# Real-Time Sign Language Translator App

Bhavesh Rajendra Desale[1], Nayan Nitin Bedse[2], Tejas Nandkumar Deore[3], Darshan Umesh Shitole[4], Ms. G. B. Patil[5]

*Computer Engineering Department, R. C. Patel Institute of Technology Shirpur, India*

***Abstract*—The Real-Time Sign Language Translator App is a mobile tool designed to bridge the communication gap between sign language users and those who don't know sign language. The app captures hand movements using the smartphone camera and translates them into spoken language or on-screen text. This feature makes communication smooth and effective. It supports various sign languages, including American Sign Language (ASL), British Sign Language (BSL), and Indian Sign Language (ISL), which helps it reach a broader audience.**

**The app benefits not only deaf individuals and those with speech challenges but also offers personalized communication support in social, educational, and professional situations. Additionally, it includes features for speech-to-sign translation and customizable gesture mapping, enhancing the user experience and providing personal flexibility.**

**Built with modern technologies, the app is developed in Kotlin for Android, uses MediaPipe for real-time hand tracking, and employs TensorFlow Lite for gesture recognition optimized for mobile devices. Firebase supports user authentication and real-time data synchronization, ensuring secure and seamless communication, with a focus on accessibility.**

**The app proves increasingly useful in healthcare, education, and everyday communication. It enables real-time, two-way interactions between people with different abilities and the general public while promoting inclusion by helping users overcome traditional communication barriers.**

***Index Terms*—Sign Language Translation, Real-Time Gesture Recognition, Mobile Accessibility, Computer Vision, Hand Gesture Detection, MediaPipe, TensorFlow Lite, Android Application, Kotlin Programming, Speech-to-Text Translation, Text-to-Sign Conversion, Assistive Technology, Inclusive Communication, Deaf and Mute Support, Human–Computer Interaction**

## I. INTRODUCTION

Communication is vital for human interactions. It allows people to share ideas, feelings, and intentions in various settings, such as social, educational, and work environments. Unfortunately, individuals with hearing or speech impairments often struggle to communicate effectively. Global health studies show that millions use sign language as their primary means of communication. However, many people do not understand or interpret sign gestures. This lack of understanding creates significant barriers in daily interactions, especially face-to-face, where interpreters or alternative communication tools may not be available. As a result, individuals with hearing or speech disabilities often have restricted access to education, healthcare, job opportunities, and social inclusion. Traditional methods of communication, like sign language interpreters, written notes, or lip reading, offer limited support and can be impractical in real-time situations. Interpreters are not always available, written communication can be slow and inefficient, and lip reading requires extensive training. This can still lead to misunderstandings. These challenges highlight the need for a scalable, immediate, technology-based solution that connects sign language users and non-signers without needing outside help. As mobile computing and artificial intelligence improve, smartphones have become powerful tools for supporting real-time, vision-based interaction systems.

Recent advancements in computer vision and machine learning allow for the recognition of human gestures through camera inputs. Frameworks like MediaPipe provide efficient real-time hand tracking. Meanwhile, lightweight deep learning models optimized for mobile devices, such as TensorFlow Lite, enable quick gesture classification with low computational demands. These technologies create new opportunities for assistive applications that can instantly interpret sign language gestures and convert them into readable text or spoken words. Despite these advancements, many existing solutions only support a single sign language, are not responsive in real-time, or require specialized hardware. This limits their usability and adoption.

To address these issues, the Real-Time Sign Language Translator App offers an inclusive and accessible mobile solution for smooth communication between sign language users and the general public. The app is developed for the Android platform using Kotlin, ensuring a strong, responsive, and user-friendly interface. By using MediaPipe for real- time hand detection and TensorFlow Lite for effective gesture recognition, the system translates sign language gestures into text and speech with minimal delay. The app supports multiple sign languages, including American Sign Language (ASL),

British Sign Language (BSL), and Indian Sign Language (ISL), making it useful across different linguistic and cultural groups.
Beyond basic translation features, the platform includes several user-friendly and educational tools. These features include real-time video translation, offline gesture recognition for continuous use, role-based authentication, and a learning mode that helps users practice and improve their sign language skills. Firebase provides secure user authentication, cloud data storage, and real-time synchronization of user preferences, ensuring scalability, reliability, and personalization. The design emphasizes accessibility and intuitive interaction to accommodate users with varying levels of technical experience.

The combination of AI-driven gesture recognition and mobile application development makes the Real-Time Sign Language Translator App a complete assistive communication system. By enabling two-way, real-time communication and supporting multiple sign languages, the application addresses practical communication challenges while promoting social inclusion and independence for individuals with hearing or speech impairments. This project contributes to the growing field of assistive technologies by showing how mobile plat- forms can effectively break down long-standing communication barriers and empower differently-abled individuals to engage confidently in society.

## II. LITERATURE REVIEW

Sign language recognition has become a key area of re- search because of the growing focus on accessibility and the rapid growth of computer vision and mobile computing technologies. Early research mainly used traditional image- processing techniques, such as background subtraction, edge detection, contour analysis, and skin-color segmentation to identify hand gestures. Studies by Patel et al. [1] showed that these methods were lightweight and suitable for early systems; however, they were very sensitive to factors like lighting changes, camera quality, and background complexity. Consequently, their performance dropped significantly in real- world situations, which limited their practical use.

To address these issues, researchers started to look into machine learning methods, especially convolutional neural networks (CNNs), for sign language recognition. Zhang et al.
[2] found that CNN models greatly improved classification accuracy for static hand gestures by automatically learning important features from image data. Their work proved that deep learning could be effective for gesture recognition and showed it was better than using manually extracted features. Still, these models often demanded high computational power, making them less ideal for use on mobile and embedded devices.

Later research aimed at optimizing deep learning models for real-time and mobile settings. Kumar and Verma [3] explored lightweight CNN designs specifically for gesture recognition. They demonstrated that model compression and parameter optimization could cut down on processing time without affecting accuracy. Despite these advancements, many systems remained limited to offline processing or depended heavily on cloud computing. This reliance led to delays, privacy issues, and the need for constant internet access.

A significant step forward in real-time gesture recognition came with the introduction of holistic hand-tracking systems. MediaPipe Hands, developed by Google, allowed for accurate real-time detection of 3D hand landmarks using just one RGB camera. Research by Liu et al. [4] confirmed that track-ing based on landmarks greatly improved reliability against changes in lighting, background noise, and hand position. By focusing on skeletal landmarks instead of raw pixel data, these systems achieved better consistency and lower computational demands, making them ideal for mobile use.

At the same time, TensorFlow Lite became a useful frame- work for running deep learning models directly on edge devices. Research by Sharma et al. [5] showed that on-device inference with TensorFlow Lite delivered gesture recognition with low latency while keeping user privacy intact by avoiding cloud processing. Their findings highlighted that real-time mobile gesture recognition systems need to strike a balance between accuracy, speed, and energy efficiency to ensure ongoing usability.

To fill these gaps, the proposed Real-Time Sign Language Translator App combines MediaPipe for reliable real-time hand tracking, TensorFlow Lite for effective on-device gesture classification, and Firebase for secure authentication and real-time data synchronization. Unlike previous systems, this solution supports multiple sign languages, allows for real-time two-way communication, and offers offline functionality. This makes it a comprehensive, scalable, and inclusive communication tool. This approach builds on existing research and pushes it forward by focusing on real-world usability and accessibility across various social and technological contexts.

## III. SYSTEM ARCHITECTURE AND METHODOLOGY

The proposed Real-Time Sign Language Translator App aims to offer an efficient, scalable, and accessible way for sign language users to communicate with non-signers in real time. The system uses mobile-based computer vision, on-device machine learning, and cloud services to ensure fast translation, secure data handling, and a smooth user experience on Android devices. The design focuses on offline use for essential translation features while also utilizing cloud services to improve personalization, synchronization, and long-term growth.

The overall system uses a hybrid model that merges on- device processing with cloud data management. This setup provides quick response times for gesture recognition and translation, while also allowing flexibility for user authentication, preference storage, and future system upgrades. The entire architecture and method are organized into four main functional layers that together support real-time sign language translation and two-way communication.

### *A. Input Acquisition and Data Capture*

The input acquisition component captures both visual and audio data needed for gesture recognition and speech interaction. The smartphone camera constantly records hand movements at high frame rates. This allows for accurate detection of both static and dynamic sign gestures. MediaPipe Hands is used to provide real-time hand tracking and extract 21 three-dimensional landmark points for each hand. This setup ensures stable detection, even in changing lighting and complex backgrounds.

Along with visual input, an audio module picks up spoken language through the device microphone. This enables speech- to-text processing. The dual-input system allows smooth two- way communication between deaf and hearing users. Processed video frames and audio signals are sent to the pro- cessing pipeline with little delay, maintaining real-time system responsiveness.

### *B. Gesture Processing and Recognition*

The gesture processing layer is the main part of the system. Here, raw hand movement data turns into patterns that machines can recognize. Landmark coordinates created by MediaPipe change into numerical feature embeddings that show hand posture and motion. A lightweight TensorFlow Lite classifier, designed for mobile devices, processes these embeddings.

The recognition model operates on the device, which re- moves the need for continuous internet access and speeds up response time. An integrated interpreter handles real-time predictions, while an error correction and smoothing system reduces noise and stops false detections from quick hand movements or overlapping gestures. Running the recognition model locally ensures steady performance, saves energy, and protects privacy.

### *C. Translation, Output, and Interaction*

Once the system recognizes a gesture or speech input, the translation and output module changes the processed information into formats that users can understand. Recognized sign gestures turn into readable text, and a built-in text-to-speech engine produces natural audio output for hearing users. The system can translate in both directions, allowing gesture-to- text or speech conversion and speech-to-sign translation using visual sign representations like images or animations.

A multilingual support system enables users to switch be- tween American Sign Language (ASL), British Sign Language (BSL), and Indian Sign Language (ISL) by using configurable gesture datasets. This flexibility promotes inclusivity and usability across different language communities. Users can interact using voice, text, or sign language, depending on their preference.

*D. Cloud Integration and Data Management*

Although the main translation feature works offline, cloud services are included to improve scalability and personalize user experience. Firebase Authentication provides a secure user login through email or OTP-based verification. Firebase Firestore and Realtime Database store user profiles, language preferences, and learning progress. Cloud Storage keeps gesture datasets and educational resources.

A synchronization manager makes sure that user data and preferences are updated across devices whenever there is network connectivity. This hybrid cloud setup boosts system reliability and maintainability without sacrificing offline access.

*E. Comparative Architectural Design*

During system design, we evaluated two architectural con- figurations. A fully on-device architecture provided fast processing and offline capability. However, it faced limitations due to storage constraints and model update flexibility. The chosen hybrid architecture combines on-device inference with cloud- based data management. This approach achieves a good balance between performance, scalability, and storage. It allows for continuous improvement through dataset updates while ensuring real-time responsiveness.

*F. Development and Implementation Methodology*

The development process followed a clear and repetitive approach to ensure accuracy, usability, and system reliability. We analyzed requirements by interacting with sign language educators and potential users to identify their needs. These included real-time translation, multilingual support, and offline operation. We also noted non-functional needs such as low latency, security, and device compatibility.

We collected gesture datasets from various public sources and added our own recorded samples. We used data preprocessing techniques to adjust lighting conditions, hand orientation, and frame size. A lightweight deep learning model based on CNN and MobileNet architectures was trained and optimized for deployment with TensorFlow Lite. We validated the model by evaluating accuracy, analyzing confusion matrices, and making improvements through iterations.

We developed the Android application in Kotlin using the MVVM pattern to improve maintainability and scalability. We used CameraX for high-performance video capture and integrated MediaPipe for real-time hand tracking. We implemented speech-to-text functionality using Google's Speech Recognition API, and we mapped the recognized text to corresponding sign representations to support two-way communication.

We incorporated Firebase services for authentication, data storage, and real-time synchronization. We conducted ex- tensive unit, integration, and user testing to assess gesture recognition accuracy, system stability, and user experience. We addressed identified issues like lighting sensitivity and gesture misclassification through ongoing optimization. Finally, we optimized the application with model quantization and prepared for deployment while ensuring appropriate privacy compliance and performance checks.

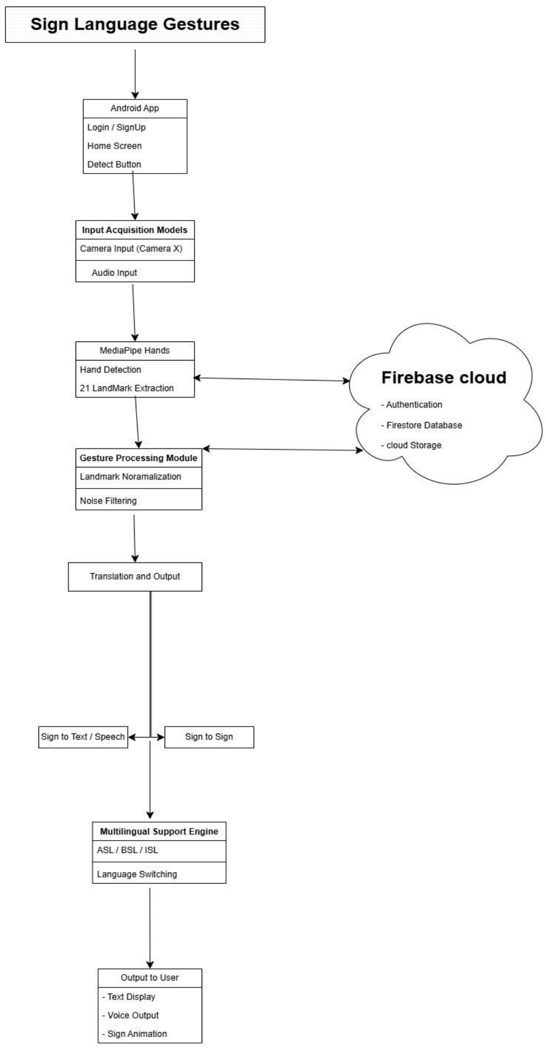


Fig. X. Overall Workflow of the Real-Time Sign Language Translator App

## IV. RESULTS

The Real-Time Sign Language Translator system was de- signed, implemented, and tested through a multi-stage experimental process to assess its effectiveness, accuracy, and real- world usability. The evaluation took place at both the model level and the application level. The results confirm accurate sign language recognition, effective bidirectional translation, and smooth integration of the trained model into an Android application.

### A. Model-Level Gesture Detection and Classification

At the start, the trained gesture recognition model was evaluated on its own using Google Colab. MediaPipe Hands detected and tracked hand landmarks in real time, capturing exact finger and palm movements. The trained classification model processed the extracted landmarks to identify related sign language gestures. The model showed stable and accurate predictions in different lighting conditions and hand positions.

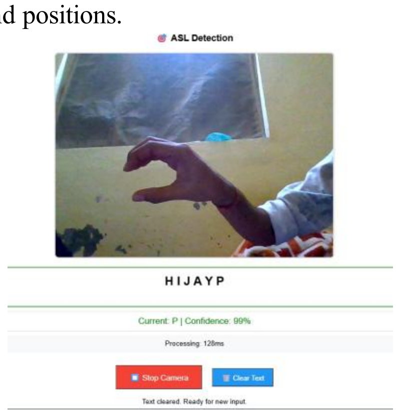


Fig. 1: Model-Level Sign Language Detection in Google Colab

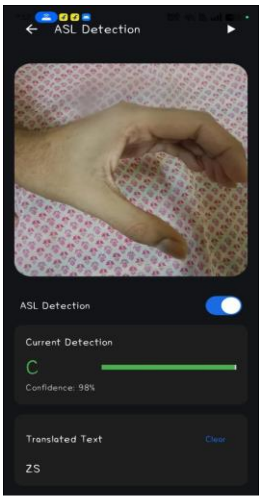


Fig. 2: Real-Time Sign Language Detection in the Android Application

### B. Bidirectional Translation: Language to Sign

The system was also tested for reverse translation, where spoken or written input was changed into corresponding sign representations. Speech recognition successfully converted spoken language into text, which was then linked to the right sign visuals. This two-way translation ability allows for meaningful communication between hearing users and sign language users.

### C. Application Authentication and Navigation

After validating the model, it was added to the Android application. The application offers secure login and signup features to control access to system functions. Once users are authenticated, they are taken to the home screen, which acts as the main navigation interface.

### D. Real-Time Detection in Android Application

From the home screen, users can start real-time sign language detection by choosing the detection option. The application uses the device camera and recognizes gestures live with the built-in model. It translates recognized gestures into text or speech immediately, with little delay.

### E. Overall System Performance and User Experience

The system showed smooth navigation among the login, home, and detection screens. On-device inference provided quick response times. User testing confirmed that the interface is easy to use and works well for real-world communication situations.

## V. CONCLUSION

The Real-Time Sign Language Translator App shows how modern mobile technology and artificial intelligence can make communication easier for everyone. The system combines hand gesture recognition, on-device machine learning, and an Android-based interface to allow real-time translation between sign language and spoken language. MediaPipe ensures precise hand landmark detection, while TensorFlow Lite enables quick and efficient gesture recognition directly on the device.The application features a simple and accessible interface that allows smooth interaction for both sign language users and those who do not use sign language. It works offline for key features, which helps in areas with low connectivity. Firebase integration offers secure login and data syncing. The system's modular design allows for easy updates and reliable performance Overall, the project meets its goal of breaking down communication barriers and encouraging inclusivity. The system lays a solid groundwork for future improvements, such as better accuracy, more language options, and greater real-world usage.

## REFERENCES


[1] Google Developers, "MediaPipe Hands: Real-Time Hand and Finger Tracking," 2023.

[2] TensorFlow Lite, "Deploying Machine Learning Models on Mobile and Edge Devices," 2023.

[3] Y. Zhang, Z. Zhou, and C. Wang, "Real-Time Sign Language Recogni- tion Based on Deep Learning," ICVISP, 2020.

[4] Kaggle, "ASL Alphabet Dataset," 2017. [Online]. Available: https://www.kaggle.com/grassknoted/asl-alphabet

[5] Firebase Documentation, "Authentication and Realtime Database," 2023. [Online]. Available: https://firebase.google.com/docs

[6] Y. Zhang, Z. Zhou, and C. Wang, "Real-time Sign Language Recog- nition Based on Deep Learning," in *Proceedings of the International Conference on Vision, Image, and Signal Processing (ICVISP)*, 2020.

[7] K. Patel, H. Rathod, and R. Gajera, "Hand Gesture Recognition Using Image Processing," *International Journal of Advanced Research in Computer Science and Software Engineering*, vol. 6, no. 5, pp. 279– 283, 2016.

[8] F. Chollet, *Deep Learning with Python*. Manning Publications, 2017.