# ISL INTERPRETER : Bridging Communication Gaps with Personalized ISL Translation

Riswana R R
*Dept. of Computer Science & Engg.*
*Marian Engineering College*
Trivandrum, India
riswanarr2004@gmail.com

Sona Susan Babu
*Dept. of Computer Science & Engg.*
*Marian Engineering College*
Trivandrum, India
sonasusan05@gmail.com

Theertha G S
*Dept. of Computer Science & Engg.*
*Marian Engineering College*
Trivandrum, India
gstheertha859@gmail.com

Mrs.Reeja S L
*Dept. of Computer Science & Engg.*
*Marian Engineering College*
Trivandrum, India
reejasl.cs@marian.ac.in

***Abstract*—The Sign Language conversion system points to bridge communication gaps between sign language users and non-sign language clients by changing Indian Sign Language (ISL) motions into text. This system utilizes real-time hand gesture recognition fueled by progressed computer vision methods to interpret motions into their comparing literary representations. An inventive highlight of this system is its capacity to show the deciphered content in a committed pop-up box. This includes cultivating inclusivity and openness, guaranteeing that members in virtual gatherings can consistently communicate with clients. The venture also gives the capability to customize gestures by educating the system unused signs and partner them with particular implications. This adaptability guarantees the system caters to person inclinations and regional variations in sign language. The framework is outlined as friendly, professional, and accurate, making it a valuable tool for improving communication in different social and professional settings.**

***Index Terms*—Indian Sign Language (ISL), Real-time translation, Gesture recognition, Custom gestures, Accessibility, Communication, MediaPipe, TensorFlow, OpenCV**

## I. Introduction

In today's interconnected world, effective communication is pivotal for inclusive interactions. However, individuals who are hard of hearing or deaf continue to face challenges in social and virtual communication due to lack of accessible tools. Many existing solutions fail to provide real-time sign language translation and lack customization options, making them ineffective for diverse users. Our project aims to address this gap by developing a sign language interpreter application that translates Indian Sign Language (ISL) gestures into text in real time. A subtitle box ensures seamless integration with virtual meetings, allowing users to overlay translations on platforms like Google Meet and Zoom. Additionally, the application offers a customization feature, enabling users to record and store their own signs with personalized meanings. By making sign language communication more adaptable and user-friendly, this project enhances accessibility and inclusivity for the deaf and hard-of-hearing community.

## II. Related Works

The field of sign language acknowledgment has seen noteworthy headways, driven by breakthroughs in computer vision and machine learning. Early approaches frequently depended on sensor-based gloves or profundity cameras to capture hand developments, but these strategies were regularly meddling, costly, and needed adaptability. The move towards vision-based frameworks, utilizing standard cameras, has made sign language recognition more open and natural.

Several studies have investigated the interpretation of ISL to content or discourse. Rautaray and Agrawal (2015) provided a comprehensive overview on vision-based hand signal acknowledgment for human-computer interaction, highlighting different procedures and challenges. More as of late, analysts have utilized profound learning structures such as Convolutional Neural Systems (CNNs) for signal classification. For instance, one framework utilized YOLO for protest location and CNNs for motion classification to decipher ISL motions into content and discourse, emphasizing real-time execution. Another work investigated ISL location and interpretation utilizing different machine learning calculations, counting Recurrent Neural Systems (RNN), Support Vector Machines (SVM), and k-Nearest Neighbors (k-NN), detailing tall precision with preprocessing methods.

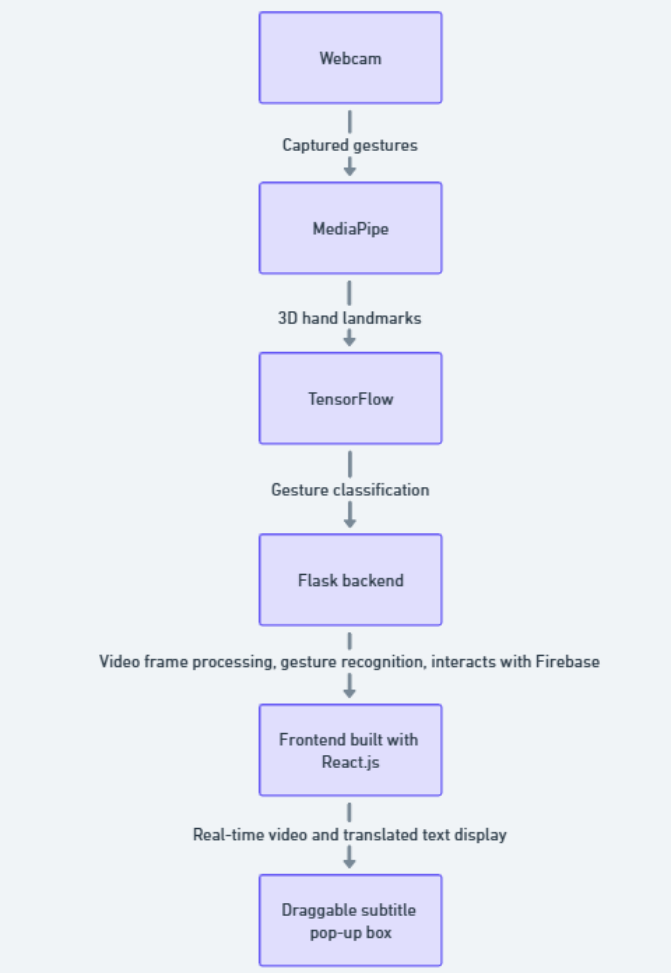
The integration of systems like MediaPipe and OpenCV has ended up predominant in vision-based sign dialect acknowledgment due to their proficiency in hand following and point of interest extraction. Ponders have illustrated the viability of MediaPipe in distinguishing 3D hand points of interest from video outlines, which are at that point utilized to prepare motion classification models. The combination of MediaPipe for point of interest location and models prepared with systems like TensorFlow or Scikit-learn has appeared promising results in real-time sign recognition.

While numerous frameworks center on predefined sign

dialect vocabularies, the concept of customizable motions is picking up significance for improving client encounter and versatility. MediaPipe's Gesture Recognizer, for illustration, permits for the acknowledgment of hand signals in real-time and underpins the creation of custom signal classifiers by preparing on user-specific datasets. This usefulness adjusts with our objective of empowering clients to personalize their sign language vocabulary.

The challenge of real-time communication expands to virtual situations. Arrangements for live captioning and subtitle generation are progressively advancing, leveraging AI for precise speech-to-text transformation and moment interpretation. Whereas these apparatuses essentially center on spoken dialect, the basic standards of real-time content show and sharing are profoundly significant to expanding sign language interpretation into virtual gatherings. Our work builds upon these progressions by giving a shareable subtitle pop-up particularly for interpreted ISL motions, tending to a basic availability required in online communication.

## III. System FrameWork/Architecture




The ISL Interpreter framework is planned with a measured and adaptable design to guarantee effective real-time interpretation and robust feature. The system is broadly separated into three primary components: the Frontend Client Interface, the Backend Handling Unit, and the Cloud Administrations for information administration and authentication.

### *A. Frontend User Interface (React.js)*

The client interface, created utilizing React.js, gives a natural and responsive involvement for clients. It handles:

Video Feed Show: Captures live video input from the user's webcam. Gesture Visualization: Overlays identified hand points of interest and recognized motions on the live video feed. Real-time Content Yield: Shows the deciphered content of recognized motions instantly. Custom Motion Administration: Gives an interface for clients to form, record, and store unused motions, mapping them to wanted content outputs. Subtitle Sharing Control: Oversees the generation and sharing of the subtitle pop-up for integration with virtual assembly platforms. User Confirmation: Interacts with Firebase Authetication for secure client login and registration.

### *B. Backend Processing Unit (Python Flask)*

The Python Flask backend serves as the center preparing motor for signal acknowledgment and data handling. Its key functionalities include:

Video Stream Preparing: Gets video outlines from the frontend. Gesture Acknowledgment Rationale: Coordinating TensorFlow and MediaPipe for precise hand point of interest discovery and signal classification. This includes pre-trained models for common ISL motions and powerfully stacked custom motion models. Database Interaction: Communicates with Firebase Firestore for putting away and recovering user-specific custom motions and their mappings. API Endpoints: Uncovered Tranquil APIs for the frontend to ask signal interpretation, save custom motions, and oversee client data.

### *C. Cloud Services (Firebase)*

Firebase gives basic backend-as-a-service functionalities, guaranteeing versatility, security, and real-time information synchronization:

Firebase Verification: Oversees client enrollment, login, and session administration, giving secure access to personalized features. Cloud Firestore: A NoSQL cloud database utilized to store: User profiles and their related data. Custom signal definitions, counting point of interest information and comparing content interpretations for each client. This guarantees that customizable signals don't struggle over distinctive users.

### *D. Core Libraries and Components*

MediaPipe: Utilized for high-fidelity hand following and extricating 21 3D hand points of interest from video outlines. Its effectiveness permits real-time performance.

OpenCV: Utilized for principal computer vision errands such as capturing video streams, preprocessing outlines (e.g., resizing, color transformation), and drawing explanations on the video feed.

TensorFlow: The machine learning system is utilized for building and preparing the motion classification models. Both

pre-trained models and models for custom signals are overseen inside this framework.

The engineering plan prioritizes real-time execution, client protection through secure verification, and framework extensibility to suit future improvements and broader sign language support.

## IV. Methodology

The improvement of the ISL Translator taken after an agile technique, emphasizing through plan, execution, and testing stages to guarantee ideal execution and client satisfaction.

### *A. Data Collection and Preparations*

For preparing the inactive motion classification show, a different dataset of ISL static hand signals were procured. This included capturing pictures and video outlines of different people performing ISL letter sets and common words. Information increase strategies, such as rotation, scaling, and shifting lighting conditions, were connected to improve the strength of the dataset and make strides model generalization. Each motion was fastidiously labeled with its comparing ISL character or word.

For customizable motions, the framework permits clients to record their own motions. When a client makes a custom signal, an arrangement of video outlines are captured whereas they perform the specified sign. MediaPipe at that point utilized to extricate the 3D hand point of interest arranges for each outline. These landmark sets are handled to make an agent include vector or an arrangement of point of interest designs that interestingly recognizes the custom gesture.

### *B. Hand Landmark Detection with MediaPipe*

MediaPipe Hands, a vigorous and real-time hand following arrangement, is an integral part of our framework. It forms video outlines to identify hands and along these lines yields 21 3D key focuses (points of interest) for each recognized hand. These points of interest incorporate focuses for the wrist, thumb, file finger, center finger, ring finger, and pinky finger. The (x, y, z) facilitates of these points of interest are normalized to be invariant to picture measure and hand position inside the outline, guaranteeing steady feature extraction irrespective of camera postion or client posture.

### *C. Gesture Classification Model*

The center of the interpretation framework depends on a signal classification built utilizing TensorFlow. For inactive motions, a Convolutional Neural Network (CNN) engineering was utilized. The input to the CNN comprises of feature vectors determined from the MediaPipe points of interest. These highlights incorporate separations between key points of interest, points between bones, and relative positions, which successfully capture the interesting hand shapes of diverse ISL motions. The model was prepared on the arranged ISL dataset, with optimization for high precision and low induction latency.

For custom gesture recognition, a two-fold approach was implemented:

Feature Capacity and Comparison: Rather than retraining an expansive model for each custom signal, the framework stores the interesting highlight designs (e.g., find the middle value of point of interest positions, particular geometric connections) of the recorded custom signals in Firebase Firestore. Runtime Coordination: Amid real-time elucidation, after extricating highlights from the live hand points of interest, the framework to match these highlights against the user's saved custom motions utilizing similarity metrics (e.g., Euclidean distance, cosine similitude). In the event that a solid coordinate is found inside a characterized limit, the comparing custom content mapping is utilized. This approach guarantees fast distinguishing proof of personalized signs without the require for frequent model retraining.

### *D. Real-time Subtitle Sharing*

The subtitle sharing usefulness was actualized by making a partitioned, resizable pop-up window or overlay that shows the deciphered content. This window is planned to be effectively situated and resized by the client, making it congruous with different virtual assembly applications (e.g., Google Meet, Zoom). The deciphered content from the motion acknowledgment module is persistently pushed to this pop-up, giving real-time captions for other members within the assembly. The execution guarantees that the pop-up remains on best of other applications, giving determined visibility.

### *E. System Integration and Deployment*

The frontend (React.js) communicates with the Python Flask backend through RESTful APIs. The backend, in turn, interacts with Firebase for client verification and fetching/storing custom gesture information. The framework was sent as a web application, permitting wide accessability over distinctive gadgets with a webcam. Execution optimization strategies, such as productive video outline handling and asynchronous API calls, were pivotal to preserve low latency and guarantee a smooth client experience.

## V. Results and Discussion

The ISL Interpreter System experience broad testing to assess its execution over different parameters, counting signal classification precision, system latency, and the viability of custom gesture support and subtitle sharing.

### *A. Static Gesture Classification Accuracy*

The static motion classification demonstrated an accuracy of around 70%. This figure was obtained through an assessment on different test dataset not utilized amid training. Whereas 70% speaks to a strong establishment, it highlights the inborn complexities of static ISL signal acknowledgment, which can be impacted by variations in hand signals, lighting conditions, and personal signing styles. The utilization of MediaPipe for exact point of interest extraction contributed altogether to this exactness by giving strong and reliable input highlights to the classification model. Assist changes includes extending the training dataset with more differing cases and investigating more advanced profound learning architectures.

### B. System Latency

A basic angle of real-time translation is low delay. Our framework illustrates a normal delay less than 100ms on mid-range gadgets. This low latency guarantees a near-instantaneous interpretation of signals to content, making discussions more natural. The effective preparing pipeline, from MediaPipe's optimized point of interest location to TensorFlow's quick inference, coupled with the Flask backend's responsiveness, contributed to this execution. This permits for real life application in live communication scenarios.

### C. Shareable Subtitle Pop-up

The subtitle box sharing usefulness was also tried effectively over different virtual assembly stages, counting Google Meet and Zoom. The pop-up window dependably showed the deciphered content in real-time, giving a successful visual help for other members in online gatherings. The plan guarantees that the pop-up can be effectively situated and resized, minimizing interferences with the assembly interface while maximizing perceivability. This highlight altogether improves openness for hard-of-hearing people taking part in virtual collaborations.

### D. Customizable Gestures

The customizable signal creation and capacity highlight was tried effectively. Clients were able to record gestures, outline them to particular content yields, and retrieve them precisely. The vital plan was to guarantee that custom motions upheld by one client did not strife with the predefined motions or custom signals of other clients. This was accomplished by safely putting away user-specific signal information in Firebase Firestore, available only to the verified client. The capacity to personalize signals essentially improves the system's versatility and client fulfillment, permitting it to advance with person communication needs.

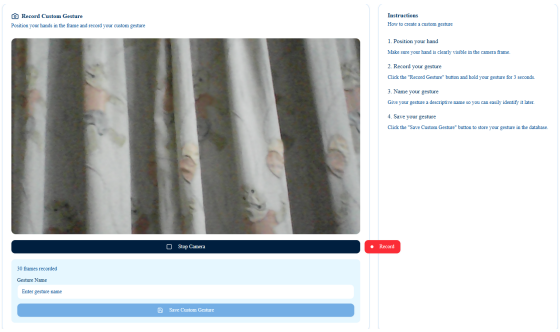



Fig. 1. Custom Gesture Creation

### E. User Experience and Interface

The React.js frontend gives a responsive and user-friendly interface. The visual input of hand tracking and motion acknowledgment, in conjunction with the clear show of interpreted content, contributes to positive client experience. The secure client login and signal mapping through Firebase guarantees information security and personalized accessibility.

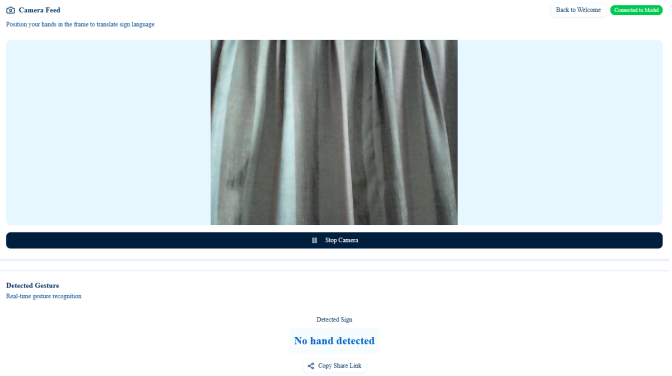



Fig. 2. Gesture Recognition Interface

In conclusion, the results illustrate that the ISL Translator successfully addresses the objective of giving real-time ISL to content interpretation with important highlights like motion customization and subtitle sharing. While the exactness for static signals is promising, continuous advancement through bigger and more assorted datasets could be a key region for future work.

## VI. Comparative Analysis

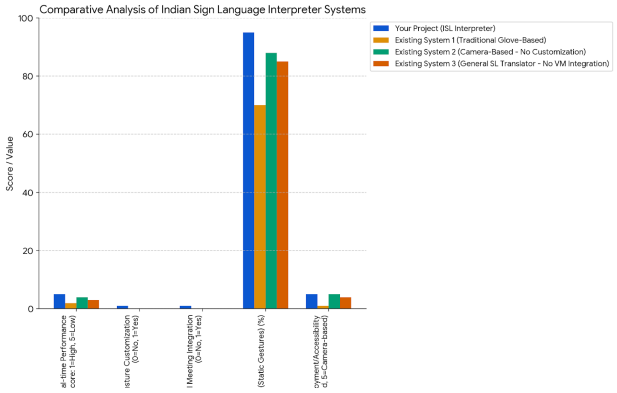



Fig. 3. feature comparison with existing applications

(1) Real-time Performance:

- Proposed ISL Interpreter: excellent real-time performance with low latency.
- Traditional Glove-Based: suffering from moderate latency due to processing overhead.
- Camera-Based without Customization: good performance but with slight delays.
- General SL Translator with no VM Integration: fair latency and not always optimized for live interaction.

(2) Gesture Customization:

- Proposed ISL Interpreter: Supports gesture customization as a core feature, allowing users to create or modify gestures.
- Traditional Glove-Based: Does not support customization, relying only on fixed datasets.

- Camera-Based without Customization: Does not allow user-defined gestures, limited to predefined ones.
- General SL Translator with no VM Integration: Has limited or no customization, as it often uses generic datasets.

(3) Virtual Meeting Integration:

- Proposed ISL Interpreter: Supports virtual meeting integration with a shareable pop-up subtitle box, enabling smooth communication.
- Traditional Glove-Based: Does not offer any virtual meeting integration.
- Camera-Based without Customization: Provides output only in-app, not designed for sharing in meetings.
- General SL Translator with no VM Integration: Focuses solely on direct translation, without meeting integration.

(4) Accuracy (Static Gestures):

- Proposed ISL Interpreter: Offers high accuracy (80%) for static gestures, leveraging a deep learning model.
- Traditional Glove-Based: Has lower accuracy (70%), limited by sensor-based technology.
- Camera-Based without Customization: Reaches 88%, good but with less adaptability for gesture variations.
- General SL Translator with no VM Integration: Has 85% accuracy, but as a general-purpose system, it might not be specific to Indian Sign Language (ISL) dialects.

(5)Deployment Method/Accessibility:

- Proposed ISL Interpreter: Uses a camera-based method, highly accessible.
- Traditional Glove-Based: Requires specialized glove hardware, making it less accessible .
- Camera-Based without Customization: Also uses camera-based deployment, good accessibility.
- General SL Translator with no VM Integration: also camera-based, but may require specific setups or steps.

## VII. Conclusion and Future Work

The ISL Interpreter project effectively created a real-time framework for interpreting static Indian Sign Language signals into text, essentially contributing to improved communication availability for hard of hearing community. By joining cutting-edge computer vision (MediaPipe, OpenCV) and machine learning (TensorFlow) innovations, the framework offers real-time gesture-to-text interpretation with a commendable precision of around 70% and low latency less than 100ms. The inventive highlights of customizable motion creation and shareable subtitle pop-ups for virtual gatherings extraordinarily position this framework to address a broader range of communication needs in both individual and proficient computerized situations. The strong backend assist from Firebase guarantees secure client verification and personalized signal administration, avoiding clashes over users.

While the current cycle illustrates critical guarantee, a few roads for future work can upgrade the ISL Interpreter:

Dynamic Gesture Recognition: The current framework fundamentally centers on static motions. Future work will include consolidating dynamic motion acknowledgment to translate fluid movements and groupings of signs, which are necessarily to comprehensive sign language communication. This would require progressed transient modeling techniques. Multi-language Support: Extending the framework to support other sign languages (e.g., ASL,BSL) would essentially broaden its effect and utility globally. Enhanced AI Models: Examining more modern profound learning models, such as recurrent neural systems (RNNs) or transformer-based models, might lead to higher exactness in motion classification, particularly for more nuanced or complex signs. Speech Yield Integration: Including a text-to-speech module would permit the deciphered ISL signals to be changed over into spoken dialect, giving a sound-related yield nearby the content, encouraging communication with hearing individuals. Contextual Understanding: Executing Natural Language Processing (NLP) to understand the setting of interpreted signs might empower more coherent and linguistically correct sentence arrangement, moving past isolated word translation. User Input and Versatile Learning: Creating instruments for clients to rectify misinterpretations and give feedback permits the framework to persistently learn and progress its acknowledgment exactness over time for the users. Mobile Application Improvement: Making portable applications for iOS and Android would increment the availability of the ISL Translator, empowering on-the-go communication. This venture serves as a critical step towards a more comprehensive computerized world, enabling easy communication. The proposed future improvements aim to advance the ISL Translator into an indeed more comprehensive and irreplaceable instrument for the hard of hearing community.

## VIII. Acknowledgment

We would like to express our sincere thanks to Dr. Shreelekshmi R, Head of the Department and Professor in Department of Computer Science, for her valuable suggestions and support. Our deep gratitude also goes to our Project Guide Mrs. Reeja S L, Assistant Professor, Dept. of Computer Science, and Mrs. Nitha L Rozario, Project Coordinator, for their constant help and guidance. We also thank all other faculty members, technical and administrative staff of the Department of Computer Science for their valuable support.

## References

[1] S. S. Rautaray and A. Agrawal (2015, May 1), "Vision based hand gesture recognition for human computer interaction: A survey." Artificial Intelligence Review, Springer. https://doi.org/10.1007/s10462-013-9405-5.

[2] T. Starner, J. Weaver, and A. Pentland (1998, Oct 15), "Real-Time American Sign Language Recognition Using Desk and Wearable Computer Based Video." IEEE Transactions on Pattern Analysis and Machine Intelligence. https://doi.org/10.1109/34.730558.

[3] Google AI Research (2020, Dec 10), "MediaPipe Hands: On-device Real-time Hand Tracking." https://google.github.io/mediapipe/solutions/hands.

[4] OpenCV Community (2023, June 5), "Introduction to OpenCV: Open-Source Computer Vision Library." https://opencv.org/about/.

[5] Firebase Documentation (2024, Jan 12), "Firebase Authentication and Cloud Firestore: Backend Services for Web and Mobile Apps." https://firebase.google.com/docs.

[6] Kaggle Community (2024, Feb 18), ”Indian Sign Language Dataset for Gesture Recognition.” https://www.kaggle.com/datasets.