Intelligent Sign Language Translation Solution for inclusive hearing

Sadhvi Thakur[1] Prof. Sandarsh Gowda MM[2]

[1]Student, Department of MCA, Bangalore institute of Technology, Karnataka, India
[1]Assistant Professor, Department of MCA, Bangalore institute of Technology, Karnataka, India

---***---

Abstract

Communication barriers for people with hearing and speech impairments pose challenges in education, healthcare, and everyday social interaction. This project, SignSpeak: Real-Time Sign Language Translator, tackles this issue by creating a gesture recognition system that translates sign language into text and speech directly in the browser. The system uses ml5.js along with MobileNet and KNN Classifier for machine learning in the browser. This setup allows for lightweight deployment without needing backend support.

Key features include real-time gesture training, prediction, and classification. Users can save and load trained models in JSON format for reuse and offline access. The design prioritizes user-friendliness and accessibility, offering tutorials, gesture-to-text, and gesture-to-speech modules. A gesture dictionary helps users manage, retrain, and review their gestures effectively.

The system was tested with various users, including members of the deaf community, and was refined based on their feedback. By providing an affordable, portable, and inclusive communication tool, this project helps connect sign language users with non-signers, promoting accessibility and social inclusion in real-life situations.

1.INTRODUCTION

Communication is a basic human need. It allows people to share ideas, feelings, and knowledge. For those in the global community who have hearing and speech impairments, sign languages are their primary way of communicating. Sign languages are rich, structured, and expressive visual languages that use hand gestures, facial expressions, and body movements. However, many people do not understand them, which limits their effectiveness in daily interactions. This creates a communication gap that can lead to social exclusion, dependence on interpreters, and barriers in education, work, and healthcare. Therefore, there is a growing need for technology that can bridge this gap by translating sign language into forms that non-signers can understand.

In the last decade, researchers have studied Sign Language Recognition (SLR) using various computational methods. Early studies used hardware solutions like sensor gloves and motion tracking devices to capture gestures accurately. While these systems were effective, they were also costly, intrusive, and impractical for everyday use. Later, vision-based methods became popular. They recognized hand gestures through traditional machine learning techniques such as K-Nearest Neighbors (KNN) and Support Vector Machines (SVM). Although these approaches had moderate success in controlled settings, they faced challenges in real-world situations with changing backgrounds, lighting conditions, and different signing styles.

The rise of deep learning marked a significant step forward in SLR. Convolutional Neural Networks (CNNs) helped extract powerful spatial features from static gestures, while Recurrent Neural Networks (RNNs) and Long Short-Term Memory (LSTM) architectures expanded recognition to dynamic gestures, capturing temporal patterns. Recently, object detection systems like YOLO have been used for real-time sign detection. They have achieved high accuracy in controlled datasets. Transfer learning has also accelerated model development. It allows pre-trained networks to be adapted to sign language tasks with smaller datasets. Despite these advances, many of these systems still require high computational power, specialized GPUs, or cloud services, which makes them less accessible to a larger audience.

At the same time, research has aimed to combine recognition with text and speech synthesis, allowing recognized signs to be displayed as written or spoken language. While this is a positive move toward practical applications, current systems often have fixed vocabularies, lack adaptability for new gestures, and rely on external hardware or proprietary platforms. These limitations restrict their widespread use in everyday communication.

To address these challenges, this work introduces SignSpeak: A Real-Time Sign Language Translator. This lightweight, browser-based system eliminates the need for specialized hardware or backend servers. It uses ml5.js, a high-level machine learning library built on TensorFlow.js, to combine MobileNet feature extraction with a KNN classifier for gesture recognition using just a webcam. Unlike traditional models, SignSpeak allows users to train and customize their own gestures interactively, save models for offline use, and easily convert recognized gestures into text and speech outputs. The system prioritizes user accessibility and inclusivity with a simple interface, gesture dictionary, and mobile responsiveness. This makes it suitable for real-world applications in education, healthcare, and personal communication.

In summary, this research responds to the need for an affordable, portable, and flexible solution to close the communication gap between signers and non-signers. By leveraging the strengths of computer vision, lightweight deep learning, and user-centered design, this project aims to support the broader goal of digital inclusivity. It demonstrates how browser-based AI applications can create practical tools for everyday accessibility.

II. LITERATURE SURVEY

Research on Sign Language Recognition (SLR) includes classical machine learning, deep learning, and real-time systems. All of these aim to improve communication between signers and non-signers. Early work focused on recognizing static gestures. This involved mapping still images of hand shapes to alphabet classes. Kumar and Singh’s Signet used a CNN to recognize Indian Sign Language (ISL) alphabets with strong accuracy in controlled settings. However, it did not support continuous signing and struggled in uncontrolled environments [1]. Classical approaches that relied on handcrafted features with KNN/SVM classifiers had competitive results for small vocabularies. Yet, their performance declined with changes in lighting, background clutter, and user pose. This showed the limitations of non-deep methods in terms of robustness [2].

To address the time-sensitive nature of signing, researchers moved toward dynamic recognition. Yuan and Zhang used an RNN-Transducer to capture sequence dependencies for continuous SLR. This approach focused on decoding not just isolated signs, but entire sentences. However, these models required more computational power and data, making them harder to deploy on standard hardware [3]. Other efforts combined skeleton and pose features with CNN classifiers. For example, OpenPose-based pipelines tailored to healthcare vocabularies achieved reliable term recognition. Still, their use was limited to specific domains and they were sensitive to camera setup and obstructions [4].

Recent advancements have come from real-time deep learning and detection-style architectures. Rahman and Hossain showed that YOLOv3 with DarkNet-53 could achieve high-throughput recognition and strong accuracy for Bangla Sign Language on selected datasets. This highlighted the effectiveness of single-stage detectors for low-latency applications [5]. At the same time, lightweight detector variants designed for embedded or browser contexts reduced inference costs while keeping practical frame rates. However, performance still declined with low-resolution inputs and fast hand movements [6]. To address data scarcity and speed up training, transfer learning methods that fine-tune pre-trained models on ISL datasets improved accuracy and convergence. Nevertheless, generalization beyond the training data and adaptation to new gestures remained challenges [7].

Translational systems are increasingly combining gesture-to-text and text-to-speech (TTS) to aid user communication. A typical mobile setup integrated YOLOv5 recognition with TTS to provide device feedback suitable for daily use. However, this setup usually worked with small, predefined vocabularies and demanded careful environmental control [8]. Broader examinations of SLR consistently point out ongoing issues: robustness in uncontrolled backgrounds, scalability for continuous signing, cross-lingual coverage, and the need for low-resource inference on common devices [9], [10].

III. EXISTING SYSTEM

Existing sign language recognition systems have made significant progress, but they still struggle with issues related to cost, scalability, and usability. Early systems relied on wearable sensors, gloves, or motion tracking devices to capture hand movements accurately. While these methods were effective, they were also expensive, bulky, and intrusive, making them impractical for everyday use. Later, vision-based methods employed image processing and traditional machine learning techniques like K-Nearest Neighbors (KNN) and Support Vector Machines (SVM) to identify static gestures from images. However, their accuracy depended heavily on lighting conditions, background complexity, and user variability, which limited their effectiveness in real-world situations.

With the rise of deep learning, researchers began using Convolutional Neural Networks (CNNs) and object detection algorithms like YOLO to achieve high accuracy and real-time gesture recognition. These systems demonstrated improved performance and could recognize both static and dynamic gestures. However, they required large labeled datasets and powerful computing resources, which made them less accessible to the general public. Some newer systems have combined text and speech synthesis with recognition, allowing gestures to be converted into readable text or spoken output. Despite this progress, these solutions usually offer only a limited set of predefined gestures and lack the flexibility for users to train or customize new signs.

In summary, existing systems are limited by hardware dependence, high costs, a narrow gesture vocabulary, and poor adaptability to real-time, user-driven contexts. These challenges highlight the need for a lightweight, low-cost, and customizable browser-based system that can operate on standard devices without the need for special hardware.

IV. PROPOSED SYSTEM

The proposed system, SignSpeak: Real-Time Sign Language Translator, aims to overcome the limitations of existing solutions by offering a lightweight, browser-based platform for real-time gesture recognition. Unlike traditional systems that rely on expensive sensors, gloves, or backend servers, this application runs entirely in a web browser using HTML5, CSS3, JavaScript (ES6), and the ml5.js library built on TensorFlow.js. This setup makes it highly portable, works on any platform, and keeps costs down. The system uses a MobileNet feature extractor and a KNN Classifier to process webcam input and classify hand gestures in real time. Users can train their own gestures, save models as JSON files, and reload them in future sessions. This ensures offline functionality and reusability.

Recognized gestures are instantly converted into text output. They can also be transformed into speech using the browser's Speech Synthesis API, providing two ways to communicate for inclusivity. The user interface is straightforward and easy to use. It includes modules for gesture learning and real-time recognition, along with a gesture dictionary that allows users to view, edit, retrain, or delete custom gestures. Features like confidence scoring, real-time feedback, and mobile responsiveness enhance user-friendliness. Accessibility options such as high-contrast themes, readable fonts, and tutorials ensure the system is suitable for a wide range of users.

By eliminating the need for specialized hardware, ensuring offline usability, and incorporating feedback from deaf users and educators, the proposed system helps close the communication gap between signers and non-signers in a practical and affordable way. Its portability and flexibility make it suitable for real-world applications in education, healthcare, public services, and everyday communication.

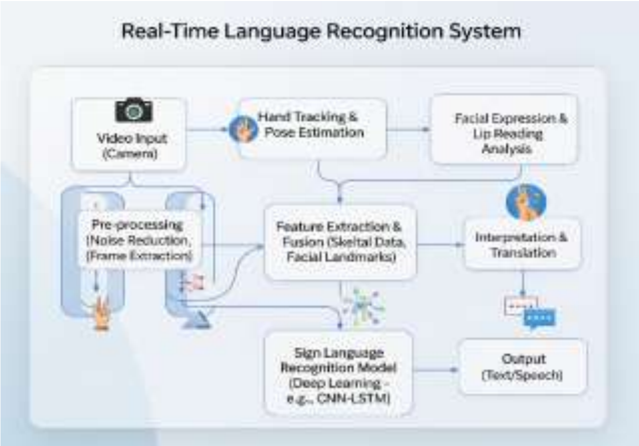

Fig 1: Proposed Model

V. IMPLEMENTATION

The development of SignSpeak: Real-Time Sign Language Translator happened in several phases. It started by setting up the technology stack, then gradually built into a fully functional system. The application was created entirely as a browser-based solution using HTML5, CSS3, and JavaScript. The main library for machine learning was ml5.js. We integrated the MobileNet feature extractor and KNN Classifier to enable real-time gesture recognition directly from the webcam. This setup eliminated the need for any backend or specialized hardware.

The process began with creating a well-organized project environment. This included folders for assets, training data, and models. We also set up a GitHub repository for version control. The user interface was designed to be accessible. It included a landing page, a gesture learning module, and a recognition module. We used CSS Flexbox and Grid to make the system mobile-responsive. We added animations and tutorials to help users learn basic signs like “Hello,” “Yes,” “No,” and “Sorry.”

We implemented the main functionality in stages. First, we captured the webcam feed using the getUserMedia API. We processed this feed through MobileNet to extract feature vectors. The KNN classifier stored these vectors along with the gesture labels provided by the user. A real-time prediction loop classifies gestures every 100 milliseconds. The predicted label and confidence score appeared live on the screen. Users received guidance through built-in counters to capture several training samples for each gesture, which improved recognition accuracy. To ensure data persistence, we included a save and load feature using JSON files. This allowed users to export, reload, and reuse trained models in different sessions.

We also added more features, such as a gesture dictionary for managing custom gestures, a gesture-to-text output, and speech synthesis integration. The Speech Synthesis API provided optional voice feedback. We improved the interface with status notifications, error handling for invalid file uploads, and user-friendly controls like “Start Prediction,” “Stop Prediction,” and “Download Model.”

We conducted extensive testing with multiple users, including members of the deaf community. This helped us evaluate recognition accuracy, usability, and accessibility. Feedback from these sessions led to improvements in tutorials, interface design, and prediction handling. This ensured that the system was practical and inclusive for real-world use.

VI. CONCLUSIONS

This work presents SignSpeak: A Real-Time Sign Language Translator, a lightweight, browser-based system that connects sign language users with those who do not use sign language. By using ml5.js, MobileNet, and a KNN Classifier, the system captures, trains, and recognizes gestures directly from a webcam in real-time. It does not need special hardware or backend support. It also includes gesture-to-text and gesture-to-speech outputs, as well as a gesture dictionary for customization. This improves accessibility and usability.

Through ongoing development and testing with a variety of participants, including members of the deaf community, the system has proven to be practical, flexible, and inclusive. Unlike current solutions that face problems with cost, hardware requirements, or fixed vocabularies, SignSpeak allows users to train and customize gestures easily. It ensures portability and offline use.

This project highlights the possibilities of browser-based machine learning applications and supports the broader goal of digital inclusion. By providing an affordable and scalable communication tool, this system can be used in various fields such as education, healthcare, public services, and personal interactions. It encourages independence and integration for individuals with hearing or speech challenges.

Future developments may include expanding support for dynamic continuous signing, adding multi-language translation, and improving accuracy on low-end devices. Still, the current version marks a significant step toward achieving real-time, accessible, and inclusive communication for everyone.

VII. FUTURE ENHANCEMENTS

Although the proposed system provides real-time sign language recognition and translation, there are several ways to improve it in the future. One major improvement is to extend the system to support continuous signing. This would allow it to recognize full sentences rather than just isolated gestures, making communication more natural and conversational. Expanding the gesture library to include various sign languages, such as Indian Sign Language (ISL), American Sign Language (ASL), and other regional variants, would make the system more inclusive and widely applicable.

Another area to improve is accuracy and reliability. By using larger and more diverse datasets and adding background segmentation techniques, the system can better handle changes in lighting, orientation, and complex environments. Additionally, making the solution available as a mobile app on Android and iOS would improve its portability and accessibility. Lightweight optimization could also allow it to run on edge devices like Raspberry Pi.

The system can be integrated with assistive technologies, including video conferencing platforms, smart glasses, or hearing aids. This would increase its usefulness in education, healthcare, and professional settings. Furthermore, personalization features like adaptive learning could help the model constantly improve its predictions based on an individual's signing style. A cloud-based gesture repository could also support shared models and collaborative enhancements. Lastly, future research may go beyond hand gestures to include facial expressions and body movements, capturing the emotional and contextual depth of sign language communication.

VIII REFERENCES

[1] A. Kumar and R. Singh, "Signet: A deep learning based Indian sign language recognition system," International Journal of Computer Applications, vol. 178, no. 7, pp. 15, 21, 2019.

[2] R. Kalsh and N. Saxena, "Hand gesture recognition using machine learning," in Proc. 2020 11th Int. Conf. Computing, Communication and Networking Technologies (ICCCNT), IEEE, 2020.

[3] Y. Yuan and L. Zhang, "RNN-transducer based continuous sign language recognition," Journal of Visual Communication and Image Representation, vol. 71, pp. 102842, 2020.

[4] S. Sahoo and P. Behera, "Dynamic Indian sign language recognition in healthcare using OpenPose and CNN," Procedia Computer Science, vol. 167, pp. 2402, 2411, 2020.

[5] M. Rahman and A. Hossain, "Bangla sign language detection using YOLOv3 and DarkNet-53," Multimedia Tools and Applications, vol. 82, no. 9, pp. 12785, 12801, 2023.

[6] J. Lin and H. Wang, "Real-time hand gesture recognition using lightweight YOLOv3," in Proc. 2021 IEEE Int. Conf. Image Processing (ICIP), IEEE, 2021.

[7] P. Das and A. Sengupta, "Transfer learning approach for Indian sign language recognition," Journal of King Saud University – Computer and Information Sciences, vol. 34, no. 6, pp. 3125, 3134, 2022.

[8] R. Verma and A. Singh, "Mobile-based Indian sign language translation using YOLOv5 and text-to-speech synthesis," International Journal of Advanced Computer Science and Applications, vol. 15, no. 2, pp. 55, 63, 2024.

[9] T. Ahmed and K. Rahman, "Gesture recognition for sign language translation using deep learning," in Proc. 2022 Int. Conf. Emerging Technologies in Computing (iCETiC), Springer, 2022.

[10] Z. Ali, M. Iqbal, and S. Khan, "A survey on sign language recognition: Methods and future directions," IEEE Access, vol. 10, pp. 54672, 54692, 2022.