International Journal of Scientific Engineering and Research (IJSER)
ISSN (Online): 2347-3878
SJIF (2025): 8.036

Analysis of Vision Transformers for American Sign Language Recognition in Unconstrained Visual Conditions

Satwinder Kaur[1], Garima Joshi[2], Sumit Budhiraja[3], Sunil Agrawal[4]

[1]Department of Electronics and Communication Engineering, UIET, Panjab University, Chandigarh, 160014, Punjab, India
Corresponding Author Email: *satwinderkaur1219[at]gmail.com*

[2]Department of Electronics and Communication Engineering, UIET, Panjab University, Chandigarh, 160014, Punjab, India
Contributing Author: *joshi garima5[at]yahoo.com*

[3]Department of Electronics and Communication Engineering, UIET, Panjab University, Chandigarh, 160014, Punjab, India
Email: *sumit[at]pu.ac.in*

[4]Department of Electronics and Communication Engineering, UIET, Panjab University, Chandigarh, 160014, Punjab, India
Email: *s.agrawal[at]hotmail.com*

These authors contributed equally to this work.

Abstract: ***Sign language is a spoken language for deaf people, but it becomes a tedious task for us to learn this language, thus creating a communication gap between them and us. The solution for this problem is automatic sign language recognition (SLR) systems that focus on translating hand gestures into their linguistic representations. But it is challenging for these systems to work in real-time due to noise in backgrounds, lighting effects, and having unbalanced datasets. The motivation of this research is to address these challenges with the help of Vision Transformer (ViT) architectures. We focus on static American Sign Language (ASL) benchmark datasets having different levels of complex backgrounds and human-induced noise. In this proposed work, stratified K-fold cross-validation and early stopping are used to reduce overfitting and tackle class imbalance, thus promoting model robustness. From experimental analysis, the ViT-based approach shows strong validation performance but notable generalization gaps when evaluated on inference datasets with a restricted variety or gesture classes that are similar in appearance. Although this is a limitation for deployment in real-time, findings reveal the importance of designing human-noise datasets that are deployment-oriented. This work describes the strengths and limitations of attention-based models and provides insights for designing a more robust framework for assistive communication applications.***

Keywords: Sign Language Recognition, Vision Transformer, American Sign Language, Hand Gesture Recognition, Assistive Communication

1. Introduction

There are about 2.5 billion people who can be affected by hearing loss by 2050, so sign language recognition (SLR) has become a necessity for assistive communication between the deaf and the hearing population [1]. Despite significant advances in computer vision–based methods, the design of robust SLR systems remains problematic due to the requirement of well-curated datasets that capture the enormous range of hand shapes, orientations, and signing styles prevalent in everyday scenarios. In an image-based SLR, linguistic information is transmitted through spatial hand configurations that are sensitive to background and ambient changes. There are different types of Sign Language as it vary from country to country. American Sign Language (ASL) is easily accessible due to its uniform structure and available datasets [2]. Current techniques for ASL are based on image datasets that are captured under controlled conditions, mostly on uniform backgrounds. Although these datasets are easily trainable, they fail to accurately work in real-time. Thus, the practical deployment of these SLR systems has become difficult, enhancing intra-class variability and decreasing recognition accuracy. Despite introducing ASL image datasets to facilitate research in this field, a clear imbalance has been noted in the literature between studies addressing datasets with uniform-background and cluttered environments. Also, a significant gap remains in research on actual visual-based ASL recognition. This gap can only be filled with the goal of generating reliable and versatile SLR systems that can support real-world communication.

The primary efforts of this work are mentioned below:

- Vision Transformer architecture is evaluated for image-based ASL recognition across multiple benchmark datasets with uniform and cluttered backgrounds.
- Stratified cross-validation with early stopping has been used to tackle class imbalance and facilitate generalization.
- The behavior of human noise and cluttered background in images has been analyzed to detect static sign language.
- Validating the proposed approach to support communication among hard-of-hearing and hearing community.

The work is presented as follows: Section 2 covers related work, while Section 3 dives into the datasets. Section 4 focus on proposed methodology. In Section 5, highlights experimental results. Further the work is summarized and

Licensed Under Creative Commons Attribution CC BY

concluded in Section 6, and finally topics and proposals for further investigation has been discussed.

2. Related Work

This discussion of the literature has offered various approaches in the recognition of hand gesture recognition (HGR), and they have been instrumental in enhancing the accuracy of the results in various datasets of sign language. In the past few years, American Sign Language (ASL) has made significant advancements with the standardized vocabulary and the availability of datasets [2]. In the same way, in order to enhance the development of other sign languages, it is necessary to extract statistical information from the existing datasets and test their performance with various detection models [3]. A system has been proposed by Roumiassa et al., in which the textural descriptors are applied for static and dynamic gestures in ASL and ArSL, where the performance of the Mi-LOOP is better than the existing methods but is dependent on the accurate segmentation of the hands [4]. Singla et al. have worked on the Indian Sign Language using various machine and deep learning approaches and obtained satisfactory results with the datasets but did not perform well in the real-world scenario [5]. HyFiNet is an efficient model for low-resource scenarios but has yet to be tested in more diverse scenarios [6]. mIV3Net using the InceptionV3 model has shown some promise in tackling complex scenes but has only been tested on one complex scene [7]. Attia et al. used attention mechanisms with YOLOv5x to enhance detection speed and accuracy, but datasets does not account for broad variability in sign language [8]. Bhatt and Dash proposed a Reinforcement Learning based Adaptive Hand gesture recognition and verification system that achieved accuracies of 99.2% and 98.2% on ASL and 20BN-jester datasets, respectively [9]. Keskin et al. created a new dataset named THOS and used hybrid model consisting of transformer decoder and biLSTM model and achieved test accuracy of 79.31% [10]. In addition to the previously stated existing method of SLRS, Table 1 provides a quick comparison of studies on deep learning-based sign language recognition. Even though some deep learning approaches outperformed traditional methods, there is still a lot of room for improvement in terms of how well they work in many aspects. The current methods have limited ability to identify subtle differences between classes and hard-to-distinguish indicators. A deep feature concatenation was used by researchers to improve recognition accuracy [11]. Subsequently, many transfer learning and data augmentation techniques are used to eradicate challenges due to limited training data, so that models are able generalize better across different conditions and user variations. Many CNN-based models performed well, however their dependancy on large labeled datasets and computational power hinders practical deployment [12]. This gives a glimpse of Vision Transformers (ViTs) that employ self-attention mechanisms that are capable of capturing long-range dependencies in images, even in cluttered background conditions. [13]. Thus, the present study uses a Vision Transformer that has been finetuned for American sign language recognition in complex environments. This methodology bridges the gaps in the existing methodologies and improves the accuracy of the recognition process in comparison to the existing ones. Accordingly, in the present study, datasets are used to recognize the sign language gestures in real-world conditions to ensure robust performance of the SLR systems.

3. Statistical Insights into Datasets

The first and foremost step is to analyze existing datasets, examining their strengths and limitations to understand what constitutes a standard dataset. With this

Table 1: Brief comparison of Sign Language Recognition Systems

Author & Year	Type of Sign Language	Gesture Type	Dataset and its' complexity	No. of classes	Model	Accuracy (%)
Adithya and Rajesh 2020 [14]	ASL	Single-handed	NUS II (Complex)	10	CNN	94.7
Joshi et al. 2020 [15]	ISL	Double-handed	Self-made (Complex)	26	Taguchi-TOPSIS	92
Palanisamy and Thangaswamy 2022 [16]	ASL	Single-handed	Jochen Triesch 10 Alphabets (Different lighting conditions) NUS II (Complex)	10, 10	CNN-LSTM	97.86, 98.32
Kasapbasi et al.2022 [17]	ASL	Single-handed	Self-made (Different lighting conditions)	26	CNN	99.38
Amin et al. 2021 [18]	BSL	Single-handed	BdSL	28	Transfer Learning+ZSL	93.68
Martin and Espejo 2021 [19]	SSL	Single-handed	LSE (Complex)	18	Deep transfer learning (VGG16, ResNet, EfficientNet)	96.42
Liang et al. 2023 [20]	CSL	Single-handed	ELRHG (Simple), ANHG (Simple)	10, 5	MMFANet	93.4 mAP, 99.1 mAP
Alharthi and Alzahrani 2023 [21]	ArSL	Single-handed	ArSL dataset (Different lighting conditions)	32	ResNet and InceptionResNet	98
Rajalakshmi et al. 2022 [18]	ArSL	Single-handed	ArASL (Simple)	32	HNN-ISLR	99.76
Gao et al. 2022 [22]	CSL	Single-handed	Self-made (Simple)	30	CNN-CB	94.88
Podder et al. 2022 [23]	BdSL	Both	BdSL-D1500 (mask generated)	87	Transfer Learning (ResNet18)	99.99
Ma et al. 2022 [24]	ASL	Single-handed	ASL (Simple), ASLA (Complex)	29	Ensemble Learning + Classification: ARS-MA	98.83
Das et al. 2023 [25]	BdSL	Both	BdSL (Background elimination	50	Deep transfer learning +	91.67

Licensed Under Creative Commons Attribution CC BY

			done)		Random Forest Classifier	
Kumar and Jaison 2023 [26]	ASL, ISL	Both	ASL (Simple), ISL (Simple)	29, 26	MobileNetV2	98.77
Nandi et al. 2023 [27]	ISL	Both	Self-made (Simple)	26	CNN + DiffGrad optimizer	99.76
Rahaman et al. 2024 [28]	BdSL	Double-handed	Self-made (Complex),BdSL 1, BdSL 2	46	ConvNeural	98.38, 92.78
Awaluddin et al. 2024 [29]	ASL	Single-handed	MUS, NUS II, Sebastian Marcel, HG14 (Made Complex)	26,10, 6,14	Deep learning based Green Screen Technique	80
Alsolai et al. 2024 [30]	ASL	Single-handed	ASL Kaggle (Simple)	29	Reptile Search Algorithm	99.43
Kolikipogu et al. 2024 [31]	ISL	Double-handed	ISL Kaggle (Simple)	26	VIT	99
Altaher et al. 2025 [32]	ASL	Single-handed	ASL (Simple)	26	Vision Mamba	99.98
This work	ASL	Single-handed	TD (Most Complex)	12	VIT + Stratified K-Fold Cross Validation	97.93

perspective, we have considered ASL image datasets with complex backgrounds. By incorporating insights from ASL uniform datasets, the cluttered datasets aim to bridge the gap in the availability of high-quality ASL datasets with human noise, providing a strong foundation for advancing research in this domain. The datasets utilized in this paper are discussed in detail here, and sample images of all datasets are shown in Figures 1 to 5.

- **NUS II dataset** is created by Pisharady et al.[33] at the National University of Singapore (NUS) against complex backgrounds. This dataset shows hand postures for 10 classes. Forty subjects, from various ethnic groups, execute the postures against a variety of diverse backgrounds. The classes in this dataset are named 1 to 10 and have letters 'a' to 'j' respectively. There are a total of 2000 images each of size 160x120 pixels. For NUSII dataset the coefficient of variance of 6.45% confirms that the dataset is well-balanced across all 10 classes. The range between smallest and largest class is 265 samples, about 20% of the smallest class (1040), which is moderate but not extreme.
- **NUS II with Human Noise** is also captured at the National University of Singapore (NUS) by Pisharady et al. [33] called as 'Hand Postures with human noise' dataset which contains 450 images of 10 classes and 15 subjects, of image size 320×240 pixels with human noises like body/face. The classes are similar to the NUS II dataset which is 1 to 10 represented as 'a HN' to 'j HN', but human noise has been added in the background to make it more complex. Every class has exactly the same number of training and testing samples. Hence there is no risk of class imbalance bias during training or evaluation. Models will treat each class equally in terms of representation.
- **Sebastien Marcel Static Hand Posture Database** is publically available and collected by Sebastien Marcel of France Telecom CNET, France.[34] There are six classes namely A, B, C, Five, Point, and V collected for 10 subjects. The images are in uniform light, dark, and complex backgrounds and of size 128×128 pixels in the .pnm format which is changed to .jpg, and augmentation has been done for fair evaluation. It has a severe class imbalance. Standard deviation is very high (396.63) relative to the mean (858.17). Some classes have nearly three times the number of samples as others (e.g., 1453 vs 489).
- **Triesch's Alphabet Dataset** is a publicly available dataset captured by Jochen Triesch of Frankfurt University, Germany [35]. Mean sample size is 434 with standard deviation of 2.11. Samples range from 430 to 435. There are single-handed hand posture images of 10 gestures of 24 subjects with different lighting conditions and complex backgrounds. It is balanced dataset with a CV of only 0.65%, the dataset is nearly uniform across all classes.
- **Triesch's Digit Dataset** is another dataset curated at Frankfurt University, Germany is the digit dataset [35]. The dataset contains 12 hand posture classes. Total images per class range from 390 to 440, with no outliers The average number of total images per class is 427, with a small standard deviation (13.05), meaning the dataset is well-balanced. The dataset has an average of 342.5 training samples per class, with a low variance (std = 5.0). Originally, the image format was .pgm which is converted to.jpg.



Figure 1: NUS II



Figure 2: NUS II with HN

Licensed Under Creative Commons Attribution CC BY

Figures 1 to 5 highlight the substantial variability across existing sign language datasets in terms of the number of signs, the diversity of subjects, and the number of samples per class. With these discrepancies, model performance can be affected, emphasizing the importance of creating datasets that are suited to the unique challenges in the sign language domain. Gaining insights from the literature, we found answers to the key research questions like the number of signs, the diversity in data to ensure signs performed between different age groups, genders, and background conditions. These are the main points that are crucial for developing models that can generalize well in real-time. This paper aims to bridge this gap in terms of datasets, so that recognition systems can be effectively used by the hearing-impaired population.

Table 2: Dataset Details and Best Accuracy on Deep Learning Model

Dataset	Classes	Subjects	Images	Best Accuracy (%)
NUS II[1]	10	40	11,720	97.78[36]
NUS II with HN[2]	10	15	4,500	—
SHP MARCEL[3]	6	10	5,149	92.72[28]
TRIESCH'S 10 ALPHABETS[4]	10	24	4,340	97.86[16]
TRIESCH'S DIGITS[5]	12	24	5,125	—

[1]https://www.ece.nus.edu.sg/stfpage/elepv/NUS-HandSet/
[2]https://www.ece.nus.edu.sg/stfpage/elepv/NUS-HandSet/
[3]https://www.idiap.ch/webarchives/sites/www.idiap.ch/resource/gestures/
[4]https://www.idiap.ch/webarchives/sites/www.idiap.ch/resource/gestures/
[5]https://www.idiap.ch/webarchives/sites/www.idiap.ch/resource/gestures/

4. Proposed Methodology

The proposed Vision Transformer (ViT) architectures for the recognition of static American Sign Language alphabets and numbers have been discussed in this section. The methodology includes several stages, and the training and evaluation pipeline is depicted in Figure 7.



Figure 3: SHP MARCEL



Figure 4: TRIESCH'S ALPHABETS

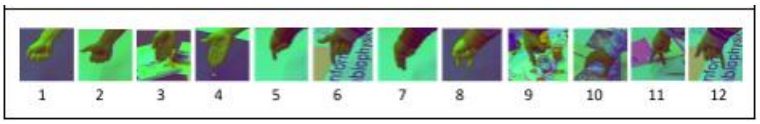


Figure 5: Triesch's Digits

4.1 Data Preprocessing

The first step is to annotate each dataset with labels corresponding to its sign to validate accurate supervised learning. Many preprocessing steps were followed to maximize the data for training with the proposed model techniques. Firstly, all the images are resized to a homogeneous scale (224x224 pixels), to match the model's input state. These pixel values are normalized according to ImageNet metrics. Data augmentation like rotation, flipping, scaling and cropping were executed to generalize the model capabilities and mitigate overfitting, particularly for datasets with imbalanced classes or background variability. Triesch's Digit dataset have highly cluttered background with low contrast, so Contrast enhancement has been done so that gestures are distinguishable for better recognition. Augmentations are applied after splitting datasets to avoid data leakage.

4.2 Vision Transformer Architecture

In the proposed work, we use ViT-Base/16 variant that serves as core feature extractor for American Gesture Recognition. The parameters of this model are described in Table 3. ViT comprises a self-attention mechanism that takes the image as sequence of patches, captures spatial information, suitable for cluttered and heterogeneous visual gestures. Suppose that we have a set of sign language images, denoted by:

$$I = \{(X_i, y_i)\}$$

Where $i = 1, 2, \ldots, n$. Here, X_i represents an image, y_i is its corresponding class label, with $y_i \in \{1, 2, \ldots, m\}$, and m is the total number of classes in the dataset. The Vision Transformer model aims to learn the mapping from a series of image patches to their corresponding semantic labels.

Licensed Under Creative Commons Attribution CC BY

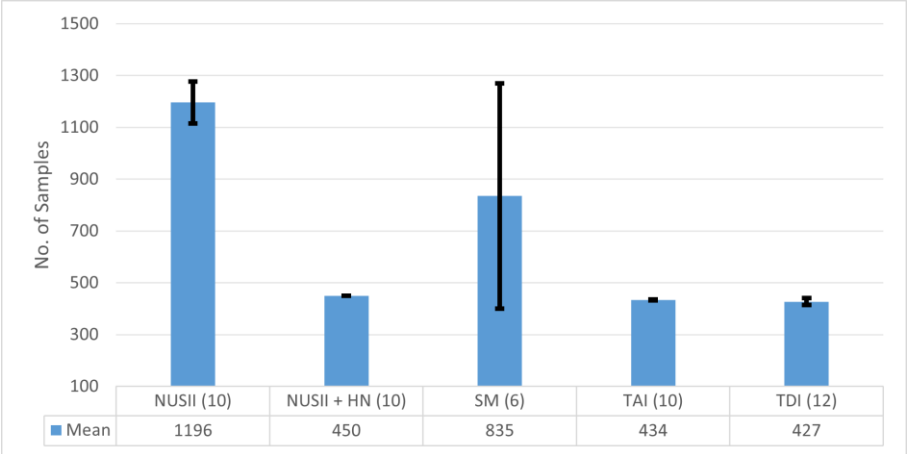


Figure 6: Dataset Analysis

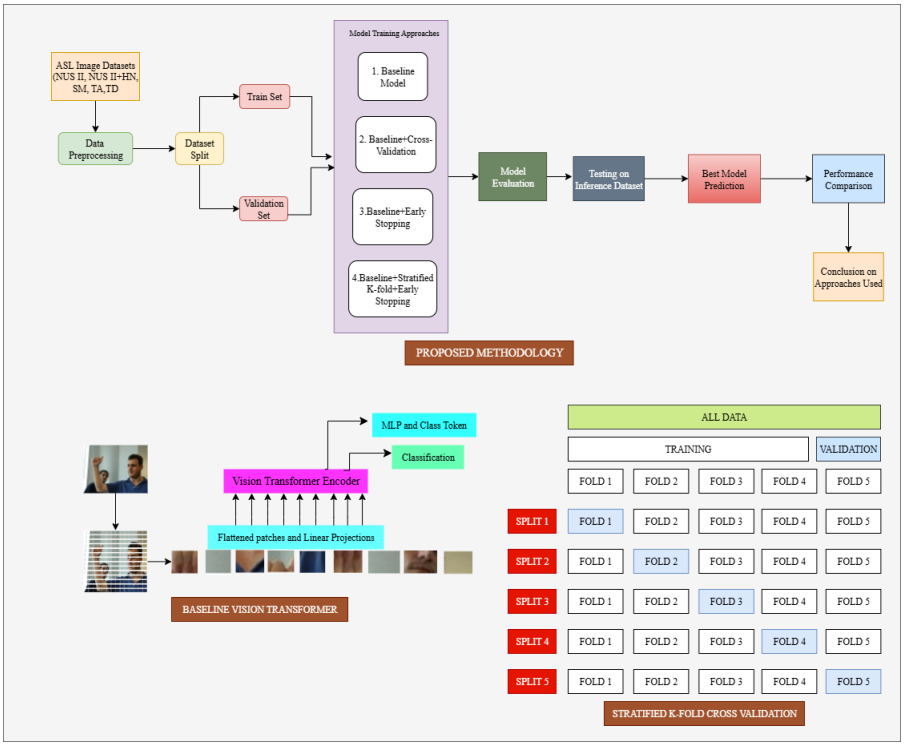


Figure 7: Proposed Training Flow

In this work, we have an image of size 224×224 which will be split into smaller 16×16 patches, creating a series of patch embeddings. To obtain patch embeddings, these flattened patches are passed through a layer of linear projection. Since transformers do not naturally comprehend spatial structure, position embeddings are introduced to preserve spatial information.

Given an image $I \in R^{h \times w \times c}$, which is a fancy matrix of colorful bits (pixels) where h is high and w is wide, with c different channels (like the RGB)

- h: Height of the image.
- w: Width of the image.
- c: Number of channels (e.g., 3 for RGB images).

A vision transformer takes an input image and chops it into smaller patches that are pixel-by-pixel. It is similar to cutting a pizza into small slices, but for pixels! After this, these patches are squished into 1D vectors, so that they are strings of pixels. Total number of patches received, i.e., N, is nothing but the height and width of the image divided by the size of the patch and then squared:

$$N = \frac{h \times w}{P^2} \quad (1)$$

Each of these patches gets turned into a vector of size $P^2 \times c$. Therefore, the set of patches I_p is represented as:

$$I_p \in \mathbb{R}^{N \times (P^2 \cdot c)} \quad (2)$$

Where $P^2 \cdot c$ is the number of pixels per patch multiplied by the number of channels.

It's like a pixel party in a vector format!

After that, each flattened patch is mapped onto a D-dimensional latent space, which is the embedding dimension of the transformer, using a trainable linear projection:

$$z_0^i = w_p \cdot I_p^i + b_p, \quad z_0^i \in \mathbb{R}^D \quad (3)$$

Licensed Under Creative Commons Attribution CC BY

Where:
- z_0^i is the i-th patch embedding.
- $w_p \in \mathbb{R}^{D\times(P^2\cdot c)}$ is the trainable projection matrix.
- b_p is the bias term.
- D is the embedding dimension used in the transformer.

Unlike CNNs which use convolutions, Transformers do not naturally possess knowledge about the spatial structure of the input. Ergo, position embeddings are added to the patch embeddings.

$$z_0 = [z_0^1; z_0^2; \ldots; z_0^N] + E_{\text{pos}} \quad (4)$$

Where:
- $E_{\text{pos}} \in \mathrm{R}^{N\times D}$ are learnable position embeddings.

The resulting input to the transformer is a sequence of patch embeddings with positional information. ViT adds a unique token to the series of patch embeddings that is referred to as the class token z_{class}. For the final classification task, this token is used to combine the data from all patches throughout training:

$$z_0 = [z_{\text{class}}; z_0^1; z_0^2; \ldots; z_0^N] \quad (5)$$

During the transformer layers, the class token is updated and utilized in the final classification phase.

4.3 Transformer Encoder

The model's central components are several transformer encoder layers, each comprising multi-head attention, feed-forward network, layer normalization, and skin connections. These layers receive the sequence of patch embeddings together with the class token. Multi-head attention allows the model to focus on different areas of the image at the same time, capturing the relationships between distinct patches. Then feed forward network processes the attention-weighted features using dense layers. To aid gradient flow and prevent vanishing gradients, outputs are normalized and skip connections are added. The following is how each transformer encoder layer handles the input:

(i) Multi-Head Self-Attention (MSA):
The model can focus on various areas of the image (patches) according to their connections with each other thanks to the self-attention mechanism. The query Q, key K, and value V matrices are calculated for every patch for every layer:

$$Q = z_l W_Q, \quad K = z_l W_K, \quad V = z_l W_V \quad (6)$$

Where:
- W_Q, W_K, W_V are the learned projection matrices for the queries, keys, and values.
- z_l is the input embedding for layer l.

The scaled dot-product attention is used to calculate the attention weights:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^\top}{\sqrt{D_k}}\right)V \quad (7)$$

where the keys' dimensionality is represented by D_k. Concatenating the results is the attention operation repeated for many heads, also known as multi-head attention.

(ii) Feed-Forward Network (FFN):
Each patch embedding has a feed-forward network applied to it after the self-attention layer. The FFN is made up of two linear layers connected by a ReLU activation.

$$\text{FFN}(z) = \max(0,\, zW_1 + b_1)W_2 + b_2 \quad (8)$$

Where W_1, W_2 and b_1, b_2 represent weights and biases of the feed-forward network.

Layer Normalization (LN) and residual connections make up each transformer layer, and as a result, each layer's ultimate output is:

$$z_{l+1} = \text{MSA}(\text{LN}(z_l)) + z_l \quad (9)$$

$$z_{l+1} = \text{FFN}(\text{LN}(z_{l+1})) + z_{l+1} \quad (10)$$

4.4 Classification Head

The series of patch embeddings is supplemented by a unique categorization token. The output embeddings will be input into a fully connected classification layer to predict the matching sign language gesture once they have passed through the transformer layers. A softmax classifier will receive the transformer's output to generate probability distributions over the sign labels.

After passing through the transformer layers, the final output of the class token Z_{class}^L (where L is the number of transformer layers) is used for classification:

$$y = \text{MLP}(z_{\text{class}}^L) \quad (11)$$

Where:
- MLP is a multi-layer perceptron applied to the class token.
- The output y represents the predicted class probabilities.

Loss Function:
For classification tasks, the standard cross-entropy loss is typically used to train the model:

$$L = -\sum_{i=1}^{K} y_i \log(\hat{y}_i) \quad (12)$$

Where:
- K is the number of classes.
- y_i is the ground truth label for class i.
- $\hat{y}_i$ is the predicted probability for class i.

4.5 Training Strategies

We investigated four different training strategies, to promote regularization and analyze evaluation:
1) **Baseline Training**: Standard splitting of dataset into train and validation sets without regularization.
2) **Baseline + Stratified K-Fold Cross-Validation**: Dataset is divided into 5 folds ensuring a class balance.
3) **Baseline + Early Stopping**: Validation loss is monitored in this strategy to avoid overfitting.
4) **Baseline + Stratified K-Fold Cross-Validation + Early Stopping**: Here both techniques are combined to achieve robustness.

We apply Stratified K-Fold cross-validation for addressing class imbalance in SHP MARCEL dataset, which maintains class ratio across each fold. The final model is chosen by

Licensed Under Creative Commons Attribution CC BY

averaging validation performance among these folds.

Table 3: Transformer parameters for the ViT-Base model

Model Parameters	Values
Model Name	ViT-Base (ViT-B/16)
Input Size	224 × 224
Patch Size	16 × 16
Number of Patches	196 (i.e., 224/16 = 14 × 14)
Hidden Size (1D vector)	768
MLP Hidden Dimension	3072
Number of Encoder Layers	12
Attention Heads	12
Attention Head Size	64 (768 / 12)
Total Parameters	85,803,270

5. Experiment and analysis

5.1 Stratified K-Fold Cross-Validation

To ensure that the model's performance was thoroughly and robustly evaluated, we employ stratified k-fold cross-validation. Every portion of data used for training and validation has been verified to be representative of the whole dataset since this approach maintains the original class distribution balance across each fold. The following steps are involved in the process:

- Dataset Split: Here, the data is divided into equivalent folds using the concept of stratification, which ensures the proportion of sign classes in the data set.
- Model Training: Model Training: In this step, the model is trained using the cross-validation technique, which has been validated for k iterations. This helps in avoiding overfitting and ensures the generalization of the model.
- Performance Evaluation: Various evaluation parameters are used to evaluate the performance of the model, which include the following: Accuracy, Confusion Matrix, Classification Reports, Precision, Recall, F1-Score, etc. In order to achieve the right approximation of the performance of the model, the parameters mentioned above are collectively used.

The model has been tested using public domain American Sign Language datasets, which are unique in their features and have varying complexities in the background. In order to prove the efficiency of the proposed Vision Transformer model, the results are compared with the existing models, which include Convolutional Neural Network models. The proposed ViT model is efficient in the classification of complex single-handed ASL gestures, in which the relationships between the gestures are very significant. The implementation of the model is done using the specifications as shown in Table 4 below.

5.2 Classification Results

The experiments are conducted using the Pytorch library. The authors of the original paper "Image is worth 16 x 16" presented several variations of the Vision Transformer (ViT) model. To find out how architectural decisions affected the model's performance, they experimented with various model configurations. The vision transformer model has three variations. ViT-Base, ViT-Large, ViT-Huge. We used pretrained vision transformer (ViT-Base/16) architecture. The hyperparameters of this base model are: image size: 384×384, patch size: 16×16, learning rate: 0.001, weight decay: 0.0001, batch size: 16, hidden size: 768, mlp size: 3072, number of heads: 12, encoder length:12, number of parameters are 86 Million. We also used cross entropy loss and Adam optimizer. Before being fed into the suggested model, the gesture photos from each class are mixed and split into training and testing datasets with a ratio of 80:20, respectively. The total number of epochs used are 50 without cross-validation and 5-folds with 10 epochs per fold with cross-validation. Early stopping has also been done to get best results. Augmentation has been done for some of the datasets used for generalization and to avoid over-fitting. The datasets utilized in this paper have number of signs of alphabets and numbers of American Sign Language (ASL) with one-handed having complex backgrounds.

Table 4: Experimental setup

Hardware/Software	Specifications
Processor (CPU)	Intel Core i5-10400F @ 2.90GHz
Memory (RAM)	32.04 GB
Operating System	Windows 11 Home Edition
GPU	NVIDIA Quadro P5000
Python Version	3.10.14 with PyTorch
CUDA / cuDNN Version	CUDA 12.2.2

Table 5: Comparison of training and validation accuracies using different strategies

Strategy	Dataset	Train Accuracy (%)	Validation Accuracy (%)
Baseline	NUS II	99.99	95.23
	NUS II + Human Noise	91.37	92.84
	SHP MARCEL	99.45	40.46
	TRIESCH's Alphabets	98.92	95.57
	TRIESCH's Numbers	93.83	71.25
	NUS II	97.02	94.15
	NUS II + Human Noise	94.96	87.07
Baseline + Cross-Validation	SHP MARCEL	99.41	98.59
	TRIESCH's Alphabets	98.47	94.53
	TRIESCH's Numbers	93.54	82.43
	NUS II	99.71	97.37
	NUS II + Human Noise	99.95	94.67
Baseline + Early Stopping	SHP MARCEL	99.30	62.82
	TRIESCH's Alphabets	100.00	95.73
	TRIESCH's Numbers	95.52	72.22
	NUS II	99.70	98.34

Licensed Under Creative Commons Attribution CC BY

	NUS II + Human Noise	99.90	99.33
Baseline + Stratified k-Fold + Early Stop	SHP MARCEL	100.00	99.69
	TRIESCH's Alphabets	100.00	100.00
	TRIESCH's Numbers	99.48	97.93

5.3 Analysis of Training Strategies on Training Dataset

Evaluation of training and validation accuracies highlights how performance strategies affect the model's ability to generalize well on unseen data.

- Models trained in the absence of cross-validation achieved consistently high accuracies in most datasets, usually around 99%. However, it failed to generalize, as noted in the case of validation accuracy. The SHP MARCEL dataset suffered 59% loss with validation accuracy of 40.46%. Further, TRIESCH's Digits dataset recorded a validation drop of over 22%, indicating overfitting, due to datasets having high variability or class imbalance.
- By adopting 5-fold cross-validation learning curve stabilized significantly, showing realistic training metrics and boosting validation scores. TRIESCH'S DIGITS dataset improved from 71.25% to 82.43%, while the validation precision of SHP MARCEL jumped to 98. 59%. Thus, the above improvements underscore the role of cross-validation across these datasets, reducing overfitting.
- Implementing early stopping as a regularization technique improved validation efficacy while stabilizing near-perfect training accuracies of 100%. For example, NUS II and TRIESCH's Alphabets gains validation accuracies of 97.37% and 95.73% respectively. However, the accuracy of the validation set remained low at 62.82% for more complex datasets such as SHP MARCEL. This indicates that the application of early stopping alone is not sufficient in managing the variability and noise in the datasets.
- Overall, the application of stratified k-fold cross-validation in combination with early stopping has proven to be a powerful protection against the inconsistencies experienced in the earlier tests. This is because all the data in the folds would be a true reflection of the data in the dataset. Moreover, the application of the two would prevent the model from over-learning the noise in the datasets. As can be seen from the results, the application of the Alphabets by TRIESCH has proven to be effective in the sense that the model obtained a perfect score in all the datasets. In the more complex NUS II dataset, the model obtained a 99.33% accuracy in the validation set. Even more impressive is the fact that the SHP MARCEL dataset improved significantly from being more likely to overfit to having a near-perfect 99.69% accuracy in the validation set.

Ultimately, this hybrid strategy proved to be the missing piece. It gives the model the backbone it needs to handle noise and complex hand shapes without folding under pressure. Because it's so robust, it's particularly well-suited for the subtle, fast-paced nature of single-handed American Sign Language in everyday settings.

5.4 Analysis of Trained Models on Inference Dataset

We weren't just looking for something that would work in a lab environment; we wanted to see how well they would work in a person's living room or on a street corner. To test this, we created an inference dataset that mimicked the uncontrolled environment of the real world- unpredictable lighting, backgrounds, and the way people move their hands in unique ways. By moving away from the 'perfect' environment of the training data, we were able to really see how well the models worked and which ones were simply memorizing patterns. This gave us a much clearer picture of the things that actually matter in a real-world application- how fast the model responds and how well it works when the image quality is subpar.

Table 6: Real-time testing on Inference Dataset

Strategy	Dataset	Accuracy	Time (sec)
Baseline	NUS II	47.35% (555/1172)	0.0078
	NUS II + HN	64.66% (291/450)	0.0061
	SHP MARCEL	75.92% (391/515)	0.0058
	TRIESCH's Alphabets	50.68% (220/434)	0.0063
	TRIESCH's Digits	38.86% (199/512)	0.0084
Baseline + Cross-Validation	NUS II	9.87% (116/1172)	0.0067
	NUS II + HN	10.88% (49/450)	0.0060
	SHP MARCEL	15.14% (78/515)	0.0059
	TRIESCH's Alphabets	8.52% (37/434)	0.0080
	TRIESCH's Digits	6.48% (33/512)	0.0058
Baline + Early Stop	NUS II	46.67% (547/1172)	0.0058
	NUS II + HN	67.77% (305/450)	0.0057
	SHP MARCEL	74.56% (384/515)	0.0058
	TRIESCH's Alphabets	49.30% (214/434)	0.0060
	TRIESCH's Digits	43.55% (223/512)	0.0087
Baseline + Stratified K-Fold + Early Stop	NUS II	44.11% (517/1172)	0.0059
	NUS II + HN	93.55% (421/450)	0.0060
	SHP MARCEL	77.47% (399/515)	0.0058
	TRIESCH's Alphabets	52.76% (229/434)	0.0059
	TRIESCH's Digits	14.06% (72/512)	0.0062

Licensed Under Creative Commons Attribution CC BY

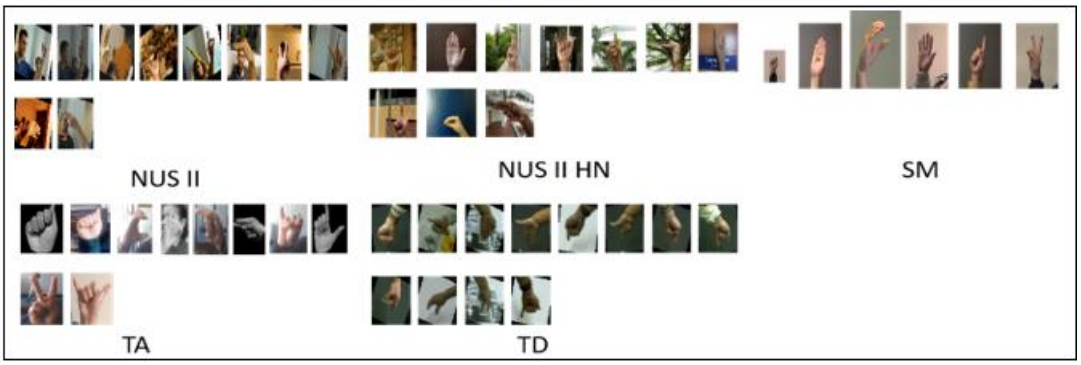
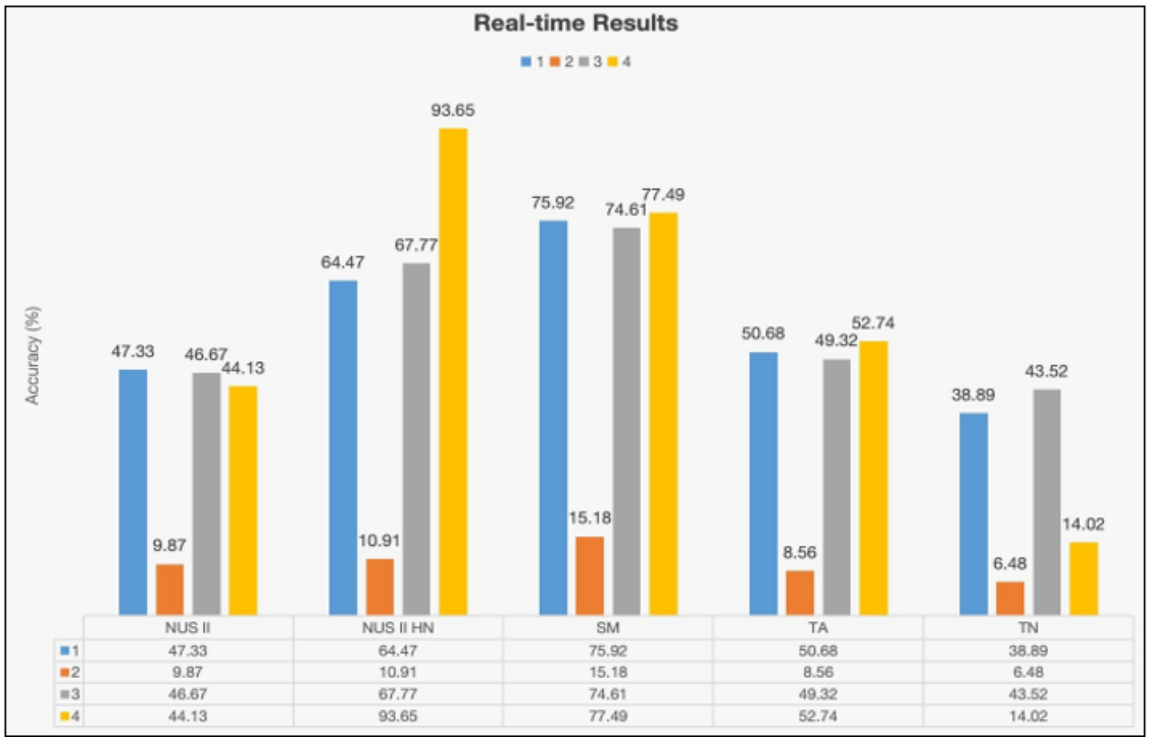


Fig. 8: Sample Images of Inference Dataset

- **Without Cross-Validation (10 Epochs):** Our basic approach of using a 10-epoch process provided us excellent speeds in inference (approximately 0.007 sec), but the accuracy was not consistent. Although SHP MARCEL performed well in this case, the high precision gaps in the TRIESCH datasets (going down to almost 39.89%) showed that our model was overfitting. It is fast enough for practical applications, but it is not wise enough to be able to generalize across different datasets without a better validation process.
- **Stratified K-Fold Cross-Validation (5 Folds):** Despite the positive indications in the training phase, the model was not able to generalize well, and the accuracy dipped to a dismal 15% or less. This just goes to show that despite the high scores in the validation phase, it may not always be the case that the scores are genuine, owing to the possibility of some sort of data leakage or improper handling of the fine-tuning phase after the validation phase. Despite the inference time still being 0.007 seconds, a fast model is of no use if it cannot even recognize the signs.



	NUS II	NUS II HN	SM	TA	TN
1	47.33	64.47	75.92	50.68	38.89
2	9.87	10.91	15.18	8.56	6.48
3	46.67	67.77	74.61	49.32	43.52
4	44.13	93.65	77.49	52.74	14.02

Figure 9: Proposed Model Training Flow

- **Early Stopping:** By employing early stopping, we were able to achieve our original training scores or even surpass them, all without allowing our model to over-train on the noise. There was a significant increase in our digit classification score of 43.52%, and consistent performance in SHP MARCEL. What was best, though, was that this technique gave us our lowest inference times so far, ranging from 0.006 to 0.008 seconds. It goes to show that it is a smart move to stop training at the right time.
- **Stratified K-Fold Cross-Validation with Early Stop:** Our last hybrid approach was also highly effective for our complex, noisy data sets; however, we do not claim that our approach was a magic bullet. We observed our approach performing at a high rate of 93.65% on NUS II + HN but struggling significantly with TRIESCH's Digits at only 14.02%. We attribute these mixed results to potential problems with our approach to fold stratification or our model's inability to recognize the numeric classes we are struggling with. Nonetheless, our model's speed in making predictions at 0.006 sec validates our approach in the right direction.

Our evaluation highlights a significant gap between 'clean'

Licensed Under Creative Commons Attribution CC BY

and 'noise-augmented' data. The model peaked at NUS II + Human Noise with a weighted F1 score of 0.9353. This implies that the variability we added to the model during training was the key to its robustness. Although SHP MARCEL showed good performance at 0.7706, the original NUS II dataset was not performing as well at 0.4615. It was probably a victim of its own lack of diversity. The TRIESCH results were more telling, however. The low F1 score at 0.4047 implies serious problems like class imbalance or the presence of similar signs that are just too similar for the model to tell apart. This brings us to a very important point: creating a successful real-time SLR system is not just about the architecture; it's also about the quality and 'realness' of the data you put into it.

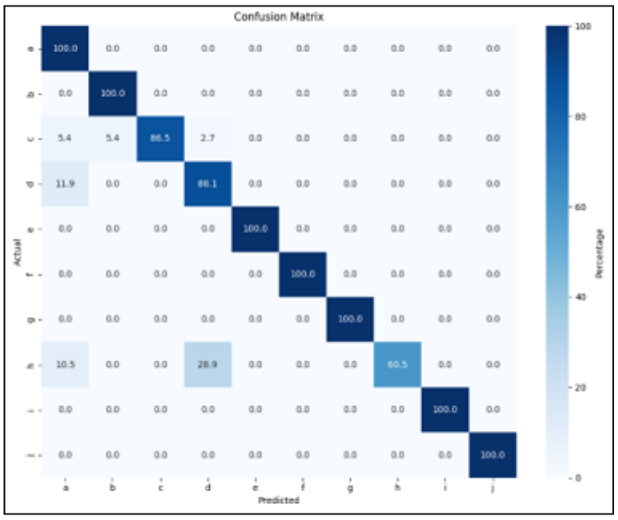


Figure 10: NUS II HN

Table 7: Evaluation metrics for real-time testing

Dataset	Precision (%)	Recall (%)	F1 Score (%)
NUS II	50.59	46.67	46.15
NUS II HN	94.88	93.65	93.53
SM	82.03	76.96	77.03
TA	62.87	52.74	50.53
TD	65.11	43.52	40.57

6. Conclusion

In the course of this research, the application of a hybrid approach in training models, specifically Stratified K-Fold Cross-Validation and Early Stopping has been explored with the explicit aim of surmounting the barriers in static American Sign Language recognition. As the following experiments using various benchmarking datasets reveal, this is not only a theoretical solution but a practically applied solution as well. By employing Stratified K-Fold Cross-Validation and Early Stopping, we have been able to not only stabilize the reliability of the validation but more importantly, put a significant dent in the problems of overfitting that tend to plague such complex models. The most convincing proof of this was demonstrated in our results with the NUS II+ Human Noise dataset. The fact that we were able to attain an inference accuracy of 93.65% in such an unorganized setting was a major milestone. It shows us that by presenting the model with "unorganized" data in the training phase, we can actually develop a much more robust system. This was also evident in our results with other datasets such as SHP MARCEL (77.49%) and TRIESCH's Alphabets (52.74%), even in settings where there is not much variation in the backgrounds. By ensuring class balancing in each fold and avoiding over-learning by implementing early stopping, we've managed to develop a system that generalizes incredibly. In addition to this accuracy, however, is the efficiency of this system. With a constant time of 0.006 seconds per input, this model is perfectly suited for real-time work. It is quick enough to be imperceptible to the end user and robust enough to account for the unpredictable nature of real-life lighting, hand positioning, and individual signer characteristics. Furthermore, this technique using human-made noise and diverse input sets brings us one step closer to a model that can work in a living room environment, rather than a laboratory setting. In terms of where this can go next, it is rather obvious that there is the possibility of further innovation. Maybe this can be done in the form of using even larger input sets in order to further hone this level of recognition. These results not only measure the performance but also give new life to the input sets that had been previously ignored.

Table 8: Performance comparison across datasets

Dataset	Literature	Train Acc.	Val. Acc.	Inference Acc.	Precision	Recall	F1 Score
NUS II	97.78 [36]	99.70	98.34	44.13	0.5059	0.4667	0.4615
NUS II with HN	93.07 [33]	99.90	99.33	93.65	0.9488	0.9365	0.9353
SHP MARCEL	92.72 [28]	100.00	99.69	77.49	0.8203	0.7696	0.7703
TRIESCH's Alphabets	97.86 [16]	100.00	100.00	52.74	0.6287	0.5274	0.5053
TRIESCH's Digits	92.08 [15]	99.48	97.93	14.02	0.6511	0.4352	0.4057

References

[1] World health organization. deafness and hearing loss (2024)
[2] Priya, K., Sandesh, B.: Developing an offline and real-time indian sign language recognition system with machine learning and deep learning. SN Computer Science **5**(3), 273 (2024) https://doi.org/10.1007/s42979-023-02482-w
[3] Mahyoub, M., Natalia, F., Sudirman, S., Mustafina, J.: Sign language recognition using deep learning. In:

Licensed Under Creative Commons Attribution CC BY

2023 15th International Conference on Developments in eSystems Engineering (DeSE), pp. 184–189 (2023). https://doi.org/10.1109/ DeSE58274.2023.10100055 . IEEE

[4] Ferhat, R., Chelali, F.Z.: Textural feature descriptors for a static and dynamic hand gesture recognition system. Multimedia Tools and Applications **83**(3), 8165–8187 (2024) https://doi.org/10.1007/s11042-023-15410-0

[5] Singla, V., Bawa, S., Singh, J.: Improving accuracy using ml/dl in vision-based techniques of islr. Multimedia Tools and Applications **83**(7), 20677–20698 (2024) https://doi.org/10.1007/s11042-023-16299-5

[6] Bhaumik, G., Verma, M., Govil, M.C., Vipparthi, S.K.: Hyfinet: Hybrid feature attention network for hand gesture recognition. Multimedia Tools and Applications **82**(4), 4863–4882 (2023) https://doi.org/10.1007/ s11042-021-11623-3

[7] Karsh, B., Laskar, R.H., Karsh, R.K.: miv3net: modified inception v3 network for hand gesture recognition. Multimedia Tools and Applications **83**(4), 10587–10613 (2024) https://doi.org/10.1007/s11042-023-15865-1

[8] Attia, N.F., Ahmed, M.T.F.S., Alshewimy, M.A.: Efficient deep learning models based on tension techniques for sign language recognition. Intelligent Systems with Applications **20**, 200284 (2023) https://doi.org/10.1016/j.iswa.2023.200284

[9] Bhatt, V., Dash, R.: Dynamic hand gesture recognition via reinforcement learning with adaptive category exclusion. The Visual Computer **41**, 11349–11372 (2025) https://doi.org/10.1007/s00371-025-04110-5

[10] Keskin, E., O¨ zcan, O., Yemez, Y.: Two-hand on-skin gesture recognition: a dataset and classification network for enhanced human–computer interaction. The Visual Computer **41**, 11641–11656 (2025) https://doi.org/10.1007/ s00371-025-04125-y

[11] Sadeghzadeh, A., Shah, A.S., Islam, M.B.: Mlmsign: Multi-lingual multi-modal illumination-invariant sign language recognition. Intelligent Systems with Applications **22**, 200384 (2024) https://doi.org/10.1016/j.iswa.2024.200384

[12] Reddy Karna, S.N., Kode, J.S., Nadipalli, S., Yadav, S.: American sign language static gesture recognition using deep learning and computer vision. In: 2021 2nd International Conference on Smart Electronics and Communication (ICOSEC), pp. 1432–1437 (2021). https://doi.org/10.1109/ICOSEC51865.2021.9591845 . https://doi.org/10.1109/ICOSEC51865.2021.9591845

[13] Alharthi, N.M., Alzahrani, S.M.: Vision transformers and transfer learning approaches for arabic sign language recognition. Applied Sciences **13**(21), 11625 (2023) https://doi.org/10.3390/app132111625

[14] Adithya, V., Rajesh, R.: A deep convolutional neural network approach for static hand gesture recognition. Procedia Computer Science **171**, 2353–2361 (2020) https://doi.org/10.1016/j.procs.2020.04.255

[15] Joshi, G., Singh, S., Vig, R.: Taguchi-topsis based hog parameter selection for complex background sign language recognition. Journal of Visual Communication and Image Representation **71**, 102834 (2020) https://doi.org/10.1016/j.jvcir.2020.102834

[16] Palanisamy, G., Thangaswamy, S.S.: An efficient hand gesture recognition based on optimal deep embedded hybrid convolutional neural network-long short term memory network model. Concurrency and Computation: Practice and Experience **34**(21), 7109 (2022) https://doi.org/10.1002/cpe.7109

[17] Kasapba¸si, A., Elbushra, A.E.A., Omar, A.-H., Yilmaz, A.: Deepaslr: A cnn based human computer interface for american sign language recognition for hearing-impaired individuals. Computer Methods and Programs in Biomedicine Update **2**, 100048 (2022) https://doi.org/10.1016/j.cmpbup.2021.100048

[18] Nihal, R.A., Rahman, S., Broti, N.M., Deowan, S.A.: Bangla sign alphabet recognition with zero-shot and transfer learning. Pattern Recognition Letters **150**, 84–93 (2021) https://doi.org/10.1016/j.patrec.2021.06.020

[19] Martinez-Martin, E., Morillas-Espejo, F.: Deep learning techniques for spanish sign language interpretation. Computational Intelligence and Neuroscience **2021**(1), 5532580 (2021) https://doi.org/10.1155/2021/5532580

[20] Liang, H., Fei, L., Zhao, S., Wen, J., Teng, S., Xu, Y.: Mask-guided multiscale feature aggregation network for hand gesture recognition. Pattern Recognition **145**, 109901 (2024) https://doi.org/10.1016/j.patcog.2023.109901

[21] Rajalakshmi, E., Elakkiya, R., Prikhodko, A.L., Grif, M.G., Bakaev, M.A., Saini, J.R., Kotecha, K., Subramaniyaswamy, V.: Static and dynamic isolated indian and russian sign language recognition with spatial and temporal feature detection using hybrid neural network. ACM Transactions on Asian and Low-Resource Language Information Processing **22**(1), 1–23 (2022) https://doi.org/10.1145/3530989

[22] Gao, Y., Zhang, Y., Jiang, X.: An optimized convolutional neural network with combination blocks for chinese sign language identification. CMES-Computer Modeling in Engineering & Sciences **132**(1) (2022) https://doi.org/10.32604/ cmes.2022.019970

[23] Podder, K.K., Chowdhury, M.E., Tahir, A.M., Mahbub, Z.B., Khandakar, A., Hossain, M.S., Kadir, M.A.: Bangla sign language (bdsl) alphabets and numerals classification using a deep learning model. Sensors **22**(2), 574 (2022) https://doi. org/10.3390/s22020574

[24] Ma, Y., Xu, T., Han, S., Kim, K.: Ensemble learning of multiple deep cnns using accuracy-based weighted voting for asl recognition. Applied Sciences **12**(22), 11766 (2022) https://doi.org/10.3390/app122211766

[25] 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) https://doi.org/10.1016/j.eswa.2022.118914

[26] Nareshkumar, M.D., Jaison, B.: A light-weight deep learning-based architecture for sign language classification. Intelligent Automation & Soft Computing **35**(3) (2023) https://doi.org/10.32604/iasc.2023.027848

Licensed Under Creative Commons Attribution CC BY

[27] Nandi, U., Ghorai, A., Singh, M.M., Changdar, C., Bhakta, S., Kumar Pal, R.: Indian sign language alphabet recognition system using cnn with diffgrad optimizer and stochastic pooling. Multimedia Tools and Applications **82**(7), 9627–9648 (2023) https://doi.org/10.1007/s11042-021-11595-4
[28] 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) https://doi.org/10.1016/j.birob.2023.100141
[29] Awaluddin, B.-A., Chao, C.-T., Chiou, J.-S.: A hybrid image augmentation technique for user-and environment-independent hand gesture recognition based on deep learning. Mathematics **12**(9), 1393 (2024) https://doi.org/10.3390/ math12091393
[30] Alsolai, H., Alsolai, L., Al-Wesabi, F.N., Othman, M., Rizwanullah, M., Abdelmageed, A.A.: Automated sign language detection and classification using reptile search algorithm with hybrid deep learning. Heliyon **10**(1) (2024) https: //doi.org/10.1016/j.heliyon.2023.e23252
[31] Kolikipogu, R., Mammai, S., Nisha, K., Krishna, T.S., Kuchipudi, R., Sureddi, R.K.: Indian sign language recognition for hearing impaired: A deep learning based approach. In: 2024 3rd International Conference for Innovation in Technology (INOCON), pp. 1–7 (2024). https://doi.org/10.1109/INOCON60754. 2024.10511911 . IEEE. https://doi.org/10.1109/INOCON60754.2024.10511911
[32] Altaher, A.S., *et al.*: Mamba vision models: Automated american sign language recognition. Franklin Open **10**, 100224 (2025) https://doi.org/10.1016/j.fropen.2025.100224
[33] Pisharady, P.K., Vadakkepat, P., Loh, A.P.: Attention based detection and recognition of hand postures against complex backgrounds. International Journal of Computer Vision **101**, 403–419 (2013) https://doi.org/10.1007/ s11263-012-0560-5
[34] Marcel, S.: Hand posture recognition in a body-face centered space. In: CHI'99 Extended Abstracts on Human Factors in Computing Systems, pp. 302–303 (1999). https://doi.org/10.1145/632716.632901.
[35] https://doi.org/10.1145/632716.632901
[36] Triesch, J., Von Der Malsburg, C.: Robust classification of hand postures against complex backgrounds. In: Proceedings of the Second International Conference on Automatic Face and Gesture Recognition, pp. 170–175 (1996). https://doi.org/ 10.1109/AFGR.1996.557260 . IEEE. https://doi.org/10.1109/AFGR.1996.557260
[37] Ma, Y., Xu, T., Kim, K.: Two-stream mixed convolutional neural network for american sign language recognition. Sensors **22**(16), 5959 (2022) https://doi.org/ 10.3390/s22165959

Licensed Under Creative Commons Attribution CC BY