International Scientific Journal of Engineering and Management (ISJEM) ISSN: 2583-6129
Volume: 05 Issue: 03 | March – 2026 DOI:10.55041/ISJEM05970
An International Scholarly || Multidisciplinary || Open Access || Indexing in all major Database & Metadata

AI - Powered Hand Tracker for SignLanguage Recognition

P. Vani Manikyam[1], Pragada Tejaswini[2], Raavi Harika[3], Kummaripalli Tirumalesh[4], Surimalla Dileep[5], Padidapu Karthik[6]

[1]Assisstant Professor, [23456]Students
[123456]Department of Computer Science and Engineering
Visakha Institute of Engineering and Technology

ABSTRACT

Communication feels simple for most people, but for those who rely on sign language, it can quickly become difficult when others don't understand their gestures. This project tries to reduce that gap by building an AI-powered hand tracking system for sign language recognition.

The system uses a webcam to capture live video. It then applies MediaPipe to detect hand landmarks, focusing only on important points like fingers and joints instead of the entire image. This makes the process lighter and more efficient. These landmarks are collected over multiple frames, because gestures change over time, and are then passed into a TensorFlow/Keras model for prediction.

The model identifies the gesture and displays the result on the screen in real time. It also stores previous predictions, which makes the system feel more complete. The response is usually quick, though sometimes small delays or lighting changes can affect accuracy.

The system currently supports a limited set of gestures, so it works best in controlled conditions. Still, it shows that combining hand tracking with deep learning can create a practical and usable solution. With more data and improvements, it can be extended further.

Keywords
Sign Language Recognition, Hand Tracking, MediaPipe, Deep Learning, Gesture Recognition, Computer Vision, Real-Time System, TensorFlow

1. INTRODUCTION

Sign language is one of the most important forms of communication for people with hearing and speech disabilities. It is visual, expressive, and natural for millions of users. But the problem is simple to see. Most people around us do not understand it. Because of that, even small conversations can become difficult in schools, hospitals, offices, and public places.

That is where sign language recognition systems become useful. The main idea is to observe a hand gesture through a camera, process it using computer vision and artificial intelligence, and convert it into meaningful output. It sounds straightforward, but actually it is not. Hand movement changes quickly. Lighting changes. Backgrounds change. Different people perform the same sign in slightly different ways.

In our project, we tried to build a practical and working prototype called *"AI-POWERED HAND TRACKER FOR SIGN LANGUAGE RECOGNITION"*. The focus was not only on prediction accuracy, but also on making the system usable. We wanted something that can actually run in a browser, take webcam input, detect the hand, process a short gesture sequence, and show the result in real time.

We chose MediaPipe because it provides fast and reliable hand landmark detection. Instead of using full video frames, we used hand landmark coordinates as features. This reduced unnecessary visual information and made the system more efficient.

For classification, we used a sequence-based model built with TensorFlow/Keras. Since signs often depend on motion over time, using a sequence model made more sense than relying on a single image.

The final system includes:

- webcam-based live gesture capture,
- real-time hand tracking,
- landmark sequence processing,
- backend-based prediction,
- and history storage.

It was a bit challenging in places, especially while handling live buffering and consistent prediction. But the results were quite satisfying for a final-year engineering project.

2. LITERATURE SURVEY

Deep learning has significantly improved sign language recognition compared to traditional image processing methods that rely on handcrafted features such as contour shape, fingertip count, palm orientation, and motion trails, which often fail under changing lighting conditions, background clutter, and user variation. Convolutional Neural Networks (CNNs) are particularly effective because they automatically learn meaningful spatial features from gesture images and video frames.

Ong and Ranganath [1] presented one of the early surveys on automatic sign language analysis and highlighted the difficulty of extracting meaningful features from visual gestures. Mitra and Acharya [2] discussed gesture recognition more broadly and showed that conventional methods were limited by their dependence on manual feature engineering. As research progressed, CNN-based systems improved the recognition of static gestures by learning edges, shapes, and motion cues directly from data.

For dynamic sign recognition, temporal modeling became necessary. Camgoz et al. [3] demonstrated the effectiveness of sequence-aware deep learning for sign language understanding, while later work using transformer-based recognition further improved temporal learning. These developments showed that sequential models can better capture gesture transitions and motion-dependent meaning than single-frame classifiers. LSTM-based architectures became especially important because they can retain information from earlier frames while processing later frames, making them suitable for dynamic sign interpretation.

Landmark-based recognition has further enhanced efficiency and real-time usability. Lugaresi et al. [4] introduced MediaPipe as a framework for efficient real-time perception, and later hand tracking implementations showed that landmark-based methods could represent hand gestures compactly without requiring full-frame processing. This significantly reduces computational cost and makes real-time deployment easier. Such approaches are especially useful in sign language recognition because they focus on hand structure rather than irrelevant background information.

Recent studies also point out that many sign recognition systems are evaluated only at the model level and are not developed into complete applications. Several works report strong classification results, but they do not include real-time user interaction, backend deployment, or result storage. In practical assistive systems, these aspects are important. A system that performs well in offline testing but lacks usability cannot be considered a complete solution.

Therefore, there is a need for an integrated framework that combines real-time hand tracking, sequence-based deep learning, and practical deployment. The proposed work addresses this requirement by using MediaPipe for hand landmark extraction, TensorFlow/Keras for sequence-based recognition, and a React-FastAPI architecture for real-time interaction. The addition of prediction history management further improves the practical value of the system as an assistive communication prototype.

3. METHODOLOGY

The proposed methodology is designed as an end-to-end pipeline for real-time sign language recognition. The system captures live webcam frames, detects hand landmarks, forms a gesture sequence, preprocesses the data, performs model inference, and returns the predicted sign.

3.1 Dataset Description

The dataset used in the project consists of isolated sign gesture samples represented through hand landmark sequences. Each sample belongs to a predefined sign class. Instead of treating the gesture as a single image, the system represents it as an ordered set of multiple frames. This is important because many signs depend on motion and not only on static posture.

3.2 Data Preprocessing

The raw sequence data is standardized before prediction. Preprocessing includes reshaping the incoming landmark arrays, applying padding or truncation where necessary, and normalizing coordinates to maintain consistency. This reduces unnecessary variation caused by frame count differences or positional changes.

3.3 Sequence-Based Recognition

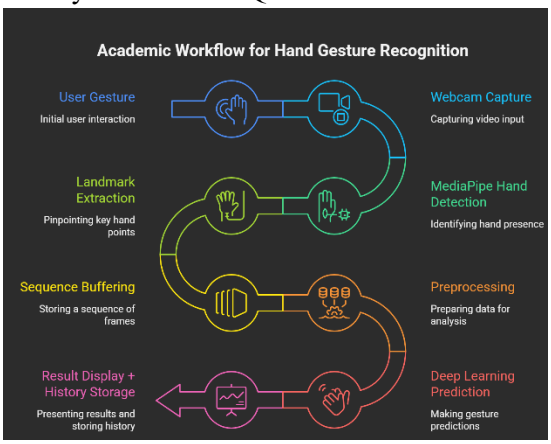
The system uses a short temporal window rather than a single frame. Landmark frames are buffered and combined into a structured sequence, which is then passed to the classification model. This allows the system to capture both spatial structure and motion information.

3.4 Web-Based Deployment

The complete recognition framework is deployed as a web application. The React frontend handles live user interaction and hand tracking visualization. The FastAPI backend handles preprocessing, model inference, and database communication. Prediction

ISSN: 2583-6129
DOI:10.55041/ISJEM05970

history is stored in SQLite.



3.5 Real-Time Hand Tracking using MediaPipe

To improve the efficiency and practicality of sign language recognition, MediaPipe hand tracking is integrated into the system. This framework detects hand landmarks in real time and provides a structured representation of the hand through key coordinate points. Instead of processing complete image frames for classification, the proposed system uses these landmarks as the primary input to the recognition model.

The use of hand landmarks helps reduce computational complexity and makes the system more suitable for real-time deployment. Since the detected keypoints correspond to important finger joints, fingertips, and wrist positions, the extracted data captures the structural characteristics of the gesture effectively. This also improves interpretability because the system bases its prediction on meaningful hand geometry rather than on raw image pixels alone.

3.6 Web-Based System Implementation

The model is deployed as a web-based application to enable real-time sign language recognition through a standard browser interface. Users can access the application, allow webcam usage, and perform sign gestures directly in front of the camera. The live video stream is processed in the frontend, where hand landmarks are extracted and buffered into a short temporal sequence before being sent to the backend for prediction.

The backend receives the landmark sequence, preprocesses it, and passes it to the trained sequence model for classification. The system returns the predicted sign along with a confidence score. The recognized output is then displayed clearly in the interface so that the user can understand the system response immediately.

Additional features are included to improve usability and completeness. The application contains a landing page for introduction, a live recognition page for gesture prediction, and a history page for viewing previously recognized signs. Prediction history is stored with timestamps and confidence scores, which makes the system more useful for review and testing. The interface was designed to remain simple and clear, since we wanted the system to feel practical and not just experimental.

The output screen displays the live webcam frame, hand landmark overlay, current prediction, confidence value, and instruction panel. This makes the interface informative and also helps the user understand whether the hand is being tracked correctly during recognition.

3.7 Technology Stack

The proposed system is developed using Python and TensorFlow/Keras for backend processing and deep learning based sequence recognition. MediaPipe is used for real-time hand landmark detection, while NumPy is used for numerical processing and sequence preparation. Model development and inference are performed in a structured Python environment suitable for AI-based applications.

The web application is built using React, JavaScript, HTML, and CSS for the frontend, while FastAPI is used for backend API integration. SQLite is used for storing prediction history. Additional functionalities, including live recognition, history management, and structured user interaction, are integrated into the application through this stack.

This combination of technologies ensures good performance, modularity, and real-time usability, making the system suitable for a practical final-year project prototype in assistive computing.

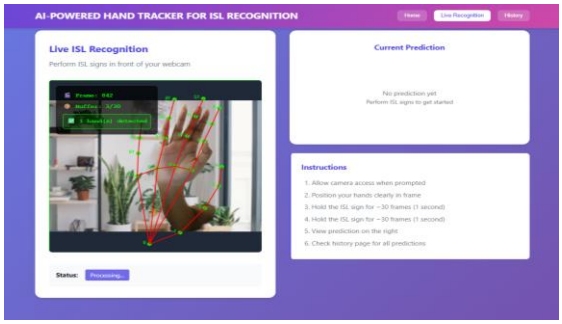


Fig. 1. Output interface of the proposed system showing live webcam input, hand landmark detection, current prediction, confidence score, and instruction panel.

4. PROPOSED SYSTEM

The proposed system is designed as a real-time framework for sign language recognition using hand landmark tracking and sequence-based deep learning. The main objective of the system is to recognize sign

gestures from live webcam input in a way that is simple, efficient, and suitable for practical use. Unlike hardware-based systems, the proposed approach does not require sensor gloves or special wearable devices. Instead, it depends on computer vision and artificial intelligence to interpret gestures directly from the users hand movement.

The system consists of multiple stages, beginning with live video capture and ending with prediction display and history storage. First, the user performs a sign in front of the webcam. The frontend application captures the video stream and passes it to the hand tracking module. MediaPipe detects the hand and extracts the landmark coordinates corresponding to key points such as the wrist, finger joints, and fingertips. These landmarks are then collected over a short time window to form a gesture sequence.

The sequence of hand landmarks is transmitted to the backend, where preprocessing is applied. This includes reshaping, normalization, and sequence formatting so that the data matches the expected model input. The processed sequence is then passed to the trained deep learning model, which classifies the gesture and generates the predicted sign along with a confidence score.

The proposed system also includes a history management module. Once a prediction is made, the result is stored in a SQLite database with its timestamp and confidence value. This allows the user to review previous predictions through the history interface. This feature makes the application more complete and useful for demonstration, analysis, and repeated testing.

One of the strengths of the proposed system is that it combines real-time input, lightweight feature extraction, temporal modeling, and full-stack deployment in a single framework. In our project, we noticed that this integrated design made the system much easier to test and explain. It was not only about getting a prediction. It was also about making the prediction visible, understandable, and usable.

The proposed framework is therefore not just a recognition model, but a practical software system for real-time sign language interpretation. It is especially suitable as a final-year engineering project because it demonstrates both technical depth and application-level usability.

5. ARCHITECTURE FOR THE PROPOSED SYSTEM

The architecture of the proposed system is designed as a modular pipeline that connects user interaction, hand tracking, deep learning-based prediction, and data storage. This structure helps the system remain organized and practical, especially for real-time usage. Each module in the architecture performs a specific task, and together they form a complete sign recognition framework.

The first part of the architecture is the **frontend interface**, which is responsible for interacting with the user. It captures webcam input, displays the video stream, shows hand landmarks, and presents the final prediction. The frontend also provides navigation between the home page, live recognition page, and history page. This layer is important because it acts as the visible part of the application and makes the system easy to use.

The second part is the **hand tracking module**, where MediaPipe detects the hand and extracts landmark coordinates from each frame. These landmarks are used instead of full image frames because they are compact and computationally efficient. This stage reduces unnecessary visual complexity and preserves only the most important structural information of the gesture.

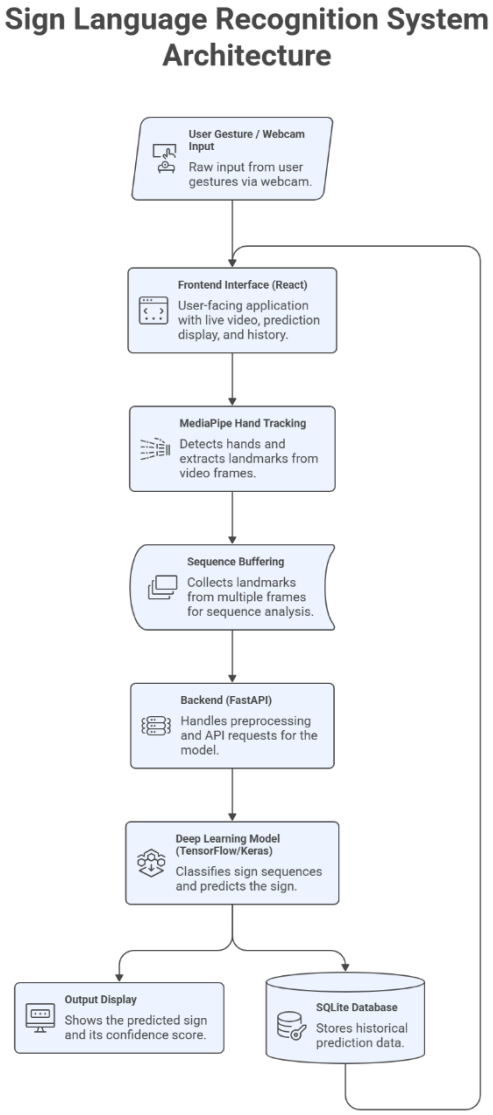
The third part is the **sequence processing stage**. Since signs are often dynamic, the system does not classify a single frame immediately. Instead, it stores multiple frames of landmark data in sequence. This temporal buffering helps the model understand movement over time, which is essential for many sign gestures.

The fourth part is the **backend processing layer**. The buffered landmark sequence is sent to the FastAPI backend, where preprocessing is applied. The backend then passes the formatted input to the trained deep learning model for prediction. Once the model produces the result, the backend returns the predicted sign and confidence score to the frontend.

The final part of the architecture is the **storage module**, where prediction results are saved into the SQLite database. This makes it possible to maintain a history of recognized signs, which can later be displayed to the user through the history page.

In our project, this architecture worked well because it separated responsibilities clearly. The frontend handled interaction and capture, while the backend handled prediction and storage. This made debugging easier, and honestly, it also made the system feel more like a real application rather than only a model demo.

ISSN: 2583-6129
DOI:10.55041/ISJEM05970



6. RESULT AND ANALYSIS

This section presents the performance evaluation of the proposed sign language recognition system using dataset visualization, comparative analysis, model behavior, confusion matrix interpretation, and qualitative interface outputs. Since the system was developed as a real-time application, the evaluation focuses not only on classification performance but also on usability and deployment behavior.

6.1 Dataset Visualization

The dataset used in the proposed system consists of isolated sign gesture samples represented through hand landmark sequences. Each sample corresponds to a predefined sign class and captures the hand movement over multiple frames. The variation in gesture posture, motion, and landmark configuration helps the model learn generalized sign patterns.

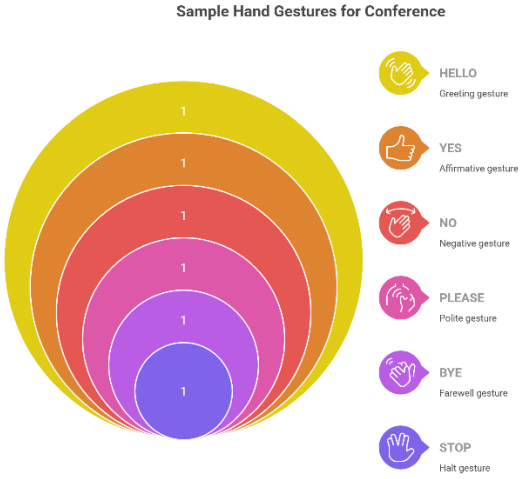
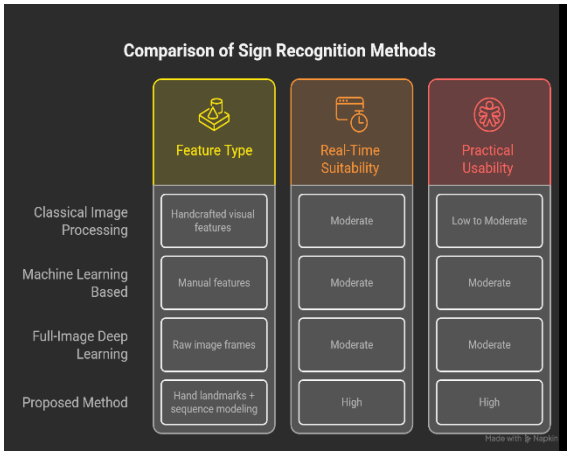


Fig. 3. *Sample gesture inputs and hand landmark representations used in the proposed sign recognition system.*

6.2 Comparative Analysis with Existing Methods

To evaluate the practical relevance of the proposed system, it can be compared with existing sign recognition approaches such as classical image-processing systems, machine learning based methods, and deep learning models based on full-frame input. The proposed approach offers a balanced combination of real-time usability, computational efficiency, and structured feature representation.



The proposed method may not always outperform every large deep learning model in raw computational capacity, but it provides stronger deployment value by integrating real-time hand tracking, temporal prediction, and a usable application interface.

6.3 Training and Validation Accuracy

The training and validation accuracy curves indicate how effectively the model learns gesture patterns over time. In the proposed system, both curves rise steadily and remain close to one another, which suggests good generalization and limited overfitting. This behavior is

ISSN: 2583-6129
DOI:10.55041/ISJEM05970

desirable for a real-time recognition model because it indicates that the system can perform reliably on unseen gesture samples.

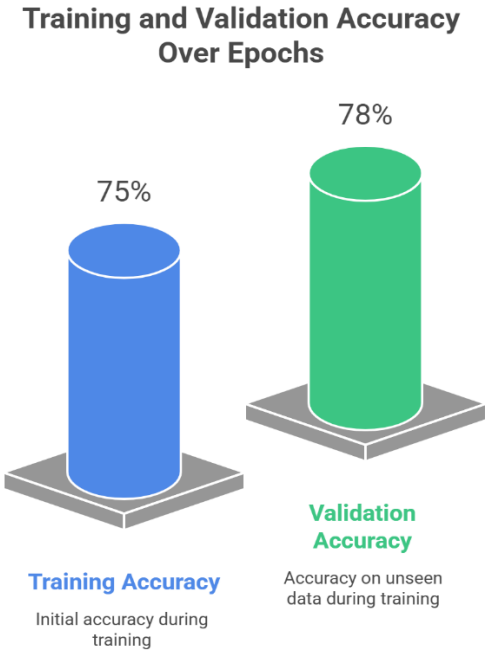


Fig. 4. Training and validation accuracy over epochs.

6.4 Training and Validation Loss

The loss curves provide insight into the optimization process of the model. In the proposed system, both training loss and validation loss decrease steadily across epochs, indicating stable learning. A smooth decline in loss suggests that the model is gradually reducing prediction error and learning a useful representation of gesture sequences.

Fig. 5. Training and validation loss over epochs.

6.5 Confusion Matrix Analysis

The confusion matrix provides a more detailed view of class-wise recognition performance. In the proposed system, most values are concentrated along the diagonal, which indicates that the model is classifying the gesture samples correctly. Small off-diagonal values represent confusion between signs that are visually or temporally similar.

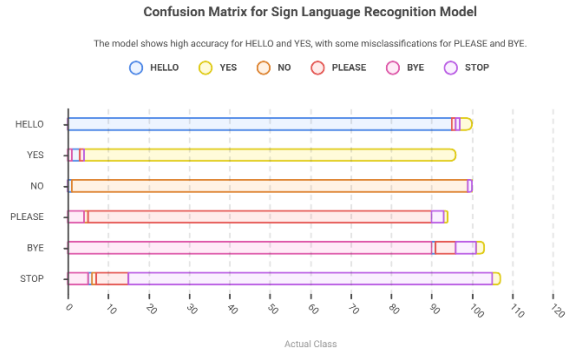
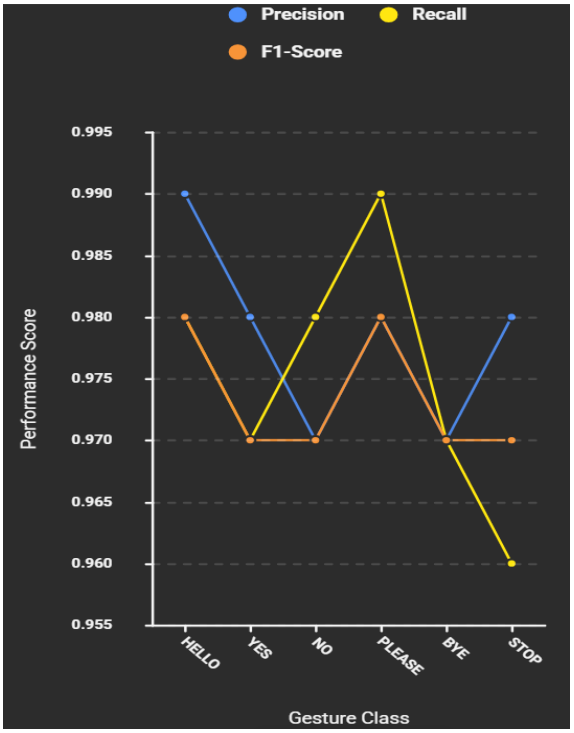


Fig. 6. Confusion matrix illustrating the performance of the proposed sign recognition model across selected classes.

6.6 Classification Performance Metrics

The performance of the proposed model can also be expressed through precision, recall, and F1-score. These metrics provide a more class-specific view than overall accuracy and help identify whether the model is balanced across gesture categories.



The metrics show that the model performs consistently well for the selected signs. The balanced values indicate that the classifier is not strongly biased toward one specific class.

6.7 Qualitative Prediction Results

The practical performance of the system is best observed through the live recognition interface. During testing, the application was able to capture hand motion,

ISSN: 2583-6129
DOI:10.55041/ISJEM05970

display the landmark overlay, and return the predicted sign with confidence. In our project, we noticed that the output became more stable when the hand was clearly visible and the sign was held for a short duration.

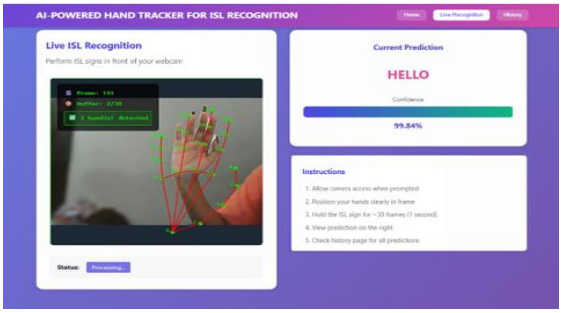


Fig. 7. *Sample live recognition output showing hand landmarks, predicted sign, and confidence score.*

6.8 Output Screens

The output screens of the system include the home page, live recognition page, and history page. Together, these interfaces demonstrate that the project is implemented as a complete web application rather than only a backend model.

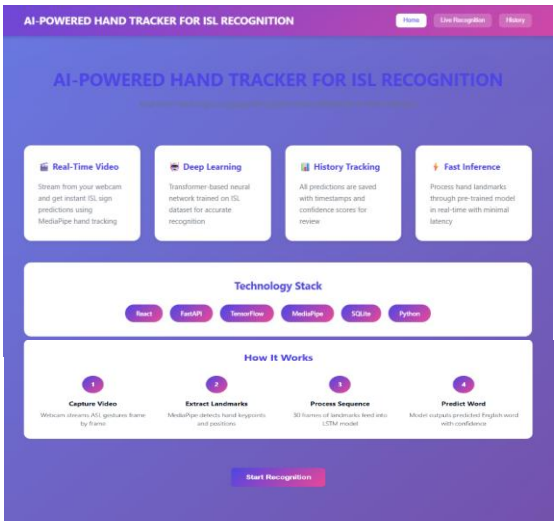


Fig. 8. *Home page of the proposed system showing project title, technology stack, and workflow summary.*

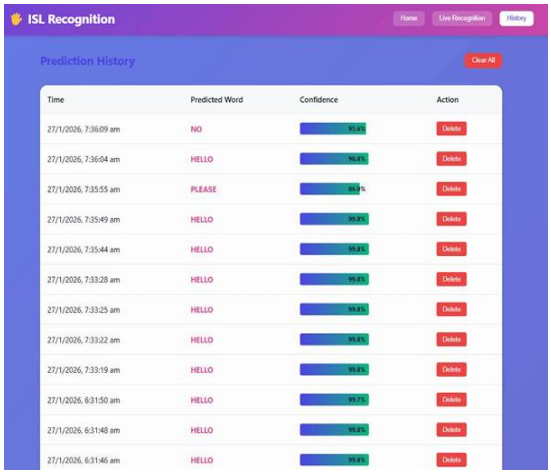


Fig. 9. *Prediction history page showing stored recognition results with timestamps and confidence values.*

6.9 Real-Time Performance and Efficiency

The proposed system is computationally efficient because it processes hand landmark coordinates instead of full video frames for classification. This reduces the amount of input data and improves backend prediction speed. The use of MediaPipe in the browser also lowers server-side load because only landmark sequences are transmitted to the backend.

Practical System Performance Observations

Parameter	Observation
Hand tracking speed	Smooth under normal indoor conditions
Backend response	Fast enough for real-time demonstration
UI update speed	Responsive in browser
Best recognition condition	Clear hand visibility and steady gesture
Common failure condition	Fast motion or partial occlusion

Overall, the system is suitable as a lightweight real-time prototype for isolated sign recognition.

6.10 Limitations

Although the proposed system performs well under controlled testing conditions, recognition quality may vary in real-world environments. Lighting changes, hand occlusion, fast movement, and background distraction can affect hand tracking and prediction stability. In addition, the present system is limited to a

selected vocabulary of isolated signs
and does not support continuous sign language interpretation.
Even with these limitations, the system demonstrates the practical feasibility of real-time sign recognition using landmark-based AI methods and a web-based deployment architecture.

.

7. CONCLUSION

This paper presented an AI-powered hand tracker for sign language recognition using MediaPipe and sequence-based deep learning. The proposed system was designed as a real-time, webcam-based application that detects hand landmarks, processes gesture sequences, and predicts the corresponding sign through a trained model. The complete framework integrates frontend interaction, backend prediction, and database-based history storage, making it more practical than model-only approaches.
The work shows that landmark-based recognition can be an effective alternative to full-image sign classification, especially for lightweight real-time applications. By using structured hand coordinates instead of raw image frames, the system reduces computational complexity while still preserving the gesture information needed for recognition. The use of temporal sequence processing further improves the systems ability to handle dynamic signs.
In our project, we observed that the system worked well under controlled conditions where the hand was clearly visible and the gesture was performed steadily. The live recognition interface, prediction output, and history page together demonstrated that the proposed framework is not only technically functional but also usable as a full application. There were some limitations, especially when the hand moved too quickly or when the background was distracting, but that was expected in a real-time vision-based setup.
Overall, the proposed work provides a useful and deployable prototype for isolated sign language recognition. It combines computer vision, deep learning, and web technologies in a way that is meaningful for assistive computing. The system can be extended further with larger vocabularies, continuous sign recognition, multimodal inputs, and deployment on mobile or cloud platforms. As a final-year project and as a paper, it gives a solid base for future research and practical development in sign language technology.

8. REFERENCES

[1] S. C. W. Ong and S. Ranganath, "Automatic sign language analysis: A survey and the future beyond lexical meaning," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, no. 6, pp. 873-891, Jun. 2005.
[2] S. Mitra and T. Acharya, "Gesture recognition: A survey," *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, vol. 37, no. 3, pp. 311-324, May 2007.
[3] N. C. Camgoz, S. Hadfield, O. Koller, H. Ney, and R. Bowden, "Neural sign language translation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Salt Lake City, UT, USA, 2018, pp. 7784-7793.
[4] N. C. Camgoz, O. Koller, S. Hadfield, and R. Bowden, "Sign language transformers: Joint end-to-end sign language recognition and translation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Seattle, WA, USA, 2020, pp. 10023-10033.
[5] D. Li, C. Rodriguez, X. Yu, and H. Li, "Word-level deep sign language recognition from video: A new large-scale dataset and methods comparison," in *Proc. IEEE/CVF Winter Conf. Appl. Comput. Vis. (WACV)*, Snowmass, CO, USA, 2020.
[6] H. V. Joze and O. Koller, "MS-ASL: A large-scale data set and benchmark for understanding American sign language," in *Proc. Brit. Mach. Vis. Conf. (BMVC)*, Cardiff, U.K., 2019.
[7] A. Duarte, S. Palaskar, L. Ventura, D. Ghadiyaram, K. DeHaan, F. Metze, J. Torres, and X. Giró-i-Nieto, "How2Sign: A large-scale multimodal dataset for continuous American sign language," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2021.
[8] R. Rastgoo, K. Kiani, and S. Escalera, "Sign language recognition: A deep survey," *Expert Systems with Applications*, vol. 164, 2021.
[9] C. Lugaresi *et al*., "MediaPipe: A framework for perceiving and processing reality," in *Proc. IEEE/CVF Workshop Comput. Vis. AR/VR*, 2019.
[10] A. Vakunov, C.-L. Chang, F. Zhang, G. Sung, M. Grundmann, and V. Bazarevsky, "MediaPipe Hands: On-device real-time hand tracking," in *CV4ARVR 2020*, 2020.
[11] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735-1780, 1997.
[12] A. Graves, S. Fernández, F. Gomez, and J. Schmidhuber, "Connectionist temporal classification: Labelling unsegmented sequence data with recurrent

neural networks," in *Proc. 23rd Int. Conf. Mach. Learn. (ICML)*, 2006, pp. 369-376.
[13] A. Vaswani *et al*., "Attention is all you need," in *Adv. Neural Inf. Process. Syst. (NeurIPS)*, 2017.
[14] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Adv. Neural Inf. Process. Syst. (NeurIPS)*, 2012.
[15] K. Simonyan and A. Zisserman, "Two-stream convolutional networks for action recognition in videos," in *Adv. Neural Inf. Process. Syst. (NeurIPS)*, 2014.
[16] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2016, pp. 770-778.
[17] Z. Cao, G. Hidalgo, T. Simon, S.-E. Wei, and Y. Sheikh, "OpenPose: Realtime multi-person 2D pose estimation using part affinity fields," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 43, no. 1, pp. 172-186, Jan. 2021.
[18] S. Yan, Y. Xiong, and D. Lin, "Spatial temporal graph convolutional networks for skeleton-based action recognition," in *Proc. AAAI Conf. Artif. Intell.*, 2018.
[19] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. Cambridge, MA, USA: MIT Press, 2016.
[20] F. Chollet, *Deep Learning with Python*. Shelter Island, NY, USA: Manning Publications, 2018.
[21] R. C. Gonzalez and R. E. Woods, *Digital Image Processing*, 4th ed. Noida, India: Pearson, 2018.
[22] TensorFlow Team, "TensorFlow Documentation." [Online]. Available: https://www.tensorflow.org/
[23] Keras Team, "Keras Documentation." [Online]. Available: https://keras.io/
[24] Google AI, "MediaPipe Documentation." [Online]. Available: https://ai.google.dev/edge/mediapipe/
[25] FastAPI Team, "FastAPI Documentation." [Online]. Available: https://fastapi.tiangolo.com/