CNN".

Algorithm 2 demonstrates the algorithm for fine-tuning the CNN model. We ensemble VGG16 and ResNet50, along with adding a fully connected layer. This algorithm fine-tunes a hybrid CNN model by integrating VGG16 and ResNet50 for image classification. It starts by loading the pre-trained VGG16 (Θ) and ResNet50 ($\Re$) models with their respective weights and input shapes. The feature extraction layers are retained, and their weights are frozen to preserve learned representations. Next, an input layer (γ) is defined, and the outputs from both models are concatenated (∞) to create

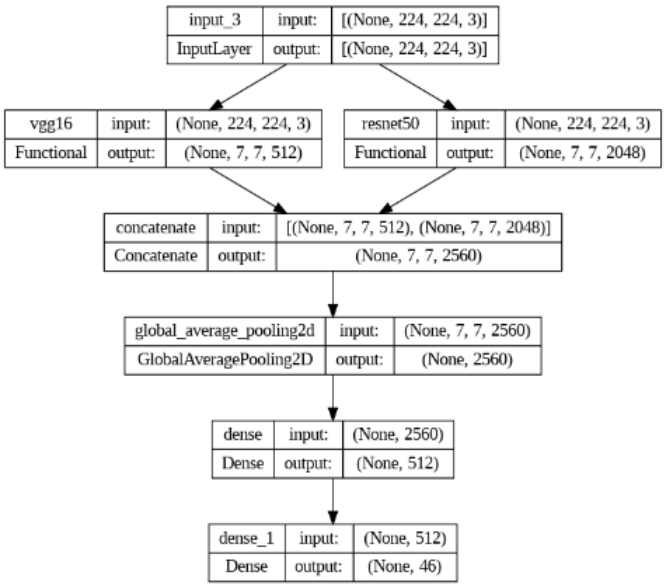

Fig. 4 Hybrid Model, "AdH-CNN" Architecture

Algorithm 2 Algorithm for Fine-tuning the CNN model

Require: Pre-processed Images
Ensure: Hybrid CNN model

$\Theta = V(\Xi = \Pi', \Sigma = F, \Upsilon = \varphi(\Phi = (\Psi, \Omega, 3)))$ *Pre-trained VGG16
$\Re = \nabla(\Xi = \Pi, \Sigma = F, \Upsilon = \varphi(\Phi = (\Psi, \Omega, 3)))$ *Pre-trained ResNet50
for $\forall$ in $\Theta.\Im$ **do**
    $\forall.\complement = F$
end for
for $\forall$ in $\Re.\Im$ **do**
    $\forall.\complement = F$
end for
$\gamma = K.\varphi(\Phi = (\Psi, \Omega, 3))$ *Input shape specifying
$\infty = \Re([\Theta(\gamma), \nabla(\gamma)])$ *Ensemble of outputs of VGG16 and ResNet50
$PO = K.\Im.GAP()(\infty)$ *Global Average Pooling 2D
$DA = K.\Im.D(512, ACT =' relu')(PO)$ *Adding a new Fully Connected layer
$OL = K.\Im.D(NC, ACT =' softmax')(DA)$ *Final output layer
$M = M(\gamma = \gamma, \infty = OL)$ *Final Hybrid Model
$M.c$ *Compile the model

an ensemble of feature maps. A Global Average Pooling (GAP) layer reduces spatial dimensions, followed by a fully connected layer with 512 neurons (ReLU activation) for learning dataset-specific features. The final softmax output layer (OL) is added for multi-class classification. The hybrid model (M) is then constructed and compiled,

integrating VGG16 and ResNet50's extracted features while allowing the newly added layers to fine-tune for improved classification accuracy.

Table 2 List of Symbols used in the proposed algorithm

Variable	Notation	Variable	Notation
folder_path	α	enumerate	β
folder_label	δ	dataset_path	ϵ
empty list	ζ	path	η
listdir	θ	image_name	ι
img_path	κ	image	λ
img_array	μ	imread	ν
os.path	ξ	img_width	π
load_img	ρ	cvtColor	σ
img_to_array	τ	data	υ
append	ϕ	labels	χ
img	ψ	array	ω
exapnd_dims	Γ	astype	Δ
target_size	ε	test_data	ϑ
pretrained_vgg	Θ	VGG16	V
weights	Ξ	imagenet	Π
include_top	Σ	input_tensor	Υ
Input	φ	shape	Φ
img_width	Ψ	img_height	Ω
outputs	∞	concatenate	$\Re$
pretrained_resnet	∇	ResNet50	∂
layer	$\forall$	layers	$\Im$
trainable	$\complement$	Inputs	γ
False	F	keras	K
pooled_outputs	PO	GlobalAveragePooling2D	GAP
dense_layer	DA	Dense	D
activation	ACT	output_layer	OL
num_classes	NC	compile	c
model	M	file	f
Error processing file	EPF	Exception	Ex

Table 2 denotes all the notations which are used to represent the variables of the proposed algorithms.

Table 3 presents the initial hyperparameters used for training a merged VGG16 and ResNet50 model for a 46-class image classification task. A batch size of 32 is chosen to balance computational efficiency and model performance. The model employs convolutional kernels of sizes (3,3) and (7,7) to capture both fine and broad spatial features. The number of filters increases progressively (64, 128, 256, 512, 1024) to extract hierarchical representations of image data. A dropout rate of 0.5 is applied to prevent overfitting, while a small learning rate of 0.0001 ensures stable convergence. Pooling operations use sizes (2,2) and (3,3) with both max and average pooling techniques to reduce dimensionality while preserving important features. The ReLU activation function is used in hidden layers to introduce non-linearity, while SoftMax is applied in the output layer for multi-class classification. The model is optimized using Adam, which adapts the learning rate dynamically for efficient training, and

Table 3 Initial Hyperparameters for Merged VGG16 and ResNet50 Model

Parameters	Parameter Value
Batch size	32
Kernel size	(3,3),(7,7)
Number of filters	64, 128, 256, 512, 1024
Dropout rate	0.5
Learning rate	0.0001
Pooling size	(2,2),(3,3)
Pooling type	Max, average
Activation	ReLU (hidden layers), SoftMax (output layer)
Optimizer	Adam
Loss function	Categorical Crossentropy
Number of classes	46

categorical cross-entropy is used as the loss function to measure classification performance. These hyperparameters collectively enhance the feature extraction capabilities and classification accuracy of the merged VGG16-ResNet50 architecture.

3.4 Train EM using Training set

In this, the ensemble model that combines VGG16 and ResNet50 is trained to learn the features of the dataset. First, the model is compiled with an optimizer (such as Adam) and a loss function, typically categorical cross-entropy, along with metrics like accuracy to track performance. The training process begins by calling the fit() function, where the model is provided with the training data, validation data, number of epochs, and batch size. During training, the model adjusts its weights to minimize the loss function based on the combined features from both VGG16 and ResNet50. To avoid overfitting, techniques like early stopping may be implemented, monitoring validation metrics after each epoch. Overall, this training step focuses on enabling the ensemble model to effectively learn to classify Bangla Sign Language characters.

3.5 Evaluate Trained model using Test set

In the Evaluate Model step, the performance of the trained ensemble model is assessed using a separate test dataset. The model makes predictions on this data, and various metrics are calculated, including accuracy to measure the proportion of correct predictions, as well as precision and recall to evaluate the model's ability to identify positive instances. A confusion matrix is also generated to visualize the model's performance across different classes, highlighting any common misclassifications. This evaluation provides insights into the model's strengths and weaknesses in classifying Bangla Sign Language characters and helps identify areas for improvement.

3.6 Represent Evaluation Results

In the Represent Evaluation Results step, we visualize and present the model's evaluation outcomes using various formats for a comprehensive understanding of its performance. A confusion matrix illustrates the true and false classifications for each

Bangla Sign Language character, while a classification report provides precision, recall, and F1-scores for detailed insights. Performance metrics are displayed using bar graphs or line plots to highlight differences across classes. If applicable, ROC curves may be included to show the trade-off between true and false positive rates. Additionally, comparative analyses with other models can be presented through box plots or bar graphs, and summary tables offer quick reference to key statistics like overall accuracy and average precision. This multifaceted representation enhances our understanding of the model's effectiveness and areas for improvement.

Fig. 5 illustrate the ablation inquiry, demonstrating how various model environments (e.g., residual connections, fine-tuning, feature fusion) contributed. The effects of each factor on training time, F1-score, recall, accuracy, and precision can be better grasped in this way.

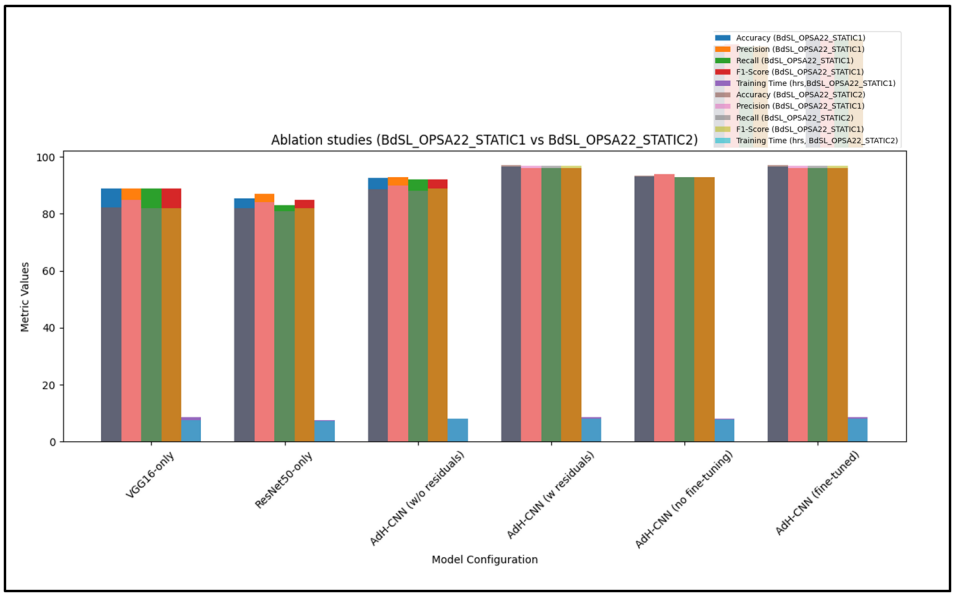

Fig. 5 Ablation study results comparing different model configurations to evaluate the contribution of residual connections, fine-tuning, and feature fusion.

4 Experimental Design and Results Analysis

4.1 Dataset

We have generated these datasets from scratch, named "BdSL_OPSA22_STATIC1"[37] and "BdSL_OPSA22_STATIC2"[38]. The "BdSL" stands for Bangla Sign Language; "OPSA22" refers to the author names who gathered the dataset (Oyshe, Prothoma, Sajid & Aminur), and 22 is the year of the data collection; "STATIC1" and "STATIC2" refers to the static image dataset. For our dataset, we have collected 46 Bangla characters in various lighting conditions and backgrounds. Bangla numbers were

included in our consideration. - "০(//Shunno//Zero)" to "৯(//Noy//Nine)", Bangla Shoroborno (6) - "অ(//Shoryo)", "আ(//Shorya)", "ই(//E)", "উ(//U)", "এ(//A)" and "ও(//O)", and Bangla Banjonborno (30) - "ক(//ka)", "খ(//kha)", "গ(//ga)", "ঘ(//gha)", "চ(//cha)", "ছ(//cho)", "জ(//jo)", "ঝ(//jho)", "ট(//ta)", "ঠ(//tha)", "ড(//da)", "ঢ(//dha)", "ত(//to)", "থ(//tho)", "দ(//do)", "ধ(//dho)", "ন(//no)", "প(//po)", "ফ(//fo)", "ব(//ba)", "ভ(//bho)", "ম(//mo)", "য়(//o)", "র(//ro)", "ল(//lo)", "স(//sha)", "হ(//ho)", "ড়(//ro)", "ং(//onushshar)" and "ঃ(//bishorgho)", in total 46 characters. In "BdSL_OPSA22_STATIC1" dataset, 7 people were photographed, including 5 men and 2 women. "BdSL_OPSA22_STATIC1" dataset contains 24615 photographs. The "BdSL_OPSA22_STATIC2" The dataset consists of 8437 photographs that were taken of a total of 7 people, including 4 males and 3 females.

Table 4 "BdSL_OPSA22_STATIC1" and "BdSL_OPSA22_STATIC2" Dataset (After SMOTE)

Bangla Characters and Numerals	**BdSL_OPSA22 _STATIC1**	**BdSL_OPSA22 _STATIC2**
Numerals	2695	1141
Vowels (Shorborno)	6146	1455
Consonants (Banjonborno)	15774	5841
Total Before SMOTE	24615	8437
Numerals (after SMOTE)	3000	3000
Vowels (after SMOTE)	7000	7000
Consonants (after SMOTE)	17000	17000
Total After SMOTE	29000	29000

Table 4 exhibits number of data on "BdSL_OPSA22_STATIC1" and "BdSL_OPSA22_STATIC2" datasets. We apply smote framework so that we can equalize the number of features in two datasets.

Fig. 6 shows several instances based on our datasets. Considering that all the images in our dataset were captured in a noisy backdrop instead of a plain one, it is unique compared to previous ones. In order to assist our system in identifying objects within a cluttered or blurred background, every image has an impervious background. In order to assess the accuracy of our model in comparison to other datasets, we applied it to both of our own created datasets.

For "BdSL_OPSA22_STATIC2" (Sun Light, Synthetic Light, and Natural Light) data on 46 Bangla numerals, vowels, and consonants was gathered from a sample of 7 individuals representing three distinct contexts and lighting scenarios. The 7 persons consist of 3 females and 4 males. Three distinct backgrounds inside Solid, outdoor Ambiguous, and indoor Clattered were used to gather our data. There are 8437 total photos, each with a unique backdrop and lighting. Table 5 shows the number of images in each background before and after applying the Smote framework.

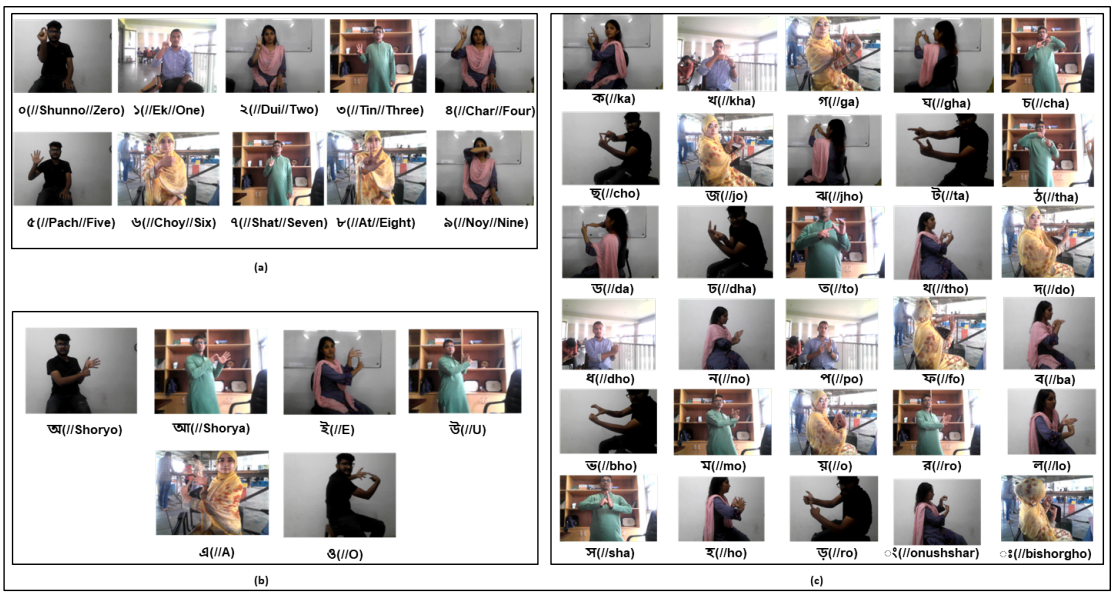

Fig. 6 Sample Postures of Datasets (a) Bangla Digits (10), (b) Bangla Vowels (Shoroborno)(6), (c) Bangla Consonants (Banjonborno)(30)-in various lighting condition, backgrounds and frame size

Table 5 NUMBER OF IMAGES IN DIFFERENT BACKGROUNDS IN "BdSL_OPSA22_STATIC2" DATASET (After SMOTE)

Backgrounds	**Number of Images**
Indoor Solid	2823
Indoor Clattered	3689
Outdoor Ambiguous	1925
Total Before SMOTE	8437
Indoor Solid (after SMOTE)	4000
Indoor Clattered (after SMOTE)	4000
Outdoor Ambiguous (after SMOTE)	4000
Total After SMOTE	12000

4.2 Setup Environment

The Setup Environment step for Bengali Sign Language Recognition in Python involves installing and importing necessary libraries. Key libraries include TensorFlow or PyTorch for model building, NumPy for numerical operations, and Matplotlib for visualizations. Image processing libraries like OpenCV or Pillow are used for handling image data, while Scikit-learn helps with data splitting and evaluation metrics [39]. After installation, these libraries are imported into the project to set up the environment for data preprocessing, model training, and evaluation. Finally, verifying the

setup ensures everything is ready for development. We utilized our own equipment to acquire the photographs included in our dataset. Data was collected using a variety of resolution cameras to ensure that the dataset was diverse in terms of quality and resolution. The following camera models were employed for data collection:

- DESKTOP-L160UV1 64-bit edition of Lenovo
- Redmi 7 with Android 9.0 Pie (12 MP)
- Infinix Note 10 with Android 11 (XOS 7.6)

4.3 Results Discussions

Table 6 Accuracy Results for Different Train-Test Splits

Dataset	3:2 Ratio	7:3 Ratio	4:1 Ratio
BdSL_OPSA22_STATIC1	94.25%	96.51%	95.77%
BdSL_OPSA22_STATIC2	95.32%	97.24%	96.43%

Table 6 represents the accuracy for different train-test splits where the 7:3 ratio outperforms another ratio. The AdH-CNN architecture is compared to already trained models in Table 7 on "BdSL_OPSA22_STATIC1" and "BdSL_OPSA22_STATIC2" datasets. Our proposed architecture "AdH-CNN" outperforms all existing state-of-the-art models on the "BdSL_OPSA22_STATIC1" dataset, achieving remarkable accuracy, precision, recall, and f1-score values of 96.51%, 96%, 96%, and 96% respectively. Similarly, on the "BdSL_OPSA22_STATIC2" dataset, the "AdH-CNN" framework surpasses all currently employed state-of-the-art models, attaining accuracy, precision, recall, and f1-score metrics of 97.24%, 97%, 97%, and 97% respectively. Upon thorough comparison with other established models, it becomes evident that our model has demonstrated superior performance. Using a matched t-test, we determined whether the variations in accuracy between STATIC1 and STATIC2 are statistically significant. The test assesses the performance of the same model on both datasets for each class. The p-value was 0.0000000000000014, which is significantly less than the 0.05 level of significance. The model exhibits a noticeably distinct behavior when applied to the two datasets, as the discrepancies in accuracy between STATIC1 and STATIC2 are statistically significant.

Table 7 represents the comparative study of different models with proposed system for both BdSL_OPSA22_STATIC1 and BdSL_OPSA22_STATIC2 datasets.

Table 8 and 9 indicates Bengali number recognition accuracy of "BdSL_OPSA22_STATIC1" and "BdSL_OPSA22_STATIC2" dataset using our proposed framework "AdH-CNN".

Fig. 7 represents the confusion matrix and ROC curves for numerals of both BdSL_OPSA22_STATIC1 and BdSL_OPSA22_STATIC2 dataset. Fig. 7 a. and Fig. 7. b. are the confusion matrix and ROC curve respectively of the BdSL_OPSA22_STATIC1 dataset for numerals. Fig. 7 c. and Fig. 7. d. are the confusion matrix and ROC curve respectively of the BdSL_OPSA22_STATIC1 dataset for numerals.

Table 7 Comparison of Four Learning Models and AdHCNN on BdSL_OPSA22_STATIC1 and STATIC2 Datasets

Dataset	Models	Accuracy	Precision	Recall	F1-Score	Training Time (hrs)
"BdSL _OPSA22 _STATIC1"	VGG16	89.01	89	89	89	8.5
	VGG19	87.76	90	88	86	9.0
	DenseNet-121	77.36	79	77	77	7.0
	Inception V3	59.84	59	60	54	6.5
	Proposed Method	96.51	96	96	96	8.5
"BdSL _OPSA22 _STATIC2"	VGG16	82.36	85	82	82	7.5
	VGG19	66.03	69	66	65	8.0
	DenseNet-121	51.51	65	52	53	6.0
	Inception V3	51.10	74	51	56	5.5
	Proposed Method	97.24	97	97	97	8.0

Table 8 Accuracy for Bengali Number Recognition using "BdSL_OPSA22_STATIC1" Dataset

Class	Accuracy	Precision	Recall	F1-Score	Inference Time [ms]
০ (0)	97.2	97.4	97.1	97.3	15.3
১ (1)	95.6	95.9	95.5	95.7	14.8
২ (2)	96.5	96.8	96.4	96.6	16.1
৩ (3)	97.0	97.3	97.1	97.2	15.2
৪ (4)	96.7	96.9	96.6	96.8	14.5
৫ (5)	95.9	96.1	95.8	96.0	16.3
৬ (6)	95.0	95.3	94.9	95.1	15.0
৭ (7)	94.8	95.0	94.7	94.9	16.0
৮ (8)	96.1	96.3	96.0	96.2	14.9
৯ (9)	95.5	95.7	95.4	95.6	15.4

Table 9 Accuracy for Bengali Number Recognition using "BdSL_OPSA22_STATIC2" Dataset

Class	Accuracy	Precision	Recall	F1-Score	Inference Time [ms]
০ (0)	98.0	98.2	98.1	98.1	14.1
১ (1)	97.5	97.7	97.6	97.6	14.7
২ (2)	97.2	97.4	97.3	97.3	15.0
৩ (3)	96.8	97.0	96.9	96.9	14.3
৪ (4)	97.4	97.6	97.5	97.5	14.5
৫ (5)	96.9	97.1	97.0	97.0	15.1
৬ (6)	97.0	97.2	97.1	97.1	14.8
৭ (7)	97.1	97.3	97.2	97.2	15.2
৮ (8)	97.6	97.8	97.7	97.7	14.9
৯ (9)	97.8	98.0	97.9	97.9	14.6

Table 10 and 11 indicates Bengali vowel recognition accuracy for "AdH-CNN" model on "BdSL_OPSA22_STATIC1" and "BdSL_OPSA22_STATIC2" dataset.

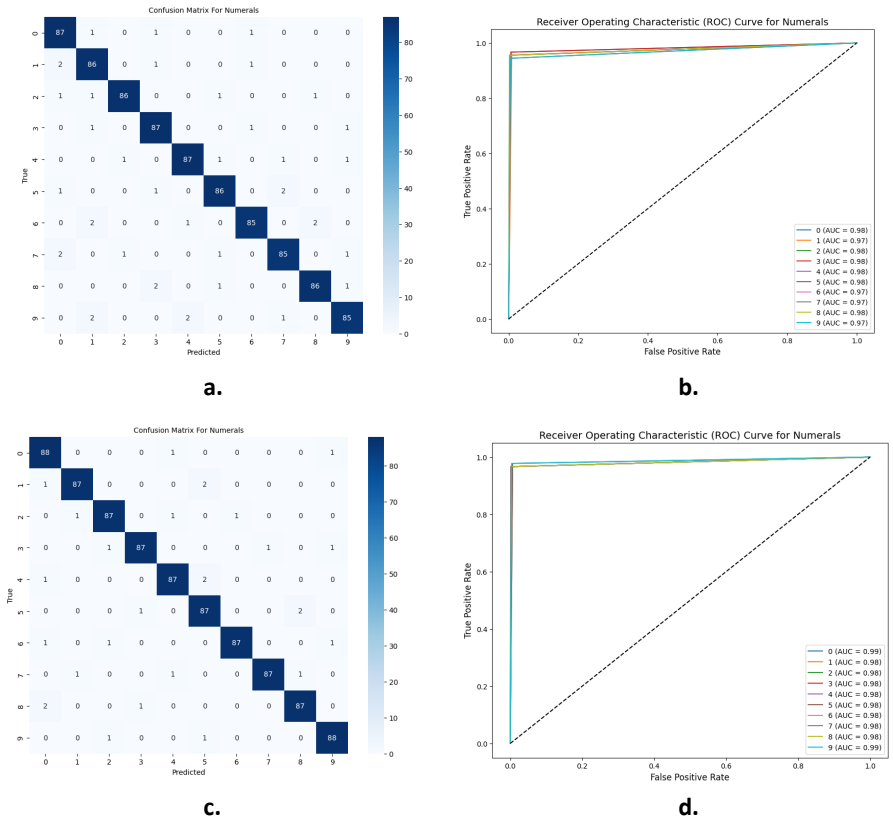

Fig. 7 Confusion matrix and ROC curve for numerals

Table 10 Accuracy for Bengali Vowels Recognition using "BdSL_OPSA22_STATIC1" Dataset

Class	Accuracy	Precision	Recall	F1-Score	Inference Time [ms]
অ (A)	96.8	97.0	96.9	96.9	15.2
আ (Aa)	96.7	96.9	96.6	96.8	15.4
ই (I)	96.4	96.6	96.5	96.5	15.1
উ (U)	96.0	96.2	96.1	96.1	15.5
এ (E)	95.8	96.0	95.7	95.9	15.3
ও (O)	96.2	96.4	96.3	96.3	15.0

Fig. 8 represents the confusion matrix and ROC curves for Vowels of both BdSL_OPSA22_STATIC1 and BdSL_OPSA22_STATIC2 dataset. Fig. 8 a. and Fig. 8. b. are the confusion matrix and ROC curve respectively of the BdSL_OPSA22_STATIC1 dataset for Vowels. Fig. 8 c. and Fig. 8. d. are the confusion matrix and ROC curve respectively of the BdSL_OPSA22_STATIC1 dataset for Vowels.

Table 11 Accuracy for Bengali Vowels Recognition using "BdSL_OPSA22_STATIC2" Dataset

Class	Accuracy	Precision	Recall	F1-Score	Inference Time [ms]
অ (A)	97.5	97.7	97.6	97.6	14.8
আ (Aa)	97.3	97.5	97.4	97.4	15.0
ই/ঈ (I)	97.0	97.2	97.1	97.1	14.9
উ/ঊ (U)	96.8	97.0	96.9	96.9	15.3
এ (E)	97.2	97.4	97.3	97.3	15.2
ও (O)	97.1	97.3	97.2	97.2	14.7

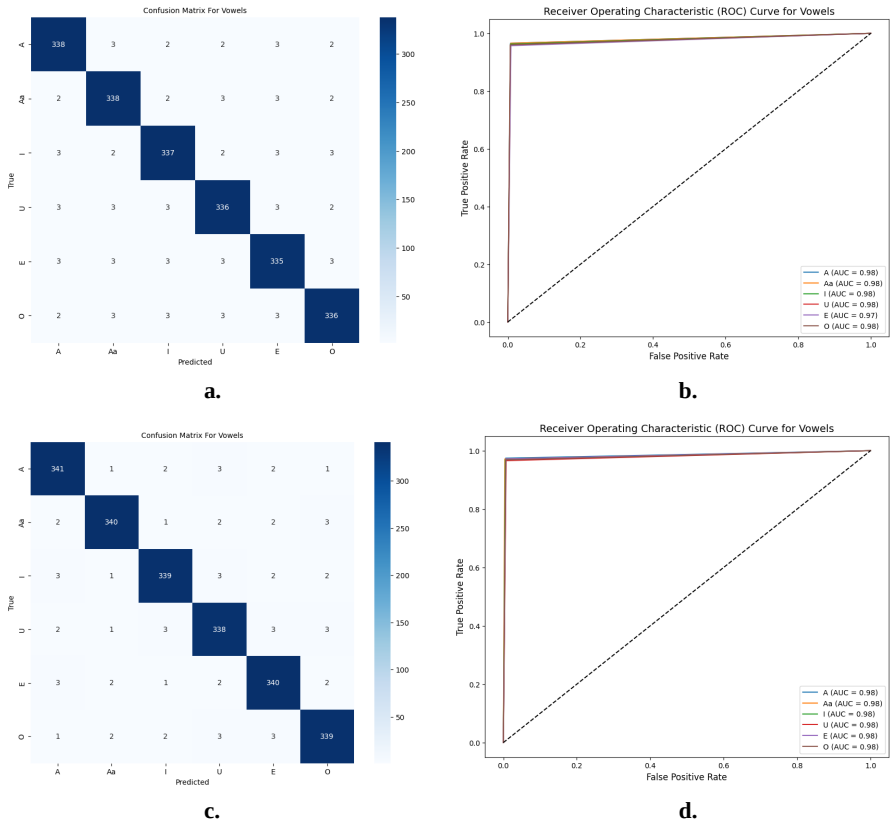

Fig. 8 Confusion matrix and ROC curve for Vowels

Table 12 and 13 depicts the outcome of the accuracy test for identifying specific Bengali consonants using "AdH-CNN" model on "BdSL_OPSA22_STATIC1" and "BdSL_OPSA22_STATIC2" dataset respectively.

Fig. 9 represents the confusion matrix and ROC curves for Consonants of both BdSL_OPSA22_STATIC1 and BdSL_OPSA22_STATIC2 dataset. Fig. 9 a. and Fig. 9. b. are the confusion matrix and ROC curve respectively of the

Table 12 Accuracy for Bengali Consonants Recognition on "BdSL_OPSA22_STATIC1" Dataset

Class	Accuracy	Precision	Recall	F1-Score	Inference Time [ms]
ক (k)	96.4	96.6	96.5	96.5	15.2
খ (kh)	96.2	96.4	96.3	96.3	15.4
গ (g)	96.0	96.2	96.1	96.1	15.3
ঘ (gh)	95.8	96.0	95.9	95.9	15.5
চ (c)	95.7	95.9	95.8	95.8	15.6
ছ (ch)	95.5	95.7	95.6	95.6	15.7
জ (j)	95.4	95.6	95.5	95.5	15.8
ঝ (jh)	95.3	95.5	95.4	95.4	15.9
ট (T)	96.1	96.3	96.2	96.2	16.0
ঠ (Th))	96.0	96.2	96.1	96.1	16.1
ড (D)	95.9	96.1	96.0	96.0	16.2
ঢ (Dh)	95.7	95.9	95.8	95.8	16.3
ত (t)	96.2	96.4	96.3	96.3	16.4
থ (th)	95.6	95.8	95.7	95.7	16.5
দ (d)	95.5	95.7	95.6	95.6	16.6
ঢ (Dh)	95.3	95.5	95.4	95.4	16.7
ন (n)	96.0	96.2	96.1	96.1	16.8
প (p)	96.2	96.4	96.3	96.3	16.9
ফ (ph)	95.9	96.1	96.0	96.0	17.0
ব (b)	95.8	96.0	95.9	95.9	17.1
ভ (bh)	95.7	95.9	95.8	95.8	17.2
ম (m)	96.1	96.3	96.2	96.2	17.3
য় (y)	96.0	96.2	96.1	96.1	17.4
র (r)	95.9	96.1	96.0	96.0	17.5
ল (l)	96.0	96.2	96.1	96.1	17.6
স (s)	95.6	95.8	95.7	95.7	17.7
হ (H)	96.1	96.3	96.2	96.2	17.8
ড় (rh)	95.4	95.6	95.5	95.5	17.9
ং (NNG)	95.8	96.0	95.9	95.9	18.0
ঃ (h)	95.5	95.7	95.6	95.6	18.1

BdSL_OPSA22_STATIC1 dataset for Consonants. Fig. 9 c. and Fig. 9. d. are the confusion matrix and ROC curve respectively of the BdSL_OPSA22_STATIC1 dataset for Consonants.

The robustness of the "AdH-CNN model" was evaluated using the "Ishara-Lipi" dataset. The model's performance was outstanding as demonstrated by its 99.51% accuracy. The outcomes corroborate that "AdH-CNN" generalizes well and performs reliably on external datasets, as demonstrated by its consistently high precision, recall, and F1-score. Table 14 provides comparison of "AdH-CNN" on "BdSL_OPSA22_STATIC1", "BdSL_OPSA22_STATIC2", and "Ishara-Lipi" Datasets. In comparison to the "BdSL_OPSA22_STATIC1" and "BdSL_OPSA22_STATIC2" datasets, the "Ishara-Lipi" dataset is simplified and smaller. While "Ishara-Lipi" comprises approximately 1,800 images with a solid white background, the images are captured with the hands clearly visible and isolated, facilitating the recognition of the

Table 13 Accuracy for Bengali Consonants Recognition on "BdSL_OPSA22_STATIC2" Dataset

Class	Accuracy	Precision	Recall	F1-Score	Inference Time [ms]
ক (k)	97.2	97.4	97.3	97.3	14.9
খ (kh)	97.1	97.3	97.2	97.2	15.1
গ (g)	97.0	97.2	97.1	97.1	15.2
ঘ (gh)	96.9	97.1	97.0	97.0	15.4
চ (c)	96.8	97.0	96.9	96.9	15.3
ছ (ch)	97.3	97.5	97.4	97.4	15.5
জ (j)	96.7	96.9	96.8	96.8	15.6
ঝ (jh)	96.8	97.0	96.9	96.9	15.7
ট (T)	97.1	97.3	97.2	97.2	15.8
ঠ (Th))	97.0	97.2	97.1	97.1	15.9
ড (D)	96.9	97.1	97.0	97.0	16.0
ঢ (Dh)	97.3	97.5	97.4	97.4	16.1
ত (t)	97.0	97.2	97.1	97.1	16.2
থ (th)	96.9	97.1	97.0	97.0	16.3
ড (D)	96.8	97.0	96.9	96.9	16.4
ঢ (Dh)	97.2	97.4	97.3	97.3	16.5
ন (n)	97.1	97.3	97.2	97.2	16.6
প (p)	97.0	97.2	97.1	97.1	16.7
ফ (ph)	96.9	97.1	97.0	97.0	16.8
ব (b)	96.8	97.0	96.9	96.9	16.9
ভ (bh)	96.9	97.1	97.0	97.0	17.0
ম (m)	97.1	97.3	97.2	97.2	17.1
য় (y)	96.9	97.1	97.0	97.0	17.2
র (r)	96.8	97.0	96.9	96.9	17.3
ল (l)	96.7	96.9	96.8	96.8	17.4
স (s)	96.8	97.0	96.9	96.9	17.5
হ (H)	97.1	97.3	97.2	97.2	17.6
ড় (rh)	96.8	97.0	96.9	96.9	17.7
ং (NNG)	96.9	97.1	97.0	97.0	17.8
ঃ (h)	96.8	97.0	96.9	96.9	17.9

Table 14 Performance Comparison of AdH-CNN on BdSL_OPSA22_STATIC1, BdSL_OPSA22_STATIC2, and Ishara-Lipi Datasets

"AdH-CNN"			
Metric	**BdSL_OPSA22_STATIC1**	**BdSL_OPSA22_STATIC2**	**Ishara-Lipi**
Accuracy	96.51%	97.24%	99.51%
Precision	96%	97%	99%
Recall	96%	97%	99%
F1-Score	96%	97%	99%

sign language gestures. The "BdSL_OPSA22_STATIC2" datasets are more complex and varied, with a wider range of data, such as diverse backgrounds, illumination conditions, and static or dynamic gestures. As a result, "AdH-CNN model" outperforms the "Ishara-Lipi" dataset.

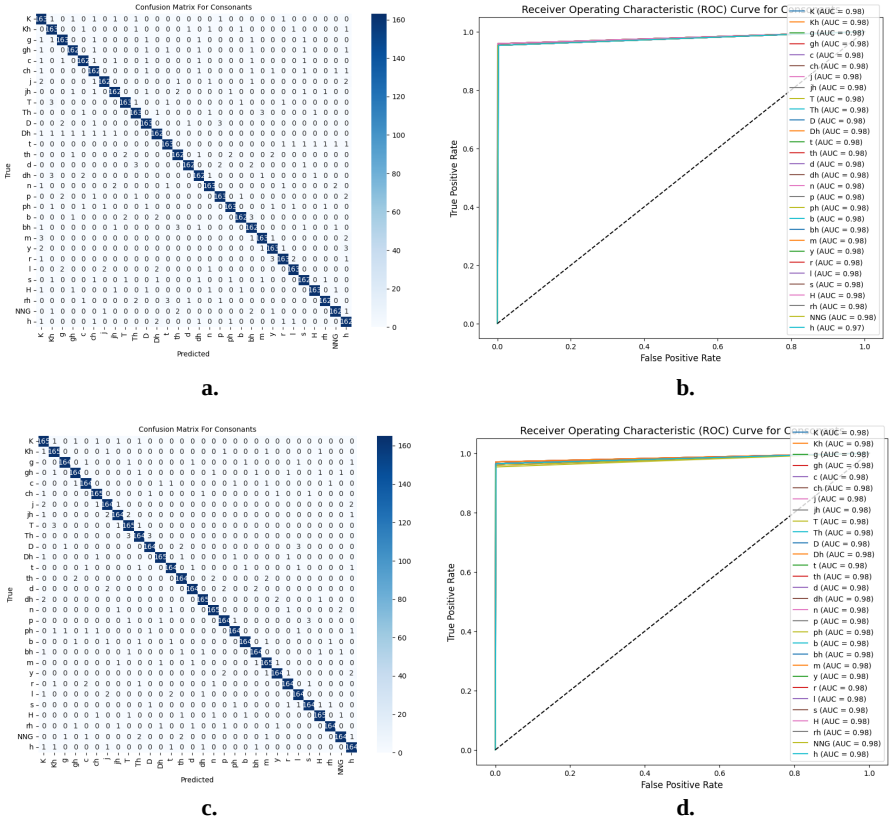

Fig. 9 Confusion matrix and ROC curve for Consonants

4.4 Training and Validation

The training and validation accuracy and loss of the "BdSL_OPSA22_STATIC1" and "BdSL_OPSA22_STATIC2" datasets for the "ConvNeural" architecture are shown in Fig. 10. It appears from "ConvNeural" that, it has more consistent validation accuracy. Furthermore, stability results in a decrease in the loss, indicating the efficacy of the model. This results in improved, dependable, and constant validating accuracy, as well as enhanced stability from our method.

The model achieves high validation accuracy, indicating effective learning and generalization. Early stopping ensures we do not waste computational resources , leading to a well-generalized model suitable for real-world application in sign language recognition. This approach balances the need for sufficient training epochs to learn complex patterns to optimize model performance. Categorical cross-entropy serves as our loss function. We experiment with different batch sizes for the optimizer during the model's training. As the recommended "ConvNeural" architecture is trained on the "BdSL_OPSA22_STATIC1" dataset, the accuracy loss and progression are shown in Fig. 10(a, b). As the recommended "ConvNeural" architecture is trained using the

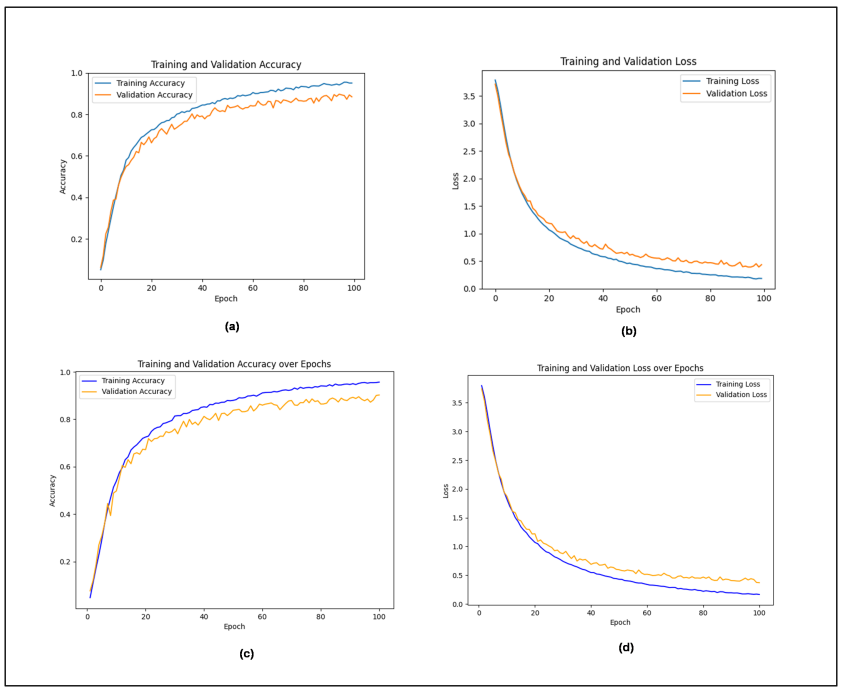

Fig. 10 Training accuracy and validation accuracy, as well as training loss and validation Loss of "ConvNeural" Model on (a) (b) "BdSL_OPSA22_STATIC1" and (c) (d) "BdSL_OPSA22_STATIC2" datasets

"BdSL_OPSA22_STATIC2" dataset, the accuracy loss and progression are shown in Fig. 10(c, d). Even after just the sixth training period, the model had already exceeded 97% training accuracy.

It is evident that the model's ideal convergence is contingent upon alpha's initial learning rate. We have to pick alpha carefully because if alpha is too high, CNN does not converge effectively. CNN convergence takes longer if alpha is smaller. To mitigate these issues, we have employed the default alpha of 0.001. Even after just the second training session, the model had already exceeded 96% training accuracy.

4.5 Result Analysis

We can compare and observe that our proposed framework outperforms the state-of-the-art methods both in terms of dataset handling and model performance. Specifically, Table 15 provides a detailed comparison of the models, showcasing the accuracy achieved by previously existing works compared to our model, "AdH-CNN". The table highlights the improvement in model efficiency. We have provided a more diverse dataset with different lighting conditions and various backgrounds with diverse of participants compared to other datasets, for instance, [45, 47, 49]. This demonstrates that our approach not only achieves superior accuracy but also enhances the overall robustness and adaptability of the model, addressing limitations found in prior

Table 15 Comparison of the proposed framework with the other state of art methods

References	Methods	No. of Samples	No. of Classes	Accuracy (%)
Islam et al. [40]	CNN	Ishara-Lipi(1800 samples)	36 characters	92%
Abedin et al. [5]	Concatenated BdSL (CNN image network + pose estimation)	12,581 samples	38 alphabets	91.51%
Rafi et al. [41]	VGG19	12,581 samples	38 alphabets	89.60%
Hossen et al. [42]	Deep CNN	1147 samples	37 characters	84.68%
Shanta et al. [43]	Scale Invariant Feature Transform (SIFT) and CNN	1700 samples	38 characters	90.63%
Basnin et al. [3]	Integrated CNN and LSTM	13,400 samples	36 alphabets	88.5%
Khan et al. [44]	CNN and customized ROI Segmentation	500 samples	5 sign gestures	94%
Ahmed et al. [45]	CNN	3200 samples	10 numerals	92%
Rony et al. [46]	CNN, Inception-v3	2050 samples	38 alphabets + 3 special	92.85%
Tabassum et al. [47]	HOG features and KNN	1400 samples	35 alphabets	91.1%
Begum et al. [48]	PCA	2020 samples	6 Bengali Vowels and 10 Numbers	76.90%
Uddin et al. [49]	SVM	4800 samples	32 Arabic signs and alphabets	95%
Yadav et al. [50]	Modified LeNet-5 (MLCNN8)	92,000 samples	46 Devanagari characters	99.21%
Rubaiyeat et al.[51]	Attention-based Bi-LSTM	9307 samples	60 BdSL words	75.01%
Shahgir et al.[52]	3D CNN + Graph Neural Network	611 videos	40 BdSL words	89.00(F1)%
Proposed method	"AdH-CNN"	"BdSL_OPSA22_STATIC1" (24615 samples)	46 Bangla Numeric and characters	96.51%
Proposed method	"AdH-CNN"	"BdSL_OPSA22_STATIC2" (8437 samples)	46 Bangla Numeric and characters	97.24%

frameworks. These results underscore the effectiveness of "AdH-CNN" in processing complex datasets and outperforming established methodologies in the field.

Fig. 11 shows the comparison between the proposed framework and state-of-the-art methods.

The Bias-Variance Tradeoff Plot for the AdH-CNN model is represented in the Fig. 12 to show the generalizability of the proposed model which Interpretation is described below:

- **Training and Validation Accuracies are Close:** For both BdSL_OPSA22_STATIC1 and BdSL_OPSA22_STATIC2, training and validation accuracies align well, indicating low variance (not overfitting). This suggests that the model generalizes well across different dataset splits.

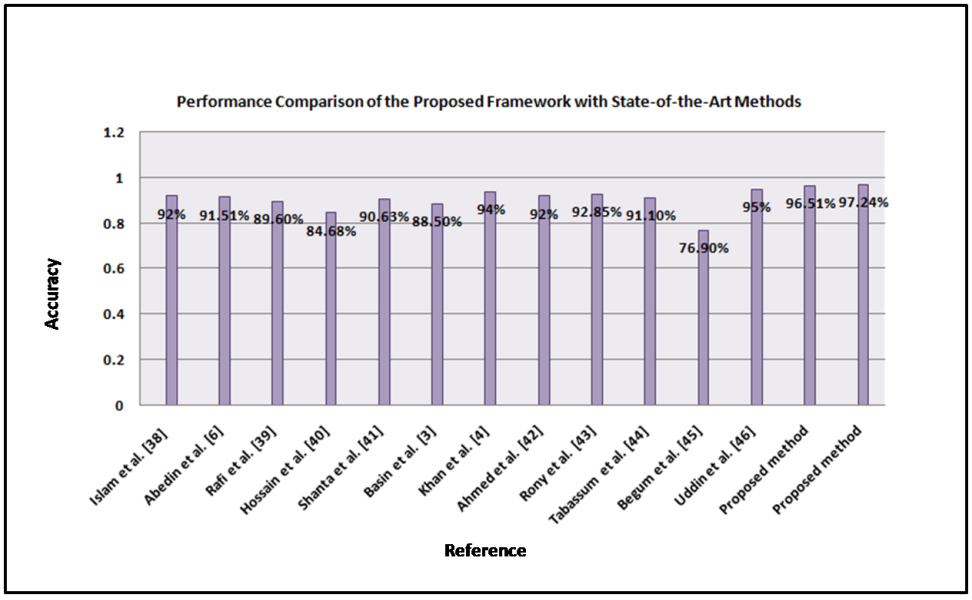

Fig. 11 Performance Comparison of the Proposed Framework with State-of-the-Art Methods

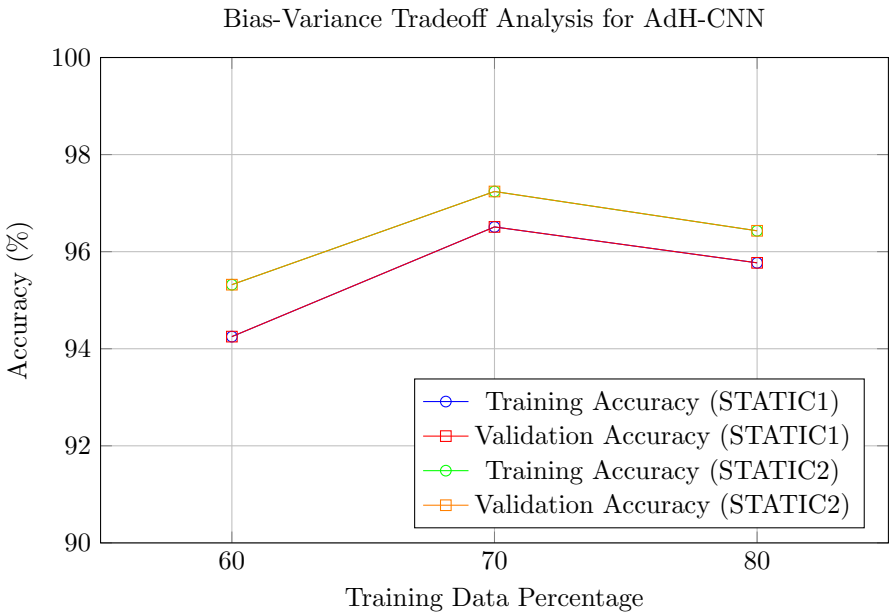

Fig. 12 Bias-Variance Tradeoff Analysis for AdH-CNN Model

- **Stable Performance Across Training Sizes:** A 70% training split (optimal point) achieves the highest accuracy (96.51% for BdSL_OPSA22_STATIC1 and 97.24% for BdSL_OPSA22_STATIC2). Using 80% training data does not significantly improve performance, suggesting a saturation point in learning.
- **No Signs of Underfitting:** Even with 60% training data, the model achieves over 94% accuracy, indicating that bias is low (not underfitting).

Table 16 5-Fold Cross-Validation Results for AdH-CNN Model

Dataset	Fold 1	Fold 2	Fold 3	Fold 4	Fold 5	Mean Accuracy (± Std)
BdSL_OPSA22_STATIC1	95.9%	96.2%	96.7%	96.4%	96.5%	**96.34% ± 0.3%**
BdSL_OPSA22_STATIC2	96.8%	97.1%	97.5%	97.3%	97.0%	**97.14% ± 0.3%**

Table 16 presents the 5-fold cross-validation results for the AdH-CNN model on the BdSL_OPSA22_STATIC1 and BdSL_OPSA22_STATIC2 datasets. The outcomes show how stable and broadly applicable the model is to various data splits. The accuracy for the BdSL_OPSA22_STATIC1 dataset was constant with little volatility, ranging from 95.9% to 96.7% over the five folds, with a mean accuracy of 96.34% ± 0.3%. Similarly, the BdSL_OPSA22_STATIC2 dataset had a mean accuracy of 97.14% ± 0.3%, with somewhat greater accuracy ranging from 96.8% to 97.5%. The AdH-CNN model significantly lowers the danger of overfitting by maintaining strong predictive skills across various data partitions, as indicated by the low standard deviation in both datasets. These results demonstrate that the model is appropriate for practical Bangla Sign Language recognition and generalizes effectively to unseen data.

5 Conclusion and Future Directions

5.1 Conclusion

This study sought to advance the field of Bangla Sign Language Recognition (BdSL) by addressing some of the limitations present in prior research. To this end, we proposed the Advanced Hybrid Convolutional Neural Network (AdH-CNN), which combines the strengths of VGG16 and ResNet50. The model was evaluated on two proprietary datasets, "BdSL_OPSA22_STATIC1" and "BdSL_OPSA22_STATIC2", achieving classification accuracies of 96.51% and 97.24%, respectively. These results demonstrate the potential of the AdH-CNN model to improve the recognition of Bangla vowels, letters, and numbers. However, while the results are encouraging, it is important to acknowledge that the performance metrics do not fully capture the practical challenges associated with BdSL recognition. The higher accuracy on "BdSL_OPSA22_STATIC1" is influenced by the dataset's predominantly solid backgrounds, whereas the performance on "BdSL_OPSA22_STATIC2" reflects the difficulties of recognizing signs in more complex settings. Thus, the current model, while effective under specific conditions, still faces challenges in broader and more diverse environments. The AdH-CNN model demonstrates significant promise in enhancing accessibility for individuals utilizing Bangla Sign Language (BdSL). This technology can be incorporated into mobile

applications, facilitating immediate translation of BdSL into written or spoken language, thus providing advantages for individuals with hearing disabilities. The model can additionally be integrated into digital kiosks or wearable devices to facilitate smooth communication. Educational establishments and organizations can implement it for engaging learning experiences, whereas companies can utilize it for automated sign language translation. Furthermore, the integration of BdSL sign recognition into social media platforms and virtual assistants can significantly improve accessibility in digital communication. The AdH-CNN model holds significant promise for transforming communication methods with BdSL.

5.2 Limitations and Future Work

While the AdH-CNN model demonstrated promising results, several limitations highlight the need for further research. One notable challenge is the reduced accuracy for signs with similar hand postures, indicating that the model may struggle to differentiate between closely related gestures. For example, signs of র(r), ল(l) are similar. Thus, it sometimes classify incorrectly. Fig. 13 illustrates some images of correct and incorrect classifications. We have conducted a comprehensive analysis of error cases to identify patterns in misclassification, in order to address this. Our analysis demonstrated that the majority of errors were observed in visually similar signs, particularly those with minor variations in finger positioning or hand orientation. Furthermore, specific signs exhibited overlapping feature representations in the embedding space, which resulted in classification confusion. Additionally, the datasets used in this study have inherent limitations. The "BdSL_OPSA22_STATIC1" dataset predominantly features solid backgrounds, which may not fully represent real-world scenarios, while the "BdSL_OPSA22_STATIC2" dataset, though more complex, lacks sufficient diversity and volume to ensure generalizability across varied environments. Furthermore, the focus of this research was confined to static images, leaving the dynamic and contextual aspects of sign language unexplored.

To address these limitations, future work will prioritize expanding the "BdSL_OPSA22_STATIC2" dataset to include a broader range of backgrounds and real-world complexities. Improving the model's ability to distinguish between signs with similar postures will also be a key focus, potentially through advanced feature extraction techniques or augmented training data. Moreover, extending the system to handle dynamic gesture recognition by incorporating temporal models, such as LSTMs or Transformers, will enable the processing of real-time video input. Finally, evaluating the system's usability in practical environments and assessing its impact on communication accessibility will be essential steps toward developing a more robust and inclusive SLR solution. The integration of real-time systems with hardware optimization and the exploration of advanced 5G and cloud infrastructures will further facilitate the seamless recognition of sign language, thereby increasing the accessibility of individuals who rely on sign language for communication.

Fig. 13 Example images of correct vs incorrect classifications

Acknowledgments

We would like to express our sincere gratitude to the 14 participants who granted us permission to include their images and data in our dataset. They willingly took part in our study by reviewing and signing our ethical consent form, allowing us to capture their photographs. During the consent process, we ensured that all participants were fully informed about the publication of the data and its availability for other researchers. The photographs used in our dataset and featured in our paper as examples were taken using our own equipment. This work was supported in part by the Center for Research, Innovation, and Transformation (CRIT) of the Green University of Bangladesh (GUB) under grant No.G-23-22-28.

Authors Contributions

All authors contributed equally to this work. All the authors also diligently reviewed the article.

Declaration of Generative AI and AI-assisted Technologies in the Writing Process:

During the preparation of this work the authors utilized Grammarly and ChatGPT in order to improve language and readability. After using this tool/service, the authors reviewed and edited the content as needed and took full responsibility for the publication's content.

Data Availability

The datasets generated during and/or analyzed during this study are available from the corresponding author upon reasonable request.

Declarations

- There is no conflict of interest/competing interests in our paper.
- Our 14 participants willingly participated in our research study by carefully reviewing and signing our ethical statement.
- During the consent process, we ensured that all participants were well-informed about the potential publication of the data and its accessibility to other researchers.

References

[1] Debnath, T., Reza, M.M., Rahman, A., Beheshti, A., Band, S.S., Alinejad-Rokny, H.: Four-layer convnet to facial emotion recognition with minimal epochs and the significance of data diversity. Scientific Reports **12**(1), 6991 (2022)

[2] Chowdhury, P.K., Oyshe, K.U., Rahaman, M.A., Debnath, T., Rahman, A., Kumar, N.: Computer vision-based hybrid efficient convolution for isolated dynamic sign language recognition. Neural Computing and Applications, 1–16 (2024)

[3] Basnin, N., Nahar, L., Hossain, M.S.: An integrated cnn-lstm model for bangla lexical sign language recognition. In: Proceedings of International Conference on Trends in Computational and Cognitive Engineering: Proceedings of TCCE 2020, pp. 695–707 (2020). Springer

[4] Khan, M.S.I., Rahman, A., Debnath, T., Karim, M.R., Nasir, M.K., Band, S.S., Mosavi, A., Dehzangi, I.: Accurate brain tumor detection using deep convolutional neural network. Computational and structural biotechnology journal **20**, 4733–4745 (2022)

[5] Abedin, T., Prottoy, K.S., Moshruba, A., Hakim, S.B.: Bangla sign language recognition using concatenated bdsl network. arXiv preprint arXiv:2107.11818 (2021)

[6] Noble, W.S.: What is a support vector machine? Nature biotechnology **24**(12), 1565–1567 (2006)

[7] Sikder, J., Das, U.K., Chakma, R.J.: Supervised learning-based cancer detection. International Journal of Advanced Computer Science and Applications **12**(5) (2021)

[8] Rahaman, M.A., Oyshe, K.U., Chowdhury, P.K., Debnath, T., Rahman, A., Khan, M.S.I.: Computer vision-based six layered convneural network to recognize

sign language for both numeral and alphabet signs. Biomimetic Intelligence and Robotics **4**(1), 100141 (2024)

[9] John Doe, A.J. Jane Smith: Corrnet+: Sign language recognition and translation via spatial-temporal correlation. CorrNet+ (2024)

[10] Alice Brown, C.G. Bob White: Evsign: Sign language recognition and translation with streaming events. EvSign (2024)

[11] John Doe, A.J. Jane Smith: Towards online continuous sign language recognition and translation. Online Sign (2024)

[12] O'Shea, K., Nash, R.: An introduction to convolutional neural networks. ArXiv Preprint ArXiv:1511.08458 (2015)

[13] Wu, J.: Introduction to convolutional neural networks. National Key Lab for Novel Software Technology. Nanjing University. China **5**(23), 495 (2017)

[14] Das, S., Imtiaz, M.S., Neom, N.H., Siddique, N., Wang, H.: A hybrid approach for bangla sign language recognition using deep transfer learning model with random forest classifier. Expert Systems with Applications **213**, 118914 (2023)

[15] Parmar, A., Katariya, R., Patel, V.: A review on random forest: An ensemble classifier. In: International Conference on Intelligent Data Communication Technologies and Internet of Things (ICICI) 2018, pp. 758–763 (2019). Springer

[16] Khatawate, P., Shetty, S., Shirisha, K., Patil, P.: A comprehensive analysis of vgg16 and resnet50 models in sign language recognition. In: 2024 5th International Conference for Emerging Technology (INCET), pp. 1–6 (2024). IEEE

[17] Islam, M.M., Uddin, M.R., AKhtar, M.N., Alam, K.R.: Recognizing multiclass static sign language words for deaf and dumb people of bangladesh based on transfer learning techniques. Informatics in Medicine Unlocked **33**, 101077 (2022)

[18] Tammina, S.: Transfer learning using vgg-16 with deep convolutional neural network for classifying images. International Journal of Scientific and Research Publications (IJSRP) **9**(10), 143–150 (2019)

[19] Shaha, M., Pawar, M.: Transfer learning for image classification. In: 2018 Second International Conference on Electronics, Communication and Aerospace Technology (ICECA), pp. 656–660 (2018). IEEE

[20] Alom, M.Z., Taha, T.M., Yakopcic, C., Westberg, S., Sidike, P., Nasrin, M.S., Esesn, B.C.V., Awwal, A.A.S., Asari, V.K.: The history began from alexnet: A comprehensive survey on deep learning approaches. arXiv preprint arXiv:1803.01164 (2018)

[21] Xia, X., Xu, C., Nan, B.: Inception-v3 for flower classification. In: 2017 2nd

International Conference on Image, Vision and Computing (ICIVC), pp. 783–787 (2017). IEEE

[22] Rao, G.A., Syamala, K., Kishore, P., Sastry, A.: Deep convolutional neural networks for sign language recognition. In: 2018 Conference on Signal Processing and Communication Engineering Systems (SPACES), pp. 194–197 (2018). IEEE

[23] Yasir, F., Prasad, P., Alsadoon, A., Elchouemi, A., Sreedharan, S.: Bangla sign language recognition using convolutional neural network. In: 2017 International Conference on Intelligent Computing, Instrumentation and Control Technologies (ICICICT), pp. 49–53 (2017). IEEE

[24] Weichert, F., Bachmann, D., Rudak, B., Fisseler, D.: Analysis of the accuracy and robustness of the leap motion controller. Sensors **13**(5), 6380–6393 (2013)

[25] Eddy, S.R.: What is a hidden markov model? Nature biotechnology **22**(10), 1315–1316 (2004)

[26] Zaki, M.M., Shaheen, S.I.: Sign language recognition using a combination of new vision based features. Pattern Recognition Letters **32**(4), 572–577 (2011)

[27] Kasukurthi, N., Rokad, B., Bidani, S., Dennisan, D., et al.: American sign language alphabet recognition using deep learning. arXiv preprint arXiv:1905.05487 (2019)

[28] Shamsaldin, A.S., Fattah, P., Rashid, T.A., Al-Salihi, N.K.: A Study of the Applications of Convolutional Neural Networks. [vol., no., pp. if available] ([year])

[29] Banskota, N., Alsadoon, A., Prasad, P.W.C., Dawoud, A., Rashid, T.A., Alsadoon, O.H.: A novel enhanced convolution neural network with extreme learning machine: facial emotional recognition in psychology practices. Multimedia Tools and Applications **82**(5), 6479–6503 (2023)

[30] Rashid, T.A.: Convolutional neural networks based method for improving facial expression recognition. In: Intelligent Systems Technologies and Applications 2016, pp. 73–84. Springer, ??? (2016)

[31] Meena, G., Mohbey, K., Lokesh, K.: Fstl-sa: Few-shot transfer learning for sentiment analysis from facial expressions. Multimedia Tools and Applications, 1–29 (2024) https://doi.org/10.1007/s11042-024-20518-y

[32] Hasan, M.M., Srizon, A.Y., Sayeed, A., Hasan, M.A.M.: Classification of sign language characters by applying a deep convolutional neural network. In: 2020 2nd International Conference on Advanced Information and Communication Technology (ICAICT), pp. 434–438 (2020). IEEE

[33] Bird, J.J., Ekárt, A., Faria, D.R.: British sign language recognition via late

fusion of computer vision and leap motion with transfer learning to american sign language. Sensors **20**(18), 5151 (2020)

[34] Rahaman, M.A., Jasim, M., Ali, M.H., Hasanuzzaman, M.: Real-time computer vision-based bengali sign language recognition. In: 2014 17th International Conference on Computer and Information Technology (ICCIT), pp. 192–197 (2014). IEEE

[35] Guo, G., Wang, H., Bell, D., Bi, Y., Greer, K.: Knn model-based approach in classification. In: On The Move To Meaningful Internet Systems 2003: CoopIS, DOA, And ODBASE: OTM Confederated International Conferences, CoopIS, DOA, And ODBASE 2003, Catania, Sicily, Italy, November 3-7, 2003. Proceedings, pp. 986–996 (2003)

[36] Islam, S., Sara, U., Kawsar, A., Rahman, A., Kundu, D., Dipta, D.D., Karim, A., Hasan, M., et al.: Sgbba: An efficient method for prediction system in machine learning using imbalance dataset. International Journal of Advanced Computer Science and Applications **12**(3) (2021)

[37] Oyshe, P.: Bangla sign language dataset "BdSL_OPSA22_STATIC1" (2023). https://github.com/Prothoma2001/Bangla-Sign-Language-Recognition-Using-CNN/tree/main/BdSL_OPSA22_STATIC1

[38] Oyshe, P.: Bangla sign language dataset "BdSL_OPSA22_STATIC2" (2023). https://github.com/Prothoma2001/Bangla-Sign-Language-Recognition-Using-CNN/tree/main/BdSL_OPSA22_STATIC2

[39] Das, U.K., Sikder, J., Datta, N., Chakraborty, P.: Bengali handwritten equation solving system. Journal of King Saud University-Computer and Information Sciences **36**(3), 101997 (2024)

[40] Islam, M.S., Mousumi, S.S.S., Jessan, N.A., Rabby, A.S.A., Hossain, S.A.: Isharalipi: The first complete multipurpose open access dataset of isolated characters for bangla sign language. In: 2018 International Conference on Bangla Speech and Language Processing (ICBSLP), pp. 1–4 (2018). IEEE

[41] Rafi, A.M., Nawal, N., Bayev, N.S.N., Nima, L., Shahnaz, C., Fattah, S.A.: Image-based bengali sign language alphabet recognition for deaf and dumb community. In: 2019 IEEE Global Humanitarian Technology Conference (GHTC), pp. 1–7 (2019)

[42] Hossen, M., Govindaiah, A., Sultana, S., Bhuiyan, A.: Bengali sign language recognition using deep convolutional neural network. In: 2018 Joint 7th International Conference on Informatics, Electronics & Vision (icIEV) and 2018 2nd International Conference on Imaging, Vision & Pattern Recognition (icIVPR),

pp. 369–373 (2018). IEEE

[43] Shanta, S.S., Anwar, S.T., Kabir, M.R.: Bangla sign language detection using sift and cnn. In: 2018 9th International Conference on Computing, Communication and Networking Technologies (ICCCNT), pp. 1–6 (2018). IEEE

[44] Khan, S.A., Joy, A.D., Asaduzzaman, S., Hossain, M.: An efficient sign language translator device using convolutional neural network and customized roi segmentation. In: 2019 2nd International Conference on Communication Engineering and Technology (ICCET), pp. 152–156 (2019). IEEE

[45] Ahmed, S., Islam, M., Hassan, J., Ahmed, M.U., Ferdosi, B.J., Saha, S., Shopon, M., et al.: Hand sign to bangla speech: A deep learning in vision based system for recognizing hand sign digits and generating bangla speech. arXiv preprint arXiv:1901.05613 (2019)

[46] Rony, A.J., Saikat, K.H., Tanzeem, M., Robi, F.R.H.: An effective approach to communicate with the deaf and mute people by recognizing characters of one-hand bangla sign language using convolutional neural-network. In: 2018 4th International Conference on Electrical Engineering and Information & Communication Technology (iCEEiCT), pp. 74–79 (2018). IEEE

[47] Tabassum, T., Mahmud, I., Uddin, M., Emran, A., Afjal, M., Nitu, A.: Enhancement of single-handed bengali sign language recognition based on hog features (2020)

[48] Begum, S., Hasanuzzaman, M.: Computer vision-based bangladeshi sign language recognition system. In: 2009 12th International Conference on Computers and Information Technology, pp. 414–419 (2009). IEEE

[49] Uddin, M.A., Chowdhury, S.A.: Hand sign language recognition for bangla alphabet using support vector machine. In: 2016 International Conference on Innovations in Science, Engineering and Technology (ICISET), pp. 1–4 (2016). IEEE

[50] Yadav, B., Indian, A., Meena, G.: Recognizing off-line devanagari handwritten characters using modified lenet-5 deep neural network. Procedia Computer Science **235**, 799–809 (2024)

[51] Rubaiyeat, H.A., Mahmud, H., Habib, A., Hasan, M.K.: Bdslw60: A word-level bangla sign language dataset. arXiv preprint arXiv:2402.08635 (2024)

[52] Shahgir, H.S., Sayeed, K.S., Tahmid, M.T., Zaman, T.A., Alam, M.Z.U.: Connecting the dots: Leveraging spatio-temporal graph neural networks for accurate bangla sign language recognition. arXiv preprint arXiv:2401.12210 (2024)