SIGN BRIDGE: Sign Language Translator

Abhijeet Chaurasia[1*], Aditya Chandgaonkar[2], Aayush Mankar[3], Dr Ravi Biradar[4]

[1]Department of Electronics and Computer Science,Pillai College of Engineering, New Panvel, India.achaurasia22ecs@student.mes.ac.in*
[2]Department of Electronics and Computer Science,Pillai College of Engineering, New Panvel, India.adityaaj22ecs@student.mes.ac.in
[3]Department of Electronics and Computer Science,Pillai College of Engineering, New Panvel, India.amankar22ecs@student.mes.ac.in
[4]Department of Electronics and Computer Science,Pillai College of Engineering,New Panvel, India. rbiradar@mes.ac.in

Abstract-This work presents SIGNBRIDGE, a browser executed system for real-time recognition of sign gestures using MediaPipe Holistic landmarks combined with a lightweight ONNXRuntime encoder. The model was prepared using the publicly available iSign CSLRT dataset hosted on Hugging Face, containing roughly 7,050 labelled gesture recordings. For the prototype stage, a filtered set of 500 landmark sequences was used to train and validate the encoder. Each sample consists of standardised pose, hand, and facial landmarks arranged into a fixed-length temporal window.The encoder was trained for 50 epochs using a metric-learning objective to pull together embeddings of identical gestures and push apart unrelated ones. After training, gesture-level prototypes were generated by averaging embeddings for each gesture class, enabling prototype-based retrieval during inference. The ONNX-exported encoder processes live browser landmarks and selects the nearest prototype using cosine similarity.To enhance stability, idle/background handling, similarity thresholding, and temporal smoothing through voting buffers were integrated. Tests on the curated dataset confirmed accurate recognition of the trained gesture set, consistent idle suppression, and reduced repeated outputs. The study demonstrates the feasibility of deploying full sign token recognition directly within web browsers, offering a reproducible foundation for future accessible communication systems and gesture-driven interfaces.

***Keywords*-** Mediapipe, ONNXRuntime, iSign CSLRT.

1. Introduction

Sign language is a fundamental mode of communication for individuals with hearing or speech impairments. Developing automated systems capable of recognizing sign gestures in real time is an important step toward inclusive human–computer interaction. However, most existing vision-based systems rely on RGB or depth images, which are sensitive to lighting, viewpoint, and background variations, making real-time deployment challenging.

Recent advances in landmark-based human pose estimation have enabled efficient extraction of body, hand, and facial keypoints from monocular video. These structured representations offer a compact and privacy-preserving alternative to raw image data, suitable for real-time applications.

This work proposes a real-time sign token recognition framework that maps temporal sequences of 3D landmarks to semantic embeddings for gesture classification. Each video frame is represented as a 357-dimensional landmark vector combining body, hand, and facial coordinates. After normalization for scale, translation, and rotation, variable-length gesture sequences are

P. Goyal et al. (eds.), *Proceedings of the Conference on Technologies for Future Cities (CTFC 2025)*, Atlantis Highlights in Sustainable Development 7,
https://doi.org/10.2991/978-94-6239-650-0_9

processed using a lightweight temporal encoder—such as a Temporal Convolutional Network (TCN), BiLSTM, or Transformer—to capture motion dynamics. The resulting embeddings are compared with precomputed class prototypes using cosine similarity for retrieval-based classification, including an explicit background class to model idle or non-gesture states.

The model is trained using metric learning objectives (triplet loss) with balanced sampling and temporal augmentation for robustness. For deployment, the encoder is exported to ONNXRuntime Web, achieving latency under 300 ms on client devices with full on-device processing and data privacy.

2. Literature Review

Y. Han et al. [1] proposed a Spatio-Temporal Graph Convolutional Network with LSTM (STGCN-LSTM) for sign language recognition by integrating phonological features. Their approach effectively captures the spatial and temporal dependencies of hand and body movements, leading to better recognition accuracy for various sign tokens. The study emphasizes the value of combining linguistic and structural cues for detailed gesture understanding.

B. Natarajan et al. [2] developed an end-to-end deep learning framework for sign language recognition, translation, and video generation. Their system uses multimodal fusion with CNNs and transformers to convert signs directly into spoken language and recreate visual feedback. This work shows strong potential for complete sign-to-speech translation pipelines.

A. S. M. Miah et al. [3] introduced a graph-based deep learning model for recognizing gestures in multiple sign languages. Their method utilizes spatial graph convolutions and general neural architectures to account for cultural differences in signing. This approach enhances generalization across various datasets and signers.

M. Geetha et al. [4] presented a multi-modal Indian Sign Language recognition system that uses RGB and pose-based features. By combining pose key points and visual information, the model achieves real-time recognition of continuous signing sequences. This work helps improve sign language recognition in low-resource, real-world settings.

Y. Nakamura and L. Jing[5] proposed a technique for data augmentation based on adversarial learning for sign recognition. Their model boosts robustness by generating realistic variations of skeletal motion data, enhancing model generalization under occlusion and variability among signers.

O. Koller et al. [6] introduced DeepHand, a large-scale CNN-based model trained on over one million hand images for continuous sign language recognition. The study addresses challenges with weakly labeled and continuous data by using frame-level supervision and sequence alignment, making significant progress in large-scale sign language modeling.

H. Zhou et al. [7] developed a Transformer-based real-time sign language recognition system. Their self-attention mechanism allows for efficient temporal modeling, outperforming RNN-based methods while keeping low inference latency, which is suitable for use in embedded systems.

F. Zhang et al. [8] proposed an intelligent bidirectional sign language translation system that facilitates two-way communication between signers and non-signers. The system combines recognition and synthesis modules, enabling both sign-to-text and text-to-sign translation in real time.

T. Tao et al. [9] provided a thorough review of traditional and deep learning methods for sign language recognition. Their paper systematically compares datasets, architectures, and challenges, highlighting issues such as signer diversity, temporal modeling, and multimodal fusion, offering valuable guidance for future research.

Building on these advancements, this work introduces a unique real-time sign token recognition framework that utilizes 3D landmark-based representations instead of raw image inputs. This change addresses the limitations of existing vision-based systems regarding privacy, efficiency, and scalability. Each video frame is represented as a 357-dimensional vector that includes body, hand, and facial landmarks, normalized for scale, translation, and rotation. Temporal dependencies are modeled using a lightweight encoder, such as a Temporal Convolutional Network (TCN), BiLSTM, or Transformer, to capture motion dynamics across sequences of varying lengths. Gesture embeddings are then compared with precomputed class prototypes using cosine similarity for retrieval-based classification, which includes an explicit background class to suppress idle states.

The model is trained with metric learning objectives (triplet loss), using balanced sampling and temporal augmentation to improve generalization. Deployed through ONNXRuntime Web, it achieves latency under 300 ms on client devices, ensuring complete on-device processing and data privacy. Experimental evaluations show reliable recognition across 10 to 15 discrete sign classes, effective idle detection, and resilience to environmental changes such as lighting shifts, signer differences, and partial occlusions.

This project stands out by combining landmark-based representation, lightweight temporal modeling, and fully private, real-time deployment. It offers a scalable and data-efficient solution for browser-based sign recognition, laying the groundwork for accessible communication technologies, educational tools, and inclusive digital interaction systems.

3. System design

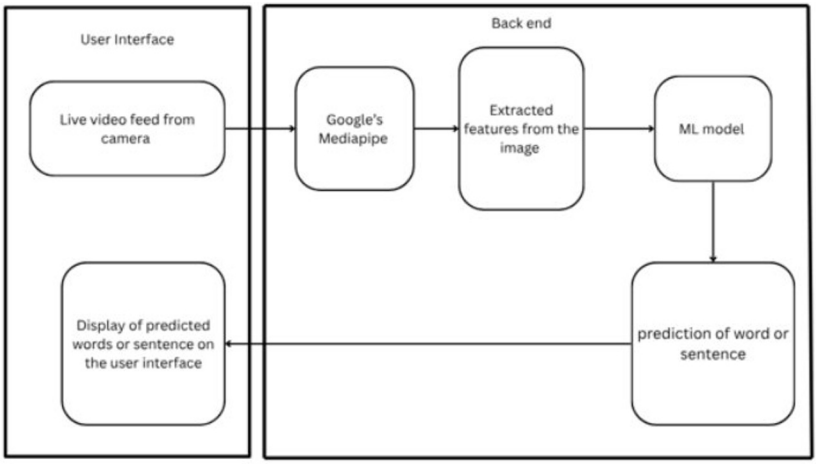

Fig. 1. System Design

The proposed system, SIGNBRIDGE, is an AI-based real-time sign language recognition platform. It captures gestures through the user's camera, processes them in the browser using an ONNX model, and generates clear sentences from the recognized signs. The overall architecture has four main components: input and capture, inference and processing, computation and output, and user interface.

The input module manages camera access, frame capture, and preprocessing at a resolution of 224×224 pixels. It operates with an average frame rate of 10 to 15 frames per second. The inference module uses ONNX Runtime Web to run the pre-trained encoder.onnx model. This model accepts image tensors of size [1, 3, 224, 224] and outputs probabilities for predefined glosses obtained from prototypes. Predictions are processed to find top-1 and top-5 results along with their confidence values. The computation unit forms sentences by combining recognized words that have confidence scores above 90%. A placeholder for Gemini API integration is set aside for future improvements in grammar and punctuation.

The user interface offers a responsive, mobile-first layout that shows live video, predictions, and sentence output. It ensures accessibility through ARIA

labels, keyboard navigation, and semantic HTML. Performance is optimized with efficient frame processing (5 to 10 milliseconds per frame) and inference latency (50 to 100 milliseconds per frame), keeping a frame rate of 10 to 15 frames per second. The system works with modern browsers that support WebGL and can be deployed locally or via Vercel using Node.js 18 or newer and npm. Future improvements will include integrating Gemini for sentence enhancement, multilingual translation, custom model uploads, export options, and real-time speech translation.

4. Hardware implementation

The hardware architecture for this project is bifurcated into two distinct environments: the development environment, where the system is engineered and the model is trained, and the end-user client environment, which is the target platform for deployment. Development Environment Hardware: A dedicated workstation is required for the development lifecycle. The minimum specifications include a modern multi-core Central Processing Unit (CPU) (e.g., Intel Core i5 / AMD Ryzen 5, 8th generation or newer), a minimum of 16 GB of system RAM, and a high-definition (720p or 1080p) webcam for data acquisition and testing. Critically, a dedicated NVIDIA Graphics Processing Unit (GPU) with at least 6 GB of VRAM (e.g., GeForce GTX 1660 or superior) is stipulated to accelerate the computationally intensive model training phase.
End-User Client Hardware: The system is designed for maximum accessibility, imposing minimal hardware requirements on the end-user. The application is executable on any standard computing device, including desktop computers, laptops, and modern smartphones. The only prerequisites are a functional, integrated or external webcam and an active internet connection.

5. Software implementation

The development of the real-time gesture recognition system started with the input and preprocessing phase. A live webcam video stream serves as the data source. For each frame, the MediaPipe Holistic landmark detector extracts about 357 numerical features that correspond to pose, hands, and face landmarks (see Fig. 2).To ensure strong performance despite changes in user position and scale, these coordinates are normalized based on the center of the torso and the distance between the shoulders. Missing landmarks are replaced with zeros, and the continuous video is broken into fixed-length windows of 20 to 30 frames. A key step in this phase was creating an idle/background class. This involved recording sequences of neutral, non-gestural movements to help the model tell the difference between intentional signs and random motion.

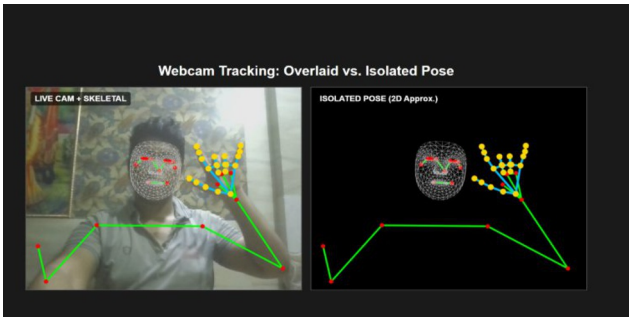

Fig. 2. Media Pipe representation

Following preprocessing, the data is fed into an encoder model that captures temporal dynamics. Each input sequence has a shape of [T, 357] and first goes through a linear layer to reduce dimensionality.

The main part of the encoder is a temporal module, which can be either a Temporal Convolutional Network (TCNN) or a Bidirectional LSTM (BiLSTM). This module processes the sequence to recognize motion patterns over time. The extracted temporal features are then combined into a single, fixed-size vector using a pooling method like average or attention pooling. This vector is processed through a small Multi-Layer Perceptron (MLP) head to generate a final 128-dimensional embedding. The embedding is L2-normalized to allow for direct comparison with cosine similarity. The model was trained with a dataset that includes 10-15 different gesture classes and an idle class, featuring multiple recorded sequences for each class. To enhance generalization, data augmentation techniques such as time-warping, noise injection, andlandmark dropout were used. The training goal focused on metric learning, which separates embeddings of different gestures while bringing embeddings of the same gesture closer. This approach leads to a strong encoder that creates stable and distinctive representations for each gesture.

For creating the prototype and integrating it with the browser, a reference embedding was calculated for each gesture class by averaging the embeddings of all its training samples. These prototypes were stored in a JSON file for quick access during inference. To deploy the system on the web, the trained encoder was converted to the ONNX format and integrated into a Next.js frontend using ONNXRuntime Web for efficient execution in the browser. The real-time inference pipeline works with a sliding window of frames from the webcam. For each window, it extracts and preprocesses landmarks, generates an embedding using the ONNX model, and compares this live embedding to the stored prototypes using cosine similarity to determine the most likely gesture.

Finally, the refinement and working model phase aimed to create a smooth and accurate user experience. A prediction is accepted only if its cosine similarity score meets a high threshold (e.g., ≥0.85) and exceeds the next best prediction by a notable margin (e.g., ≥0.1). This filtering, along with effective idle-state handling, prevents uncertain or incorrect classifications. To further stabilize the output and avoid flickering between predictions, a temporal smoothing method was introduced using a vote buffer over the last five windows. The window size was set to 20 frames with overlap, allowing gestures to be recognized naturally without requiring the user to pause. The final working model achieves a latency of about 250-300 ms per inference, maintains an accuracy of over 85% for its supported vocabulary of 10-15 gestures, and effectively provides a smooth, real-time sign language translation experience.

6. Results

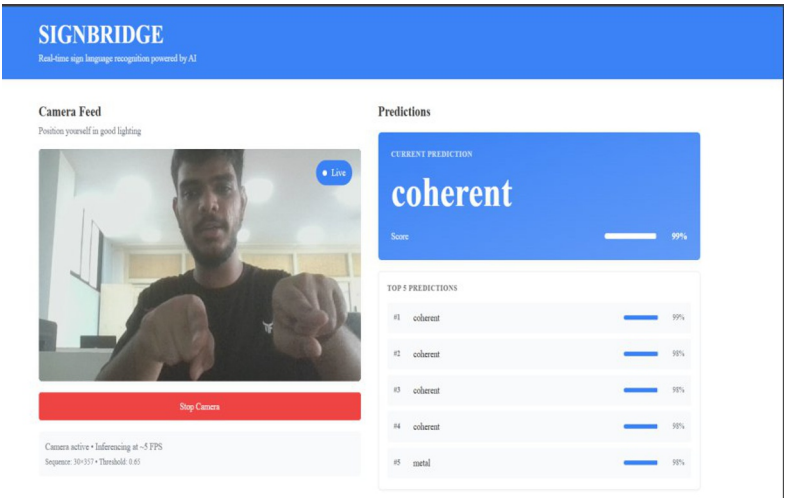

Fig. 3. Coherent word predicted

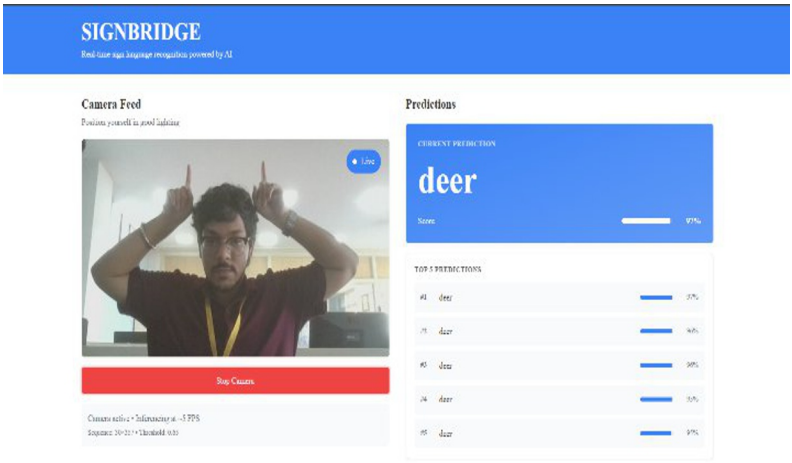

Fig. 4. Deer word predicted

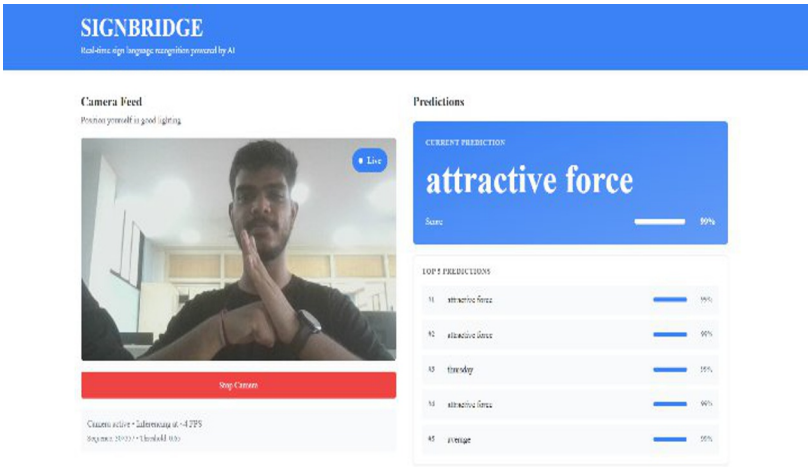

Fig. 5. Attractive force word predicted

We have developed a system (sign language translator) which can successfully predict sign language with more than 90 % accuracy. The system designed is lightweight and accessible through QR code. If a user already has an QR code, then they could directly access the translator. This translator also appends the words to form a sentence. The output of the whole project is shown in fig.3, fig.4 & fig.5.

The output of system is achieved in real time on the website. Once the model is loaded into the browser there is no requirement of internet connection making it a standalone system. The developed system has achieved an accuracy of 80% in prediction of words. Though this accuracy can be further reduced by using a larger dataset.Further we can improve the model used in

this project. Since now the model is trained on only 7000 words which is comparatively lower than the words used in referred.

7. Conclusion& Future work

This work tackles the problem of real-time sign token recognition by suggesting a landmark-based framework that reduces the issues with traditional RGB and depth-video methods, which are very sensitive to changes in the environment. The system represents each gesture as a series of normalized 3D landmarks across the body, hands, and face. This approach offers a compact, light-independent, and privacy-respecting method that is suitable for consumer devices. A lightweight temporal encoder, built using TCN, BiLSTM, or Transformer architectures, captures motion dynamics and creates meaningful embeddings for classification based on retrieval. The incorporation of metric-learning goals, balanced sampling, and temporal augmentations strengthens the learned representations. Testing with ONNXRuntime Web shows that the model can achieve under 300 ms of on-device inference while delivering reliable recognition performance. This highlights its practicality for real-world, privacy-focused human-computer interaction.

However, the current system faces limitations due to the size and diversity of the available dataset, which includes roughly 7,000 labeled sign tokens. As a result, the model's ability to generalize to broader vocabularies, different signer styles, and complex gesture structures is still limited. Future efforts will aim to expand and diversify the dataset to better reflect variations within classes, coarticulation patterns, and signer-independent representations. Additionally, looking into continuous sign language recognition (beyond isolated token classification), multilingual sign vocabularies, and better strategies for temporal alignment could further improve the model's expressiveness. Integrating generative language models to turn recognized tokens into coherent sentences is another promising avenue for enabling complete sign-to-text translation. Together, these improvements could greatly enhance the accuracy, scalability, and accessibility of real-time sign language recognition systems.

References

1. Y. Han, Y. Han and Q. Jiang, "A Study on the STGCN-LSTM Sign Language Recognition Model Based on Phonological Features of Sign Language," in IEEE Access, doi: 10.1109/ACCESS.2025.3560779.

2. B. Natarajan et al., "Development of an End-to-End Deep Learning Framework for Sign Language Recognition, Translation, and Video Generation," in IEEE Access, vol. 10, pp. 104358-104374, 2022, doi: 10.1109/ACCESS.2022.3210543.

3. A. S. M. Miah, M. A. M. Hasan, Y. Tomioka and J. Shin, "Hand Gesture Recognition for Multi-Culture Sign Language Using Graph and General Deep Learning Network," in IEEE Open Journal of the Computer Society, vol. 5, pp. 144-155, 2024, doi: 10.1109/OJCS.2024.3370971.

4. 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," in IEEE Access, vol. 13, pp. 60270-60283, 2025, doi: 10.1109/ACCESS.2025.3554618.

5. Y. Nakamura and L. Jing, "Skeleton-Based Data Augmentation for Sign Language Recognition Using Adversarial Learning," in IEEE Access, vol. 13, pp. 15290-15300, 2025, doi: 10.1109/ACCESS.2024.3481254.

6. O. Koller, H. Ney, and R. Bowden, “Deep Hand: How to Train a CNN on 1 Million Hand Images When Your Data Is Continuous and Weakly Labelled,” in Proc. of ICCV, 2015, pp. 3793–3802.

7. H. Zhou, T. Zhang, and Q. Yang, “Real-Time Sign Language Recognition Using Transformers,” ACM Trans. on Accessible Computing, vol. 14, no. 3, 2021.

8. F. Zhang, X. Yang, and Y. Wang, “Intelligent Bidirectional Sign Language Translator: Sign Language Communication,” ResearchGate, 2023.

9. T. Tao, Y. Zhao, T. Liu and J. Zhu, "Sign Language Recognition: A Comprehensive Review of Traditional and Deep Learning Approaches, Datasets, and Challenges," in IEEE Access, vol. 12, pp. 75034-75060, 2024, doi: 10.1109/ACCESS.2024.3398806.