International Journal of Advanced Engineering Application (IJAEA)
Volume 3 Issue 11 Nov 2026
ISSN :3048-6807

# Multilingual Sign Language Translation using Computer Vision and Generative AI

**Chichula Abhishek[1],Kotilingala Jeevitha[2],Attada Anvek[3], Gudla Sai Lahari[4], Dr Ajit Kumar Rout[5]**
[1,2,3,4]Department of Information Technology, GMRIT Deemed to be University, Rajam, Andhra Pradesh, India
*[5]Professor, Department of Information Technology, GMRIT Deemed to be University, Rajam, Andhra Pradesh, India*
Email: abhishekchichula3@gmail.com.

**Abstract:**
***There is a great difficulty in communicating between Indian sign language users and nonusers. The paper introduces a multilingual sign language translation framework, which is based on computer vision, deep learning, and Generative AI to facilitate communication. The sign recognition module is responsible for capturing data from the video using OpenCV, detecting the hands in the image using MediaPipe Hands, extracting the spatial features using EfficientNet-B0 and learning the temporal features using Bidirectional Gated Recurrent Unit network. For the recognition model, the training and evaluation set consisted of INCLUDE which contains a total of 262 ISL sign classes. 32 frames were extracted for each input video sequence and 1,280 dimensional feature vectors were used. The performance obtained from the experimental evaluation for 619 samples was found to be 86.75% with 537 correct predictions. The framework also includes a user-controlled gloss buffer where in known sign labels are gathered and sent for sentence generation when the user has manually triggered translation. After the user manually triggers translation, Generative AI is used to convert the buffered sign labels into grammatically meaningful sentences. The generated sentences are then translated into English, Telugu, Hindi, and Tamil using the multilingual translation module and converted into speech using the Text-to-Speech module.***
***The following components for additional sentence generation, multilingual translation and speech are part of the framework, but have not been quantitatively assessed in this study. The results confirm the viability of the proposed architecture for the recognition of isolated signs using the ISL, and serve as a basis for developing the system towards multilingual sign language communication.***

**Keywords:** ***Indian Sign Language, Isolated Sign Recognition, Computer Vision, EfficientNet-B0, Bidirectional GRU, Generative AI, Multilingual Translation, Text-to-Speech.***

## 1. INTRODUCTION

Communication is a vital life skill which enables people to exchange information, ideas and feelings. However, people with hearing and speech impairments or deaf people who don't know sign language may have communication challenges if other people don't know sign language. These barriers can impact their daily lives, learning, health care, work and social relationships. While there are a number of systems that have been developed to support a sign language system, many of these systems have limited support for continuous sign recognition, multilingual translation, natural sentence generation, and speech output. The recent progress in Computer Vision, Deep Learning and Generative Artificial Intelligence (AI)

offers the promise of creating intelligent systems that enhance communication and access. The difficulty of sign language translation lies in the fact that hand shapes, positions and movements evolve all the time. Understanding individual gestures isn't enough to make out an entire sentence or conversation. Furthermore, there are speed differences between signs, background noise, and individual signers which can impact recognition. Deep Learning and Generative AI has been also explored for the improvement in sign language recognition and translation in the recent years by Geetha et al. and Wang et al. respectively for continuous Indian Sign Language recognition using RGB video and pose information. Kumar et al. studied the use of Large Language Models (LLMs) for sign language translation, demonstrating the potential of language models for creating meaningful text from recognized signs. Other research has explored multimodal feature extraction and deep learning methods for enhancing the recognition performance. The above studies illustrate the need to fuse two or three of the following: visual feature extraction, temporal sequence learning, and natural language processing (NLP) to achieve more efficient sign language translation systems. The project titled “Multilingual Sign Language Translation Using Computer Vision and Generative AI” proposes to create a

system that translates sign language gestures in real-time from a webcam into meaningful text using computer vision and Generative AI. It leverages OpenCV and MediaPipe Hands for video processing and hand landmark tracking, EfficientNet-B0 for extracting spatial features from the images, and Gated Recurrent unit (GRU) for learning temporal patterns within consecutive frames. The recognized sign labels are processed using Generative AI to generate meaningful sentences. The sentences generated are then translated to English, Telugu, Hindi and Tamil and speech synthesized using a Text-to-Speech (TTS) module. The proposed system will integrate these elements within a web-based application in order to minimize the problems of communication and accessibility. This research study's key Goals are: Develop real-time sign language recognition system using Computer Vision and Deep Learning techniques. To utilize MediaPipe Hands, EfficientNet-B0, and GRU for hand tracking, spatial feature extraction and temporal sequence recognition.

To use Generative AI to translate known sign labels into meaningful and grammatically correct sentences.

To translate the sentences that are generated into many languages such as English, Telugu, Hindi, Tamil etc. To translate and read out words (Text-to-Speech (TTS) module). To create a user-friendly web app that incorporates sign recognition, sentence generation, multilingual translation, speech output, and conversation history.

## 2. LITERATURE REVIEW

There are lots of research done on sign language recognition and translation using Computer Vision, Machine Learning and Deep Learning techniques. Thanks to the development of artificial intelligence, researchers have created systems that can identify hand gestures, decipher continuous sign language sequences and convert sign language into speech. In recent years, studies have also been conducted on the use of Transformer models for the improvement of recognition and sentence generation, as well as multimodal feature extraction and the use of Large Language Models (LLMs). However, problems like small data set, different signing style, continuous sign recognition and multilingual speech output still exist. To give a general overview of the current research on sign language recognition and translation, the following studies are examined.

[1] Geetha et al suggested the multimodal approach for continuous real-time Indian Sign Language (ISL) recognition using pose and RGB video information. They have created their Sign Flow model by integrating 3D Convolutional Neural Network (CNN), Transformer encoder, and Connectionist Temporal Classification (CTC) for sequence decoding. Continuous sign recognition in various signing speeds and environments was studied. The Word Error Rate (WER) in the reported results was 19 and the recognition accuracy was ~95% in the best-case scenario. But, quick signing and variations of the signer, background noise and sign beyond the training vocabulary still pose challenges. This highlights the need for effective models that can support reliable real-time sign recognition and accurately identify signs across consecutive video frames.

[2] Wang et al proposed a multi-scale spatial-temporal feature enhancement method. They have integrated the spatial feature extraction, multi-temporal perception, Bidirectional Long Short-Term Memory (BiLSTM), and CTC-based decoding into their STNet model. The model was tested with the datasets, including PHOENIX14, PHOENIX14-T, and CSL-Daily. The study showed that it improved over the existing methods for the datasets assessed. Some limitations remain, however, such as computational cost, data imbalance, hand occlusion and fast motions which make recognition more difficult. These constraints stimulate the creation of effective models that are able to recognize sign sequences in real world settings. This study highlights the importance of developing sign-language translation approaches that can address differences between sign languages and support effective communication.

[3] Kumar et al explored deep learning and Large Language Models. They utilize a Random Forest Classifier and CNN for sign recognition, an LLM for sign language video correction, and a RIFE-Net for smooth sign-language video generation. The results reported here were 93% gesture recognition accuracy and 94.2% text correction accuracy. The video output was also at 60 frames per second. The approach is, however, primarily static (only alphabet level gestures are supported) with some limited support for dynamic signs and for generating native word level ISL glosses (only for complete sentences). This means the need for systems that are able to interpret continuous sign sequences and produce meaningful sentences.

[4] Shams et al modified the multimodal ensemble approach to recognize Bangla Sign Language using spatial, skeletal and edge features. They propose a method based on CNN models to extract various kinds of visual information and fuse the results of the models to enhance recognition. The reported test accuracy was around 97.65% and the dataset for

the study consisted of 11,061 training images representing 38 signs. The method shows how it is possible to use a collection of visual features to achieve gesture recognition. The system, however, is mainly targeted at image-based signs, rather than sentence-to-sentence translation. Multiple feature-processing methods can result in higher computational complexity, as well.

[5] Novopoltsev et al proposed a foundation model for Automatic Sign Language Recognition (ASL). They employ a Vision Transformer-Large (ViT-L) backbone and self-supervised learning to learn spatial-temporal representations from sign-language videos with their approach of Video Masked Autoencoders (VideoMAE). It was tested on several datasets, such as Slovo, WLASL, GSL, and AUTSL. For the Slovo data set, the study showed state-of-the-art performance, but it showed competitive results on other data sets that were evaluated. But, the existence of large annotated data sets is still limited, especially for low resource sign languages. The need to capture detailed linguistic information in various sign languages is also difficult. This has made them aware of the need for the use of systems that are appropriate for regional sign languages, like ISL.

In [6] Said et al proposed an adaptive framework for continuous sign language recognition and translation based on Transformer. The model of Adaptive Transformer (ADTR) is designed to handle sign-language videos, incorporating adaptive masking, local clip self-attention, and adaptive feature fusion. The study adopted the data set of ArabSign consisting of 9,335 samples from six signers, representing 50 sentences and 95 signs. For gloss-free translation, the reported BLEU-4 was 22.73 and for gloss based translation, the BLEU-4 score was 24.93 and ROUGE was 50.89. This work shows that the Transformer-based methods are capable of continuous sign translation. But the need to adapt these to different sign languages and incorporate them into a comprehensive multilingual system is important.

[7] Liu et al proposed a method to sign language translation from end-to-end based on an Adaptive Video Representation Enhanced Transformer. Their strategy is to enhance the representation of the video information prior to converting the sign-language sequences to spoken-language text. Models of a sequence of parts and relations can be represented using a transformer architecture, which enables the production of more meaningful translations. The quality and diversity of training data and the flexibility of the signatures to be handled by the system are, however, the keys to the success of the system. Additional research is required to ensure the creation of efficient and easily usable sign translation tools for real world applications.

[8] Najib had reviewed the techniques of sign language interpretation using Machine Learning and Artificial Intelligence. In this study, they studied how to automatically recognize signs and turn them into text or speech. These systems can be useful to lower barriers to communication between sign language users and non-sign language users. There are still some challenges with limited vocabulary and continuous sign sequences that are difficult to recognize. This means the development of systems that integrate all the function modules of gesture recognition, sentence generation, multi-language translation and speech output.

[9] Shen et al proposed Diverse Sign Language Translation to generate diverse and accurate SGT of continuous sign-language videos. They employ multi-reference training and reinforcement learning for improving translation generation. The framework was applied to the datasets PHOENIX14T and CSL-Daily, and the evaluation results are reported for the metrics of BLEU, ROUGE, BLEURT and pairwise BLEU in terms of translation quality and diversity. The study emphasizes the need to create the suitable translations instead of a set sentence. However, there are still some challenges: the size of the available datasets is still limited, and the evaluation of both the quality and diversity of the translation has still been a challenge.

[10] Talaat et al introduced a real-time Arabic sign-language translation system for deaf and hearing people, based on deep learning. Their method is based on YOLOv8 object detection model and data enhancement to enhance the performance of gesture recognition in various situations. The study demonstrates the performance of up to 99.4% precision, 99.5% recall and 99.5% mean Average Precision (mAP) on Arabic sign-language data sets such as AASL, SLDI and ArSL. However, the system can be further improved by expanding the gesture vocabulary and supporting continuous sign-language translation. Overall, the reviewed studies demonstrate that CNN-based architectures are useful for spatial feature extraction, while recurrent networks and Transformers can learn temporal relationships in sign-language videos. Multimodal approaches can further improve visual representation, and Generative AI and Large Language Models can help convert recognized sign representations into meaningful sentences. However, many existing approaches still face challenges related to limited datasets, signer variation, continuous sign recognition, multilingual

translation, and computational complexity. Therefore, an efficient framework that combines visual feature extraction, temporal recognition, and natural-language generation can provide a useful direction for developing accessible sign-language communication systems.

These studies have revealed that researchers have been investigating different methods to enhance sign language recognition and translation. CNN-based models can be employed to extract visual features, and recurrent networks and Transformers can be used to understand the temporal relationships between the frames of a video. Multimodality helps to better leverage visual information, while Generative AI and LLMs can transform identifiable signs into meaningful sentences. Most of the current systems, however, are limited to gesture recognition for isolated gestures or only limited support is available for continuous signing, generation of natural sentences, multilingual translation, and speech synthesis.

Other forms of difficulties include the small size of the ISL datasets, variations in signing, variations in the lighting and background, and the high computational needs of deep learning models. The proposed project is the integration of sign recognition, sentence generation, multilingual translation and speech synthesis into a single web-based system Multilingual Sign Language Translation Using Computer Vision and Generative AI.

The webcam video capture and processing use OpenCV, and the hand landmark detection and tracking use MediaPipe Hands. EfficientNet-B0 is used to extract spatial features from the video frames, while an Bidirectional GRU network is used to learn temporal features across consecutive video frames to recognize sign sequences.

Recognized sign labels (also called glosses) are fed into Generative AI to form meaningful, grammatically correct sentences. The sentences thus generated are then translated to English, Telugu, Hindi and Tamil and converted to speech using Text-to-Speech (TTS) module. The key value added by the proposed system is the development of these as an easy to use application that will enable better communication between and more accessibility for different language communities.

TABLE I. Contributions of various research on Sign Language Recognition and Translation on different datasets.

| | | | | | |
|---|---|---|---|---|---|
| Gadarla and Mane [11] | 2024 | Multilingual sign language recognition using 11 datasets | ML.NET-based machine learning approach | Reported accuracies of approximately 92.33% and 89.4% | Need for more languages, larger vocabularies, sentence recognition, and multilingual TTS |
| Shahin and Ismail [12] | 2024 | Sign language translation and NLP architectures | Transformer Encoder–Decoder approaches | BLEU,ROUGE discussed; | Differences between sign-language grammar and spoken-language grammar |
| Virag and Katona [13] | 2026 | Real-time hand-sign recognition; more than 5,600 annotated images | RetinaNet + ResNet50 | More than 93% accuracy | Focus on isolated signs rather than continuous signing |
| Alaftekin et al. [14] | 2024 | Turkish sign-language numbers (0–9) | YOLOv4-CSP + Transformer | 98.95% precision, 98.15% recall, 98.55% F1-score, and 99.49% mAP | Limited to isolated gestures, without continuous sentence translation |
| Violet and Sri [15] | 2025 | ISL-CSLTR; 700 videos, 100 sentences, and 7 signers | CNN + MediaPipe + Bi-LSTM + NMT + Dynamic GAN | >95% ,BLEU 38.06 | Limited dataset size and signer diversity |

## *3 METHODS AND MATERIALS*

The project, titled 'Multilingual Sign Language Translation Using Computer Vision and Generative AI', will aim to identify Indian Sign Language (ISL) gestures and translate them into a relevant text and speech in various languages. This system features computer vision, deep learning, and generative AI models that interpret hand gestures, interpret sign sequences, create natural-language sentences and output in multiple languages. These key technologies are: OpenCV, MediaPipe Hands,
EfficientNet-B0,Bidirectional Gated Recurrent Unit (Bi-GRU), Generative AI, machine translation, and Text-to-Speech (TTS).The proposed method involves video acquisition, video preprocessing, hand detection, spatial features' extraction, temporal sequence recognition, sentence generation, multilingual translation, and speech generation.

### A. Dataset Collection

The proposed Indian Sign Language recognition model was trained and evaluated using the publicly available INCLUDE dataset. The dataset contains isolated ISL sign videos representing different sign classes. For this study, 4,284 video samples belonging to 262 normalized sign classes were used. The videos were associated with their corresponding sign labels and organized into training, validation, and testing sets. The dataset was divided into 3,046 training samples, 619 validation samples, and 619 testing samples.

Each video was processed to extract a sequence of 32 frames for feature extraction and temporal recognition. The INCLUDE dataset was used for the quantitative evaluation of isolated sign recognition. Although other resources, such as ISLTranslate, provide continuous sign-language videos and sentence-level annotations, they were not used in the training or quantitative evaluation of the recognition model in this study.

### B. Video acquisition and pre-processing

The input to the proposed system is a live video stream obtained from a webcam or a video of a sign language message. Video is captured and basic image processing is done using OpenCV library.The video captured is split up into frames and hand movements expressed over time are analyzed. Pre-processing involves the shrinking of frames, conversion of image formats, if necessary, and preparing hand regions for feature extraction. The frames are processed in the same form to ensure consistent frames for model training and model recognition.The preprocessed frames are then given to the MediaPipe Hands module to detect hand landmarks and hand detection.

### C. Hand Detection with MediaPipe Hands

Hand movements are detected and tracked from the input video frames using the MediaPipe Hands. It is able to detect the 21 important landmarks on each hand, which are the key joints and fingertip positions.

The landmarks offer hand position, finger movement and hand structure information. The extracted landmarks are used to identify the hand area and to analyse the gestures of sign language.The hand landmark detection process is shown as follows:

$L_t = H_(\theta)(I_t)$

Where:

- $I_t =$ is the video frame input at time (t).
- $H_(\theta)$ = is a function that is implemented by MediaPipe Hands.

The detected hand landmarks are represented by the variable, $L_t$.

### D. Spatial Feature Extraction with EfficientNet-B0

The hand information detected is sent to the feature extraction stage. The efficientNet-B0 model is employed to extract spatial features. Here, efficientNet-B0 is used for spatial feature extraction.

The processed hand images are passed as input to Efficientnet-B0 which extracts the most crucial spatial features. It is a convolutional neural network (CNN) architecture which learns visual patterns from images via convolutional operations.

The subsequent processed hand images are fed into EfficientNet-B0 in the suggested system. The network captures key visual features like hand shape, arrangement of fingers, and hand pattern that relate to gestures. Finally, the last classification layer is removed and/or replaced in order to obtain the feature representations rather than the final sign prediction.

The feature extraction process is depicted as following:

$F_t = E_(\theta)(I_t)$

Where:
$I_t$ = is the processed frame or hand image.
The feature extraction network is EfficientNet-B0 denoted as $E_(\theta)$.
- $F_t$ = denotes the extracted feature vector at time .

The feature vectors obtained are kept in the same order and are fed to the GRU network.

E. Temporal Sequence Recognition with GRU

Gated Recurrent Unit (GRU): This is a type of Recurrent Neural Network (RNN) used for processing sequential data. It is employed in the proposed system for learning the temporal patterns of hand movements from consecutive frames of videos.

Unlike CNN, which mainly extracts spatial information, BiGRU analyzes the variation of the features extracted over time. It has two important gates: Update Gate and Reset Gate. The gates determine what information will be retained and what new information will be taken into account.
The operations of the GRU are as follows
1. Update Gate
The amount of information to retain from the previous hidden state is set by the update gate.
$z_t = \sigma(W_{(z)}F_t + U_{(z)}h_{t-1} + b_{(z)})$
2. Reset Gate
The reset gate determines the amount of the previous information that is used when determining the new candidate state.
$r_t = \sigma(W_{(r)}F_t + U_{(r)}h_{t-1} + b_{(r)})$
3. Candidate Hidden State
The new information from the current input and the previous hidden state is stored in the candidate hidden state.
$\tilde{h}_t = \tanh(W_{(h)}F_t + U_{(h)}(r_t \odot h_{t-1}) + b_{(h)})$
4. Final Hidden State
The last hidden state is a fusion of the previous and new information.
$h_t = (1 - z_t) \odot h_{t-1} + z_t \odot \tilde{h}_t$
Where:

- The feature vector taken at time t is called $F_t$.
- $h_{t-1}$ is the previous hidden state.
- The update gate is $z_t$.
- The reset gate is represented with $r_t$.
- candidate hidden state is denoted by $\tilde{h}_t$.
- The trainable weights and biases are represented by W, U and b.
- σ is the sigmoid activation function.
- The hyperbolic tangent activation function is represented by tanh, which is used to help control the output range of the GRU.
- ⊙ represents element-wise multiplication.

The final temporal representation generated by the Bidirectional GRU is passed to the fully connected classification layers to predict the corresponding ISL sign class. By processing feature vectors across consecutive video frames in both forward and backward directions, the BiGRU learns temporal patterns associated with each isolated sign. This enables the model to recognize hand movements using information from the complete input sequence rather than relying on individual frames alone.
.

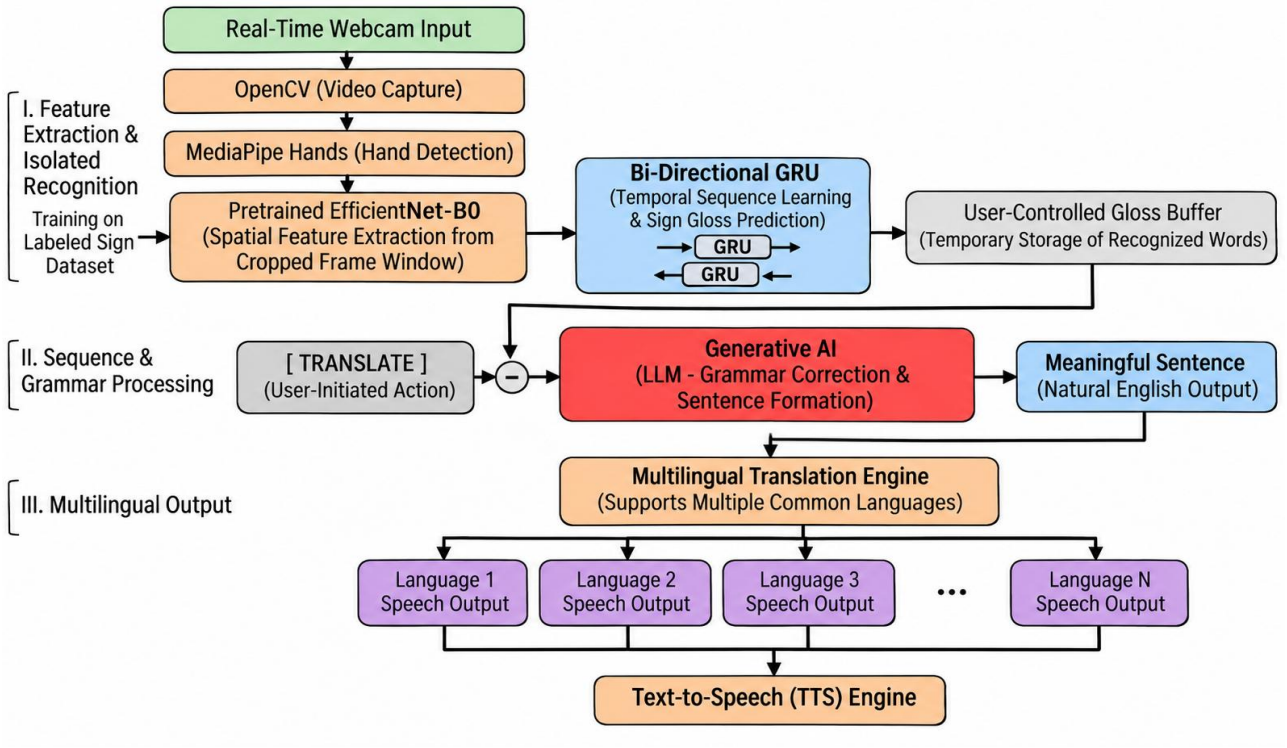



*Fig 3.1 Proposed System Architecture*

F. User-Controlled Gloss Buffer

After the recognition of each isolated sign, the predicted sign label, also known as a gloss, is checked against the existing entries in the buffer. A duplicate-check mechanism prevents an already stored gloss from being added again. If the predicted gloss is not already present, it is added to the user-controlled buffer. This allows users to review and manage the recognized signs before sentence generation.

The system does not automatically generate a sentence after every sign prediction. Instead, the user manually presses the Translate button when the signing input is complete. Only then are the stored glosses submitted to the Generative AI module to generate a grammatically meaningful sentence. This approach provides user control over the sentence-generation process and helps avoid unnecessary or repeated words.

G. Natural Sentence Generation with Generative AI

The recognized sign labels and/or the sequence of gloss are delivered to the Generative AI module after sign recognition. This module will aim to translate the recognised sign representation back into meaningful and grammatically correct sentences.

Sign-language word order can vary from the word order of spoken languages. Thus, Generative AI can be used to organize the identified signs into natural-language statements, while maintaining the underlying meaning.

The sentence generation process can be represented as:

$S = G_{(\varphi)}(Z)$

Where:

(Z) is the known sequence of signs.

$G_{(\varphi)}$ is the Generative AI model.

(S) is the sentence generated in natural language.

In the implementation, Google Gemini (gemini-3.5-flash-lite) is used as the primary Generative AI model for sentence generation. The recognized glosses are submitted to the model only after the user manually presses the Translate button. The model generates grammatically meaningful sentences while preserving the intended meaning of the recognized signs. A local rule-based NLP provider is also included as an offline fallback.

H. Multilingual Translation

The sentence generated is given to a multilingual translation module to get the sentence in the desired language. The proposed system will be able to support four languages; English, Telugu, Hindi and Tamil.

The user may choose which language he/she would like to output the sentence in, and the translation module

translates the generated sentence into that desired language, keeping the meaning the same.

The translation process is shown as:

$T_l = M_{(\psi)}(S, L)$

Where:

(S) is a generated sentence.

(L) indicates the selected target language.

The module for machine translation is called $M_{(\psi)}$

$T_l$ is the translated text.

The translation module is implemented using Google Translate through the deep-translator Python library and its GoogleTranslator interface. Gemini-based multilingual translation is also supported when available. The system supports English, Telugu, Hindi, and Tamil. An offline dictionary-based fallback is provided for selected essential phrases.

I. Text-to-Speech Conversion

The translated text is then passed to the Text-to-Speech (TTS) module, which converts the text into speech in the specified language. TTS translates the text and outputs spoken words for the users to understand the translated sign-language content.

The generation process of the speech is shown as:

$A_l = TTS(T_l, L)$

Where:

$T_l$, L is the translated text.

Selected Language is indicated by (L).

The audio output is the $A_l$ result.

The Text-to-Speech module is implemented using Microsoft Edge Neural Text-to-Speech through the edge-tts Python library. Language-specific neural voices are configured for English, Telugu, Hindi, and Tamil. The gTTS library is included as a fallback speech-generation option.

J. Web Application Integration

This module enhances communication skills through the input of both text and speech translation integrating Web Application.

The proposed system is integrated with a web based application to have the user-friendly interface. Frontend is implemented in React with Vite and users can interact with the system via web browser.

Sign-language input can be done using a webcam or video source. The backend processes the recognition module and translation module and feeds the output of the translation to the frontend, along with the translated text and the speech output.

The application also includes a conversation history to enable users to access translations generated in the past. This integration enables the system to be accessible and allows for real-time interaction, depending on the processing speed and model performance.

K. Performance Evaluation

The proposed system can be evaluated using classification metrics, translation-quality metrics, and processing-time measurements. In this study, the isolated ISL recognition model was quantitatively evaluated using accuracy, macro precision, macro recall, and macro F1-score. The real-time webcam recognition pipeline was separately evaluated using processing latency and frame rate.

**Accuracy:** Measures the proportion of correctly classified samples among all test samples.

Accuracy = (Correct Predictions / Total Predictions) × 100

**Precision:** Measures the proportion of correctly predicted positive samples among all samples predicted as positive.

Precision = TP / (TP + FP)

**Recall:** Measures the proportion of actual positive samples correctly identified by the model.

Recall = TP / (TP + FN)

**F1-Score:** Represents the harmonic mean of precision and recall.

F1-Score = 2 × (Precision × Recall) / (Precision + Recall)

In this study, precision, recall, and F1-score are reported using macro averaging across the sign classes.

**BLEU Score:** Can be used to evaluate the similarity between generated translations and reference sentences.

**Word Error Rate (WER):** Can be used to measure the difference between predicted and reference word sequences.

**Processing Time:** Measures the time required for the vision-based recognition pipeline to process an input frame.

Processing Time = End Time − Start Time

BLEU and WER were not quantitatively evaluated in this study because independent reference translations were not used. The reported processing-time measurements cover the webcam-based vision recognition pipeline and exclude external Generative AI, translation, and Text-to-Speech response times.

## 4. RESULTS AND DISCUSSIONS

A. Experimental Setup

The Indian Sign Language (ISL) Recognition System aims to identify isolated ISL words in video via computer vision and deep learning approaches. The system integrates the hand detection, tracking with MediaPipe, spatial feature extraction with EfficientNet-B0, and temporal sequence learning with Bidirectional Gated Recurrent Unit (BiGRU).

The proposed model was evaluated and trained using the INCLUDE Indian Sign Language (ISL) dataset. The dataset includes 4284 video samples of 262 video classes, which have been normalized. The data set was split into training, validation and test sets. There are 3,046 video samples in the training set, 619 video samples in the validation set, and 619 video samples in the testing set.

First, a process is applied to video sequences to retrieve 32 frames which correspond to the sign gesture. Hand detection and landmark tracking uses MediaPipe. The feature representations of the extracted frames are used in a pre-trained model EfficientNet-B0 to get the spatial feature representation. A feature vector of 1,280 dimensions is generated for each frame. The sequential feature vectors are then fed into a two-layer Bidirectional GRU network, which learns temporal patterns with respect to the sign gestures.

The model uses a two-layer Bidirectional GRU network with a hidden size of 128 in each direction. The final temporal representation is processed by a Layer Normalization layer with 256 features, followed by fully connected layers consisting of Linear (256–128), ReLU, Dropout (0.4), and Linear (128–262) classification layers. The output layer predicts one of the 262 ISL sign classes. The implementation uses PyTorch and was trained on Google Colab, using GPU acceleration initially and CPU execution when GPU resources became limited.

Training accuracy, validation accuracy, validation loss, and test accuracy were considered as a performance metric for the proposed model. These measurements were employed to gain an insight into the learning behaviour and classification ability of the proposed ISL recognition model.

B. Training and Validation Performance.

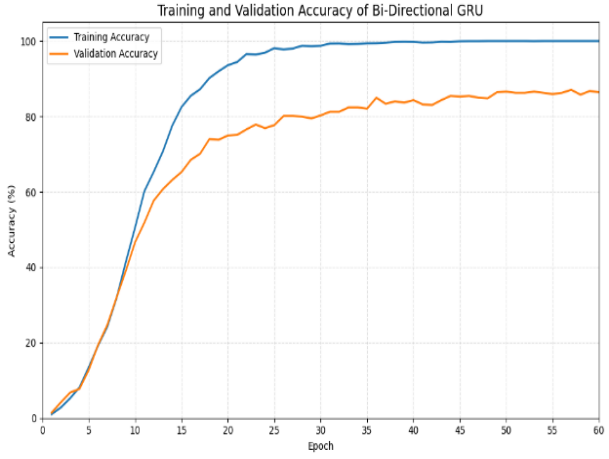


*Fig. 4.1. The accuracy of the proposed ISL Recognition Model during training and validation*.

For assessing a deep learning model's learning performance, it is crucial to consider training and validation accuracy. Training accuracy is the percentage of the correctly classified samples of the training set while validation accuracy is the percentage of the correctly classified samples of the independent set (not used to update the parameters of the model).

The proposed model combines EfficientNet-B0 and Bidirectional GRU to learn spatial and temporal information from ISL video sequences. During training, the training accuracy increased and reached approximately 100% in the later epochs. The selected checkpoint, corresponding to Epoch 57, achieved approximately 86.27% validation accuracy. The difference between training and validation accuracy indicates a generalization gap and suggests possible overfitting. Further improvements through data augmentation, regularization, and additional training data may help improve the model's generalization ability.

C. Training and Validation Loss Analysis

To get insight into the optimization behaviour of a deep learning model, the training loss and the validation loss are used. The training loss is used to predict errors on the training data while the validation loss is used to predict errors on the unseen data of the validation set.

The proposed model extracts spatial features with EfficientNet-B0 and temporal sequence features with a Bidirectional GRU network. In training, the network tunes its parameters to reduce the classification mistakes made and will get better at differentiating the various ISL signs.

The model checkpoint was selected based on the validation loss. The selected checkpoint corresponds to Epoch 57, with a validation loss of 1.1465 and a validation accuracy of approximately 86.27%. The validation loss represents the classification error measured on the validation dataset and was used to select the model checkpoint. The difference between training and validation performance indicates that further regularization and data augmentation may help improve the model's generalization ability.

The performance on the training set was around 100% in the later epochs, but the performance on the validation set was still around 86%. This difference suggests that the model was able to better learn the training samples than the validation samples that it had not seen before.

The gap between the training and validation performance is notable, which may indicate overfitting. Data augmentation, regularization, and other training experiments could be useful to enhance the model's generalization ability.

D. Confusion Matrix Analysis

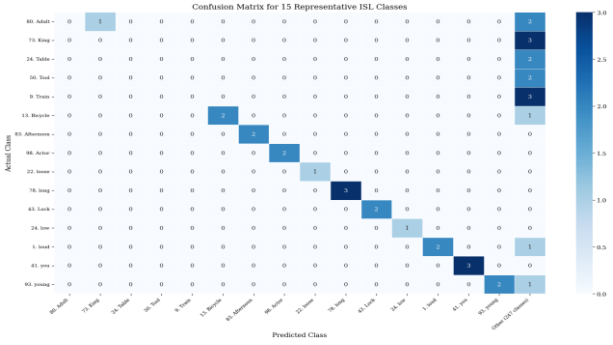



*Fig . 4.2 Confusion Matrix for sign Language Recognition*

One of the key evaluation techniques that can be used to get a better understanding of the classification performance of a model is a confusion matrix. It compares the actual class labels with the predicted class labels. The diagonal elements show correct classifications and off diagonal elements are incorrect predictions.The confusion matrix is used in the proposed ISL recognition model to analyse the effectiveness of the model to distinguish between the 262 sign classes.

The test set consists of 619 video samples. Of these, 537 samples were classified correctly and 82 samples were misclassified by the model. This yielded a global accuracy of 86.75% on the test.

The misclassified samples suggest that some of these sign gestures might be hard to differentiate between. Hand shape, hand motion, and video appearance can impact the recognition process.

E. Classification Performance Analysis.

The classification performance of the proposed ISL recognition model was evaluated using accuracy, macro precision, macro recall, and macro F1-score. The evaluation was performed on 619 test video samples covering 262 ISL classes. Accuracy measures the proportion of correctly classified samples, while macro precision, macro recall, and macro F1-score provide an average assessment of classification performance across the individual classes.

The model correctly classified 537 samples and misclassified 82 samples, achieving a test accuracy of 86.75%. The macro precision was 89.13%, macro recall was 88.20%, and macro F1-score was 87.35%

The following results were found:

| Evaluation Metric | Result |
|---|---|
| Total Test Samples | 619 |
| Correct Predictions | 537 |
| Incorrect Predictions | 82 |
| Test Accuracy | 86.75% |
| Macro Precision | 89.13% |
| Macro Recall | 88.20% |
| Macro F1-Score | 87.35% |

The results indicate that the proposed combination of EfficientNet-B0 and Bidirectional GRU can recognize isolated ISL signs with good classification performance. The macro-averaged metrics provide additional information about the model's performance across the 262 sign classes

F. Overall Model Performance

The best result from the validation checkpoint and the test evaluation results are summarized to show the overall performance of the proposed ISL recognition model.

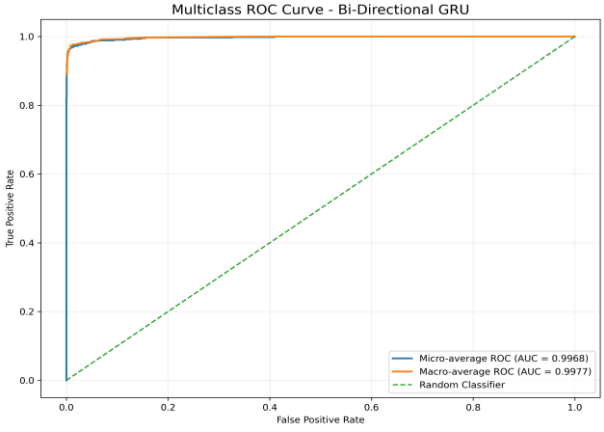



Fig 4.3 Receiver Operating Characteristic (ROC) Curve for ISL Sign Classification.

TABLE II. Summary of Dataset and Model Performance

| Dataset | Include |
|---|---|
| Total Video Samples | 4,284 |
| Number of Classes | 262 |
| Training Samples | 3,046 |
| Validation Samples | 619 |
| Testing Samples | 619 |
| Input Sequence Length | 32 Frames |
| Feature Dimension | 1,280 |
| Validation Accuracy at Selected Checkpoint | 86.27% |
| Validation Loss at Selected Checkpoint | 1.1465 |
| Test Samples | 619 |

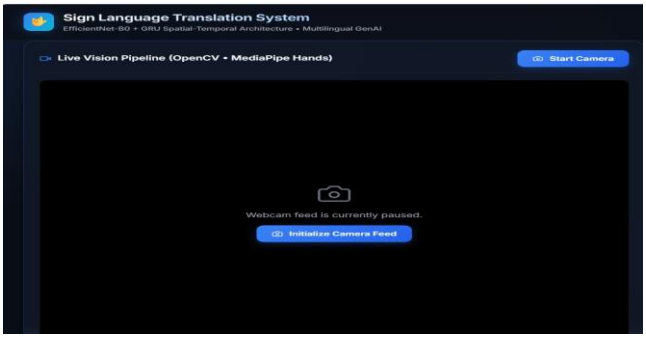
| | |
|---|---|
| Correct Test Predictions | 537 |
| Incorrect Test Predictions | 82 |
| Test Accuracy | 86.75% |

The results show that the proposed architecture achieves effective isolated ISL word recognition. The selected checkpoint achieved 86.27% validation accuracy, while the model achieved 86.75% test accuracy on 619 unseen samples. These results demonstrate the model's classification performance on the evaluated dataset.

The evaluated model provides the foundation for the ISL recognition application. Further evaluation with different signers, recording environments, lighting conditions, and backgrounds will be useful to assess its generalization under real-world conditions.

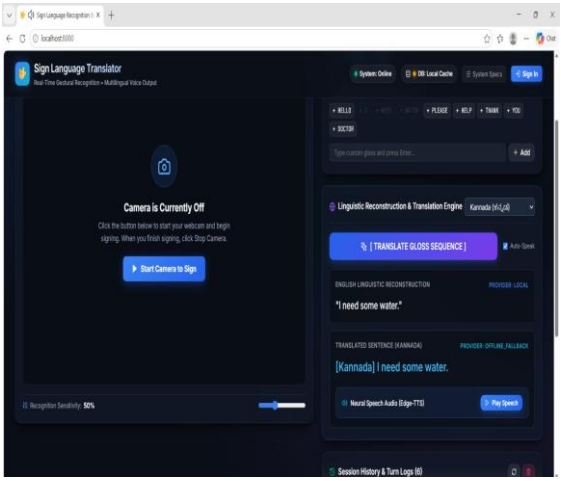
G. System Interface and Implementation

1. Live Vision Pipeline Interface




The live vision pipeline interface is the input part of proposed ISL recognition application. It's meant to take video from a webcam, and analyze the frames for the recognition of signs.Video capture and frame processing is done using OpenCV, hand detection and landmark tracking are done using MediaPipe. The information extracted from the visual data is fed through the feature extraction and recognition process.This interface offers an interactive input of users using sign language gestures. The implementation of the web camera is a way of linking the visual input with the proposed recognition architecture.In addition to evaluation using the prepared video dataset, the real-time webcam recognition pipeline was assessed separately using processing latency and frame rate measurements, as described in Section 4.G.3.

2. Sign Language Recognition Interface




The sign language recognition interface is the interface for the application level interaction in displaying the recognized ISL words. The interface is created to interface the input video module to the trained recognition model.Relevant spatial and temporal features of the input video sequence are extracted. The trained BiGRU classifier infers the corresponding sign class and the recognized word can be visualized as a text in the application. The interface provides the foundation for integrating additional communication features, including Generative AI-based sentence generation, multilingual

translation, and speech output. These components extend the isolated-word recognition system. However, the classification metrics reported in this study evaluate only sign recognition and do not measure the quality or effectiveness of sentence generation, translation, or speech output.

**3. Real-Time Webcam Performance**

The real-time webcam recognition pipeline was evaluated based on its processing latency and frame rate. The system achieved an average per-frame processing latency of 36.81 ms, with a median (P50) latency of 36.31 ms and a 95th percentile (P95) latency of 40.21 ms. The measured processing speed was approximately 27.16 frames per second (FPS), demonstrating the ability of the vision-based recognition pipeline to perform real-time sign recognition on a standard CPU. These measurements include hand landmark detection, spatial feature extraction, and temporal inference. They do not include the response time of external Generative AI, translation, or Text-to-Speech services.

H. Discussion

The experimental results demonstrate the effectiveness of the proposed MediaPipe, EfficientNet-B0, and Bidirectional GRU architecture for isolated Indian Sign Language word recognition. The model achieved approximately 100% training accuracy in the later epochs, while the selected checkpoint achieved 86.27% validation accuracy with a validation loss of 1.1465 at Epoch 57. On the independent test set of 619 samples, the model achieved 86.75% accuracy, correctly classifying 537 samples.

The difference between training and validation performance indicates a generalization gap and suggests possible overfitting. Data augmentation, regularization, and additional training data may help improve generalization. The results demonstrate the usefulness of combining spatial feature extraction through EfficientNet-B0 with temporal feature learning through BiGRU.

However, the evaluated model focuses on isolated-word recognition across 262 ISL classes. Continuous sign language recognition and sentence-level translation have not been quantitatively evaluated in this study. Further work is required to extend the system and independently assess these capabilities under real-world conditions.

A comparison with the original INCLUDE dataset study by Sridhar et al. provides a useful reference point. Their reported recognition accuracy was 85.6% on the full 263-class INCLUDE dataset, whereas the proposed model achieved 86.75% accuracy on 619 test samples across 262 ISL classes. Although the proposed model shows a numerical improvement of 1.15 percentage points, the class count and experimental configurations are not identical. Therefore, this comparison should be interpreted as a reference to previously reported results rather than a direct benchmark under identical conditions.

I. Conclusion of Experimental Results

The Indian Sign Language recognition model was trained and evaluated using the INCLUDE dataset. MediaPipe, EfficientNet-B0, and Bidirectional GRU were used to extract spatial and temporal information from sign-language video sequences. The selected checkpoint at Epoch 57 achieved a validation accuracy of approximately 86.27% and a validation loss of 1.1465. The model achieved a test accuracy of 86.75% on 619 test samples from 262 ISL classes. Although the training accuracy reached approximately 100% in the later epochs, the difference between training and validation performance indicates a generalization gap. Future work will focus on improving generalization, evaluating class-wise performance, and testing the system under different real-world conditions.

## 5. CONCLUSION

This paper presents a multilingual Indian Sign Language (ISL) translation framework designed to reduce communication barriers between sign language users and people who do not understand sign language. The proposed recognition architecture integrates OpenCV for video capture, MediaPipe Hands for hand landmark detection, EfficientNet-B0 for spatial feature extraction, and a Bidirectional GRU network for temporal feature learning.

The model was trained and evaluated using the INCLUDE dataset containing 262 ISL sign classes. Experimental evaluation on 619 test samples achieved an accuracy of 86.75%, with 537 correct predictions. The results demonstrate the effectiveness of the proposed architecture for isolated ISL word recognition.

The framework also incorporates a user-controlled gloss buffer, Generative AI-based sentence generation, multilingual translation into English, Telugu, Hindi, and Tamil, and Text-to-Speech output. These components provide an integrated communication workflow; however, their sentence-generation quality, translation accuracy, and speech performance have not been quantitatively evaluated in this study.

Future work will focus on improving recognition generalization, supporting continuous sign language recognition, expanding the vocabulary, and conducting independent evaluations of sentence generation, multilingual translation, and speech output under real-world conditions.

**Conflict of Interest**

The authors declare that they have no conflict of interest related to this research and publication.

**Declaration of AI Tools**

ChatGPT (OpenAI) was used to assist with language refinement and editing during the preparation of this manuscript. Google Gemini was used as a component of the proposed system for on-demand sentence generation, as described in the methodology. The authors reviewed the manuscript and take full responsibility for its content.

**6. REFERENCES**

[1] M. Geetha, N. Aloysius, D. A. Somasundaran, A. Raghunath, and P. Nedungadi, “Toward real-time recognition of continuous Indian Sign Language: A multi-modal approach using RGB and pose,” *IEEE Access*, vol. 13, pp. 60270–60283, 2025.

[2] Z. Wang, D. Li, R. Jiang, and M. Okumura, “Continuous sign language recognition with multi-scale spatial-temporal feature enhancement,” *IEEE Access*, vol. 13, pp. 5491–5506, 2025.

[3] M. Kumar, S. S. Visagan, T. Mahajan, A. Natarajan, and P. S. Sreeja, "Enhanced Sign Language Translation Between American Sign Language and Indian Sign Language Using LLMs," IEEE Access, vol. 13, pp. 156270–156284, 2025.

[4] K. A. Shams, M. R. Reaz, M. R. U. Rafi, S. Islam, M. S. Rahman, R. Rahman, M. T. Reza, M. Z. Parvez, S. Chakraborty, B. Pradhan, and A. Alamri, “MultiModal Ensemble Approach Leveraging Spatial, Skeletal, and Edge Features for Enhanced Bangla Sign Language Recognition,” IEEE Access, vol. 12, pp. 83638–83657, 2024, doi: 10.1109/ACCESS.2024.3410837.

[5] M. Y. Novopoltsev, A. Tulenkov, R. Murtazin, R. I. Akhidov, I. Zemtsova, E. E. Bojarskaja, D. N. Bondarenko, A. V. Savchenko, and I. Makarov, “From Data to Signs: A Foundation Model for Multilingual Sign Language Recognition,” IEEE Access, vol. 13, pp. 188170–188181, 2025, doi: 10.1109/ACCESS.2025.3627130.

[6] Y. Said, S. Boubaker, S. M. Altowaijri, A. A. Alsheikhy, and M. Atri, "Adaptive Transformer-Based Deep Learning Framework for Continuous Sign Language Recognition and Translation," *Mathematics*, vol. 13, no. 6, Art. no. 909, Mar. 2025

[7] Z. Liu, J. Wu, Z. Shen, X. Chen, Q. Wu, Z. Gui, L. Sen, and H. Shu, "Improving End-to-End Sign Language Translation With Adaptive Video Representation Enhanced Transformer," IEEE Transactions on Circuits and Systems for Video Technology, 2024.

[8] F. M. Najib, “Sign Language Interpretation Using Machine Learning and Artificial Intelligence,” Neural Computing and Applications, vol. 37, no. 2, pp. 841–857, 2025, doi: 10.1007/s00521-024-10395-9.

[9] X. Shen, L. Shen, S. Yuan, H. Du, H. Sun, and X. Yu, "Diverse Sign Language Translation," International Journal of Computer Vision, vol. 134, p. 329, 2026.

[10] F. M. Talaat, W. El-Shafai, N. F. Soliman, A. D. Algarni, F. E. Abd El-Samie, and A. I. Siam, "Real-time Arabic avatar for deaf-mute communication enabled by deep learning sign language translation," Computers and Electrical Engineering, vol. 119, p. 109475, 2024.

[11] S. K. Gadarla and S. B. Mane, “Multilingual Sign Language Recognition,” TechRxiv preprint, 2024.

[12] N. Shahin and L. Ismail, “From Rule-Based Models to Deep Learning Transformers Architectures for Natural Language Processing and Sign Language Translation Systems: Survey, Taxonomy and Performance Evaluation,” Artificial Intelligence Review, vol. 57, no. 10, Art. no. 271, 2024, doi: 10.1007/s10462-024-10895-z.

[13] N. Virag and J. Katona, “AI-powered real-time hand sign recognition: Enhancing communication for the hearing- and speech-impaired community,” *Universal Access in the Information Society*, vol. 25, p. 73, 2026.

[14] M. Alaftekin *et al*., “Real-time sign language recognition based on YOLO algorithm,” *Neural Computing and Applications*, vol. 36, pp. 7609–7624, 2024.

[15] I. M. M. Violet and R. Leena Sri, "A Comprehensive Framework on Recent Advances and Challenges in Sign Language Recognition Systems," Discover Artificial Intelligence, vol. 5, p. 419, 2025.
[16] A. Sridhar, R. G. Ganesan, P. Kumar, and M. Khapra, "INCLUDE: A Large Scale Dataset for Indian Sign Language Recognition," in *Proceedings of the 28th ACM International Conference on Multimedia (MM '20)*, 2020, pp. 1366–1375, doi: 10.1145/3394171.3413528.