SIGN LANGUAGE ALPHABET IDENTIFIER

CS19643 – FOUNDATIONS OF MACHINE LEARNING

Submitted by

JOHN PRATHAP SINGH S **(2116220701112)**

in partial fulfillment for the award of the degree

of

BACHELOR OF ENGINEERING

in

COMPUTER SCIENCE AND ENGINEERING

RAJALAKSHMI ENGINEERING COLLEGE

ANNA UNIVERSITY, CHENNAI

MAY 2025

RAJALAKSHMI ENGINEERING COLLEGE, CHENNAI

BONAFIDE CERTIFICATE

Certified that this Project titled **"SignAlpha - Sign Language Alphabet Identifier"** is the bonafide work of **"JOHN PRATHAP SINGH S (2116220701112)"** who carried out the work under my supervision. Certified further that to the best of my knowledge, the work reported herein does not form part of any other thesis or dissertation on the basis of which a degree or award was conferred on an earlier occasion on this or any other candidate.

SIGNATURE

Dr. V. Auxilia Osvin Nancy, M.Tech., Ph.D.,
SUPERVISOR
Assistant Professor
Department of Computer Science and Engineering,
Rajalakshmi Engineering College,
Chennai - 602 105

Submitted to the Project Viva-Voce Examination held on _____

Internal Examiner **External Examiner**

ABSTRACT

The Sign Language Alphabet Identifier project aims to bridge the communication gap between hearing-impaired individuals and the broader community by translating hand gestures representing alphabets into readable text using computer vision and machine learning techniques. This system captures real-time video feed via a webcam and identifies hand signs corresponding to the sign language alphabet.

The project consists of four main stages: data collection, dataset generation, model training, and real-time prediction. First, images of hand gestures for multiple classes (representing alphabets or numbers) are collected using a webcam and stored in a structured dataset. These images are processed using MediaPipe to extract hand landmarks, which serve as robust and lightweight features for training. A Random Forest classifier is then trained on the extracted landmark features to recognize different sign language gestures. Finally, the trained model is deployed in a real-time application where it continuously processes live video input, detects hand landmarks, and predicts the corresponding alphabet using the trained model.

This approach provides a cost-effective, real-time solution for sign language recognition without the need for specialized equipment. The system demonstrates high accuracy and responsiveness, showcasing its potential for integration into educational tools, accessibility technologies, and communication aids for the hearing-impaired community.

ACKNOWLEDGMENT

Initially, we thank the Almighty for being with us through every walk of our lives and showering his blessings through the endeavor to put forth this report. Our sincere thanks to our Chairman, **Mr. S. MEGANATHAN, B.E., F.I.E.**, our Vice Chairman, **Mr. ABHAY SHANKAR MEGANATHAN, B.E., M.S.,** and our respected Chairperson, **Dr. (Mrs.) THANGAM MEGANATHAN**, **Ph.D.,** for providing us with the requisite infrastructure and sincerely endeavoring to educate us in their premier institution.

Our sincere thanks to **Dr. S.N. MURUGESAN, M.E., Ph.D.,** our beloved Principal, for his kind support and facilities provided to complete our work in time. We express our sincere thanks to **Dr. P. KUMAR, M.E., Ph.D.**, Professor and Head of the Department of Computer Science and Engineering, for his guidance and encouragement throughout the project work. We convey our sincere and deepest gratitude to our internal guide, **Dr. S. VINOD KUMAR, M.Tech., Ph.D.,** Professor of the Department of Computer Science and Engineering. Rajalakshmi Engineering College for their valuable guidance throughout the course of the project. We are very glad to thank our Project Coordinator, **Mr. M. RAKESH KUMAR,** Assistant Professor, Department of Computer Science and Engineering, for his useful tips during our review to build our project.

JOHN PRATHAP SINGH S 2116220701112

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF ABBREVIATIONS

S. No	ABBR	Expansion
1	AI	Artificial Intelligence
2`	API	Application Programming Interface
3	ASL	American Sign Language
4	CV	Computer Vision
5	DFD	Data Flow Diagram
6	GB	Gradient Boosting
7	GUI	Graphical User Interface
8	JPEG/JPG	Joint Photographic Experts Group
9	JSON	JavaScript Object Notation
10	ML	Machine Learning
11	MP	MediaPipe
12	OpenCV	Open Source Computer Vision Library
13	RF	Random Forest
14	ROI	Region of Interest
15	SVM	Support Vector Machine
16	UI	User Interface
17	XML	eXtensible Markup Language

CHAPTER - 1 INTRODUCTION

1.1 GENERAL

SignAlpha: Revolutionizing Communication Through AI-Powered Sign Language Interpretation

SignSpeakAI is an intelligent assistive technology platform designed to bridge the communication gap between individuals with hearing or speech impairments and the general population. By leveraging advanced machine learning models and real-time computer vision techniques, the platform enables automatic recognition and translation of American Sign Language (ASL) alphabet into readable text.

At its core, SignAlpha utilizes sophisticated hand-tracking frameworks like MediaPipe and OpenCV to detect hand gestures and process them through trained machine learning algorithms, including Random Forest and Support Vector Machine classifiers. The system captures hand signs through a webcam, classifies them in real-time, and displays the corresponding alphabet on-screen, providing a seamless and non-intrusive interface for communication.

Built using modern web technologies and integrated into a secure, scalable environment, SignAlpha prioritizes speed, accuracy, and user accessibility. It features an intuitive dashboard, real-time gesture feedback, and visual guides to help users understand and navigate the system effortlessly. This solution not only supports inclusive communication but also offers educational benefits for learners and institutions aiming to teach or learn sign language effectively.

Through continuous innovation, user-focused design, and AI-driven learning, SignAlpha transforms the way we interpret hand signs, making everyday interaction more inclusive, accessible, and empowering for everyone.

1.2 OBJECTIVE

The primary objective of SignAlpha is to develop a smart, real-time, and user-friendly platform that recognizes hand gestures from American Sign Language (ASL) and translates them into corresponding alphabets using machine learning and computer vision.

The platform aims to:

- **Accurately detect and recognize ASL Alphabets** using real-time webcam input and advanced gesture tracking tools such as MediaPipe and OpenCV.
- **Classify Hand Signs** with high precision using trained machine learning models like Random Forest and Support Vector Machine.
- **Enhance Communication Accessibility** for individuals with hearing or speech impairments by providing an intuitive digital interface for sign-to-text translation.
- **Provide Real-Time Feedback** through a responsive user interface that displays recognized characters immediately as signs are made.
- **Support Educational Applications** by serving as a learning aid for students and teachers interested in ASL.
- **Ensure Performance and Usability** by optimizing recognition speed, accuracy, and responsiveness for diverse users, regardless of technical background.
- **Promote Inclusion** by reducing communication barriers and fostering understanding across different communities.

1.3 EXISTING SYSTEM

Currently, individuals who rely on sign language for communication often face challenges when interacting with people who do not understand ASL. Traditional solutions include human interpreters or manual chart-based learning systems, which are not always practical or readily available in real-time scenarios.

While some mobile applications and websites offer static sign language charts or video tutorials, they lack the ability to recognize and translate signs dynamically. Others rely heavily on pre-recorded datasets or require highly controlled environments for sign recognition, limiting their real-world usability.

Moreover, many existing tools suffer from limitations such as:

- Lack of real-time recognition and feedback
- Poor accuracy in varied lighting or backgrounds
- Limited support for different hand orientations or sizes
- Absence of machine learning personalization or adaptability

These constraints make current systems insufficient for everyday, responsive, and user-friendly sign-to-text communication. As a result, users are left without reliable, accessible solutions to assist in natural, real-time conversations—especially in educational, social, or emergency settings.

CHAPTER 2
LITERATURE SURVEY

“Real-Time Hand Gesture Recognition using MediaPipe and CNN” [1] (2023) by Sharma et al. investigates the integration of MediaPipe with Convolutional Neural Networks (CNN) for real-time recognition of hand gestures. The system leverages MediaPipe’s hand landmark detection to extract feature vectors, which are then classified using CNN models to detect alphabets and basic gestures. While the framework demonstrates high speed and accuracy in controlled environments, it encounters challenges in varied lighting conditions and with complex or overlapping gestures.

“AI-Based Sign Language Interpreter using Computer Vision” [2] (2024) by Khan and Verma presents an AI-powered platform that translates American Sign Language (ASL) gestures into text using OpenCV and deep learning techniques. The system applies a feed-forward neural network trained on hand sign images to recognize individual letters. Though it succeeds in identifying isolated signs, it struggles with continuous gestures and real-time responsiveness, highlighting the need for more optimized algorithms and gesture segmentation.

“Machine Learning Techniques for Hand Sign Classification” [3] (2022) by Li and Zhou evaluates multiple machine learning models—Random Forest, Support Vector Machines (SVM), and k-NN—for classifying ASL alphabets based on extracted hand shape features. Random Forest and SVM models achieved higher accuracy and better generalization on unseen test data. However, limitations included the requirement for large labeled datasets and difficulties in handling background noise or partial occlusion in images.

“Gesture Recognition using MediaPipe and Transfer Learning” [4] (2023) by Dasgupta et al. introduces a hybrid model that combines MediaPipe hand tracking

with transfer learning models like MobileNetV2. This approach improves model training efficiency and enables accurate alphabet recognition even with smaller datasets. While transfer learning accelerates development, the paper notes limitations in model interpretability and the need for high-quality pre-trained models compatible with sign language datasets.

“Real-Time ASL Detection Using YOLOv5 and Deep Learning” [5] (2024) by Mukherjee and Iyer implements a real-time sign recognition system using YOLOv5 object detection on video frames to identify hand regions and classify them into ASL alphabets. The system offers fast detection rates but has a higher resource requirement and can experience false positives in dynamic backgrounds.

“Smart Assistive Systems for Sign Language Translation” [6] (2023) by Hernandez et al. explores the development of assistive devices and software that translate sign language into spoken or written text using AI. The study discusses sensor-based gloves, camera-based tracking, and hybrid systems, emphasizing user-centric design and accessibility. Despite promising results, challenges include high implementation cost, variability in individual signing styles, and the need for continuous system updates to accommodate different sign languages and dialects.

CHAPTER 3

PROPOSED SYSTEM

3.1 GENERAL

The proposed system aims to develop a robust, real-time Sign Language Alphabet Recognition System leveraging Artificial Intelligence (AI), Machine Learning (ML), and Computer Vision technologies. Designed to assist individuals with hearing or speech impairments, the system translates hand signs representing English alphabets into corresponding textual output, fostering better communication between sign language users and non-signers.

At the core of the system is a camera-based input module that captures hand gestures performed by the user. Using **MediaPipe** for efficient hand landmark detection, the system extracts key hand features such as finger positions, angles, and joint coordinates. These features are fed into a trained ML model—such as a Convolutional Neural Network (CNN) or Random Forest classifier—that has been optimized to identify static signs representing alphabets (A–Z).

The recognized output is displayed in real-time on the screen in textual form, providing instant visual feedback. The system is designed to work across varied lighting conditions and backgrounds, incorporating preprocessing techniques like background filtering and normalization to improve accuracy. It is lightweight and capable of running on consumer-grade hardware without specialized sensors, making it affordable and accessible.

This solution not only supports inclusive communication but also serves as an educational tool for learning sign language. The system is built with scalability in mind, enabling future enhancements such as sentence-level recognition, multi-language support, and integration with text-to-speech modules.

By combining modern AI techniques with intuitive design, the proposed system represents a significant step toward breaking down communication barriers and enhancing digital accessibility for the differently-abled community.

3.2 SYSTEM ARCHITECTURE DIAGRAM

The architecture of the proposed Sign Language Alphabet Recognition System is designed for efficient real-time gesture detection, classification, and output display. It integrates multiple components that work seamlessly to process visual input and convert it into accurate textual output. The system architecture can be broadly divided into the following layers:

1. Input Layer – Video Frame Acquisition

- The system captures real-time video input using a webcam or device camera.
- Each frame is extracted and passed to the processing unit.
- The video stream is continuous, allowing for real-time alphabet recognition.

2. Preprocessing Layer

- **Hand Detection**: MediaPipe Hand or similar vision library detects and tracks the hand within each frame.
- **Landmark Extraction**: Key hand landmarks (e.g., finger tips, knuckles) are extracted and normalized to maintain scale and rotation invariance.
- **Noise Reduction**: Background clutter, varying lighting conditions, and motion blur are filtered using smoothing and thresholding techniques.

3. Feature Extraction Layer

Extracted hand landmark coordinates are used to compute spatial features like:

- Euclidean distances between joints

- Angles between fingers
- Relative positions of fingers and palm center

4. Classification Layer (Machine Learning Model)

- A trained model (e.g., Random Forest, CNN, or SVM) is used to classify the extracted features into one of the 26 English alphabets (A–Z).
- The model is pre-trained on a labeled dataset of hand signs for each alphabet.
- The classifier outputs the most probable alphabet label.

5. Output Layer

- The recognized alphabet is displayed on the screen in real-time.
- The system continuously updates the output with each new frame.
- Optionally, the output can be stored or converted to speech using a text-to-speech (TTS) engine in future expansions.

6. User Interface Layer

- A simple, user-friendly GUI displays the live camera feed and the corresponding recognized alphabet.
- Additional controls may be added for training, switching modes, or reviewing recognized text.

This modular architecture ensures scalability, allowing for easy integration of advanced features such as sentence recognition, speech synthesis, and multilingual support.

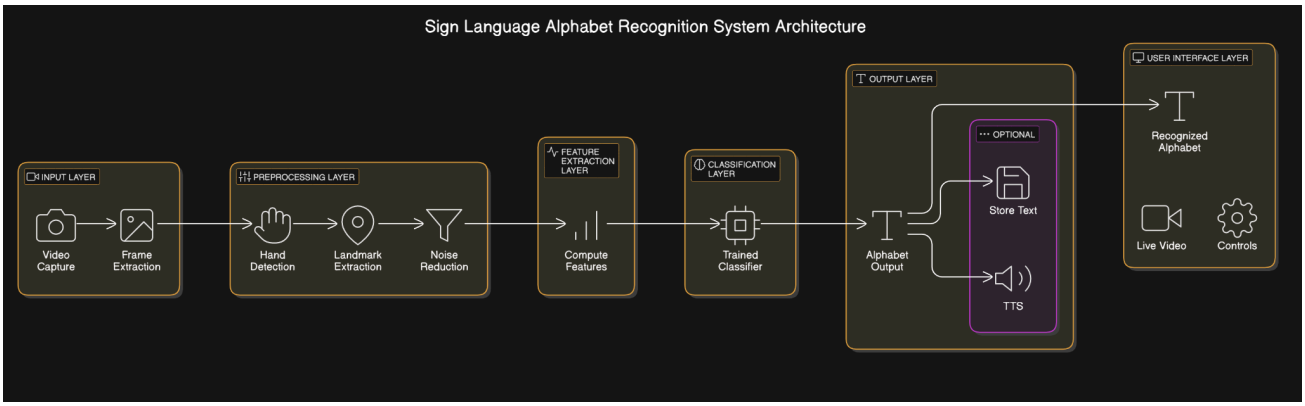

Fig 3.1: System Architecture

3.3 DEVELOPMENTAL ENVIRONMENT

3.3.1 HARDWARE REQUIREMENTS

The hardware specifications are essential for the efficient development, testing, and deployment of the SignAlpha system. They act as a reference standard for both development teams and hosting infrastructure.

Table 3.1 Hardware Requirements

COMPONENTS	SPECIFICATION
PROCESSOR	Intel Core i3
RAM	4 GB RAM
POWER SUPPLY	+5V power supply

3.3.2 SOFTWARE REQUIREMENTS

The software requirements define the tools and frameworks necessary for building the complete SignAlpha application. They are critical for ensuring consistency in development and supporting maintainable, scalable deployment.

Table 3.2 Software Requirements

COMPONENTS	SPECIFICATION
Operating System	Windows 7 or higher
Frontend	Python (OpenCV for webcam capture)
Backend	Flask
Machine Learning	Scikit-learn, TensorFlow
Model Serialization	Pickle
Libraries	OpenCV, MediaPipe, NumPy
Web Framework	Flask

3.4 DESIGN OF THE ENTIRE SYSTEM

3.4.1 ACTIVITY DIAGRAM

An activity diagram is a UML diagram that provides a visual representation of the flow of activities or tasks in the system. Here's the activity diagram for your Sign Language Recognition project, outlining the process from capturing hand gestures through the webcam to displaying predictions.

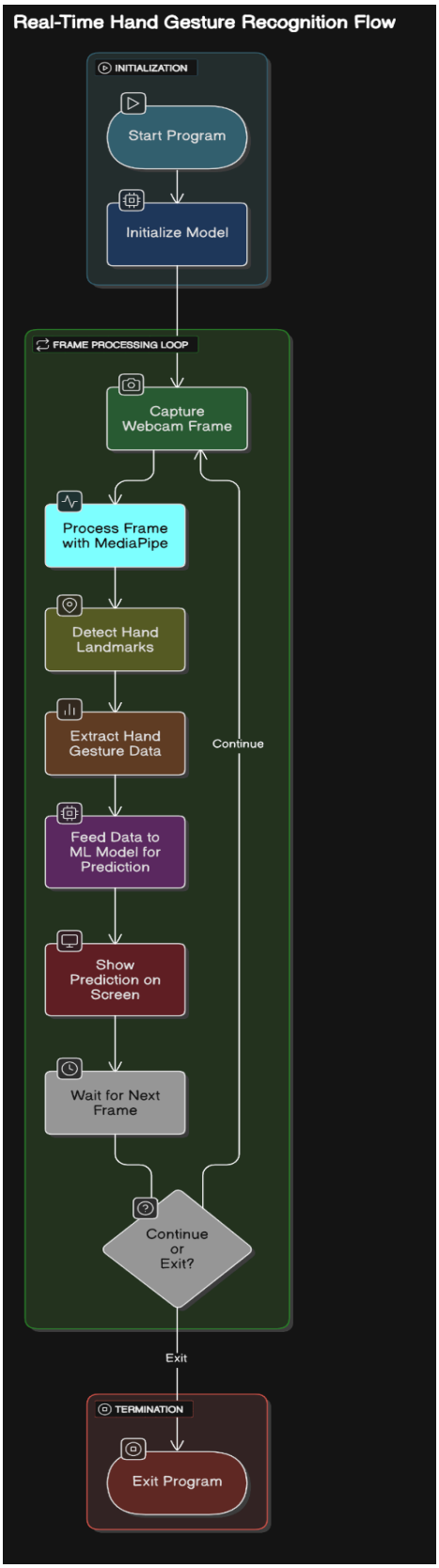

Fig 3.2: Activity Diagram

3.4.2 DATA FLOW DIAGRAM

The data flow diagram (Fig 3.3) outlines the process of credit card debt optimization and management using a smart recommendation engine built with Flask and ReactJS. The process begins with the user input, where individuals enter their credit card details, including outstanding balance, interest rate, minimum payment, and monthly repayment capacity. This input data undergoes preprocessing, where the system validates entries, handles missing information, and ensures data consistency. During the optimization phase, each strategy is analyzed based on parameters like earliest debt-free date, total interest paid, and monthly payment feasibility. The best strategies are ranked and recommended to the user. The results, including estimated debt-free dates, remaining balance projections, and savings analysis, are then served back to the frontend dashboard built with ReactJS

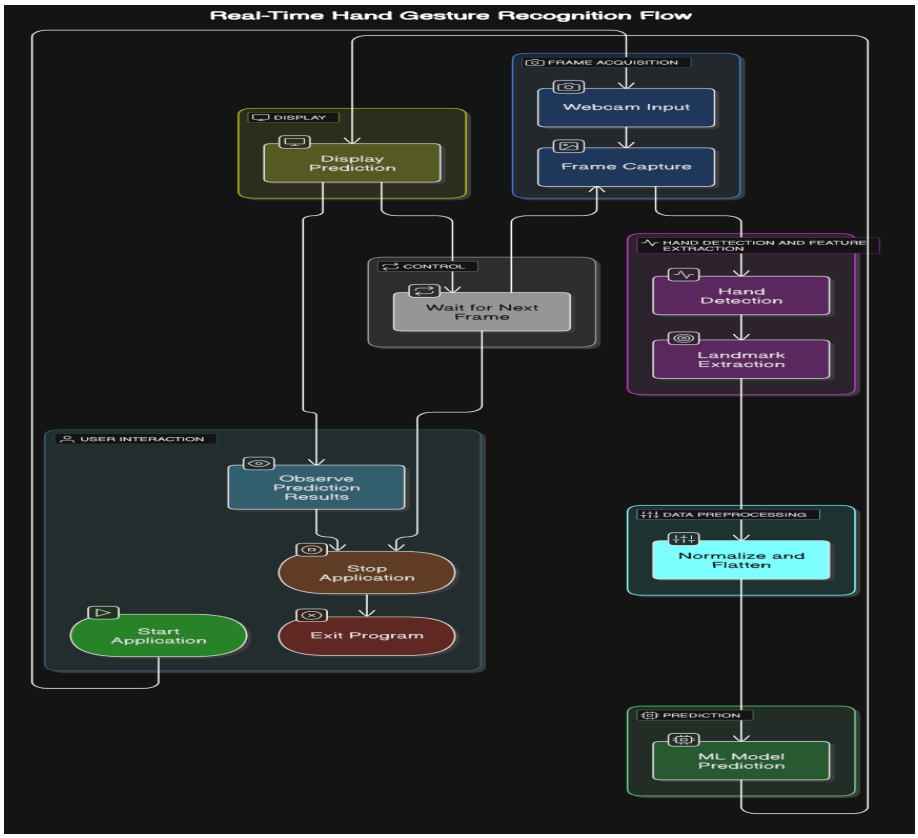

Fig 3.3: Data Flow Diagram

3.5 STATISTICAL ANALYSIS

The statistical analysis of the existing and proposed systems for the Sign Language Recognition project helps to compare key aspects, highlighting the improvements and expected outcomes. Below is the comparison table:

Table 3.3 Comparison of features

Aspect	Existing System	Proposed System	Expected Outcomes
Gesture Recognition	Basic gesture recognition with limited accuracy	Advanced machine learning model for improved accuracy	Higher recognition accuracy, reduced misinterpretation rates
Hand Detection	Uses basic image processing techniques	Uses MediaPipe for precise hand landmark detection	Enhanced precision in detecting hand gestures
Model Performance	Limited to basic models with low generalization	Use of pre-trained models and better generalization	Faster predictions, better handling of varied inputs
User Interface	Basic display of recognized gestures	Dynamic, real-time display of recognized gestures with webcam feedback	Better user experience with instant visual feedback
Prediction Speed	Relatively slow, depending on the hardware	Optimized model for faster predictions, possibly using hardware acceleration (GPU)	Reduced lag time between gesture input and output display

CHAPTER 4

MODULE DESCRIPTION

The workflow for the proposed SignAlpha system ensures a structured, efficient, and accurate process for real-time sign language recognition using computer vision and machine learning techniques. The process flows through modules including user interface interaction, video input processing, gesture detection, prediction, and response display.

4.1 SYSTEM ARCHITECTURE

4.1.1 USER INTERFACE DESIGN

The user interface, built with ReactJS and styled using CSS, offers an intuitive experience. Users can activate their webcam to perform gestures, view real-time recognition results, and optionally access their gesture history.

Security & Accessibility:

- Secure login with optional OTP (future enhancement).
- Responsive design ensures accessibility across devices.

Sequence Diagram (Fig 4.1):

- The user initiates webcam access → the system detects hand landmarks → the ML model classifies the gesture → the prediction is displayed in real-time on the dashboard.

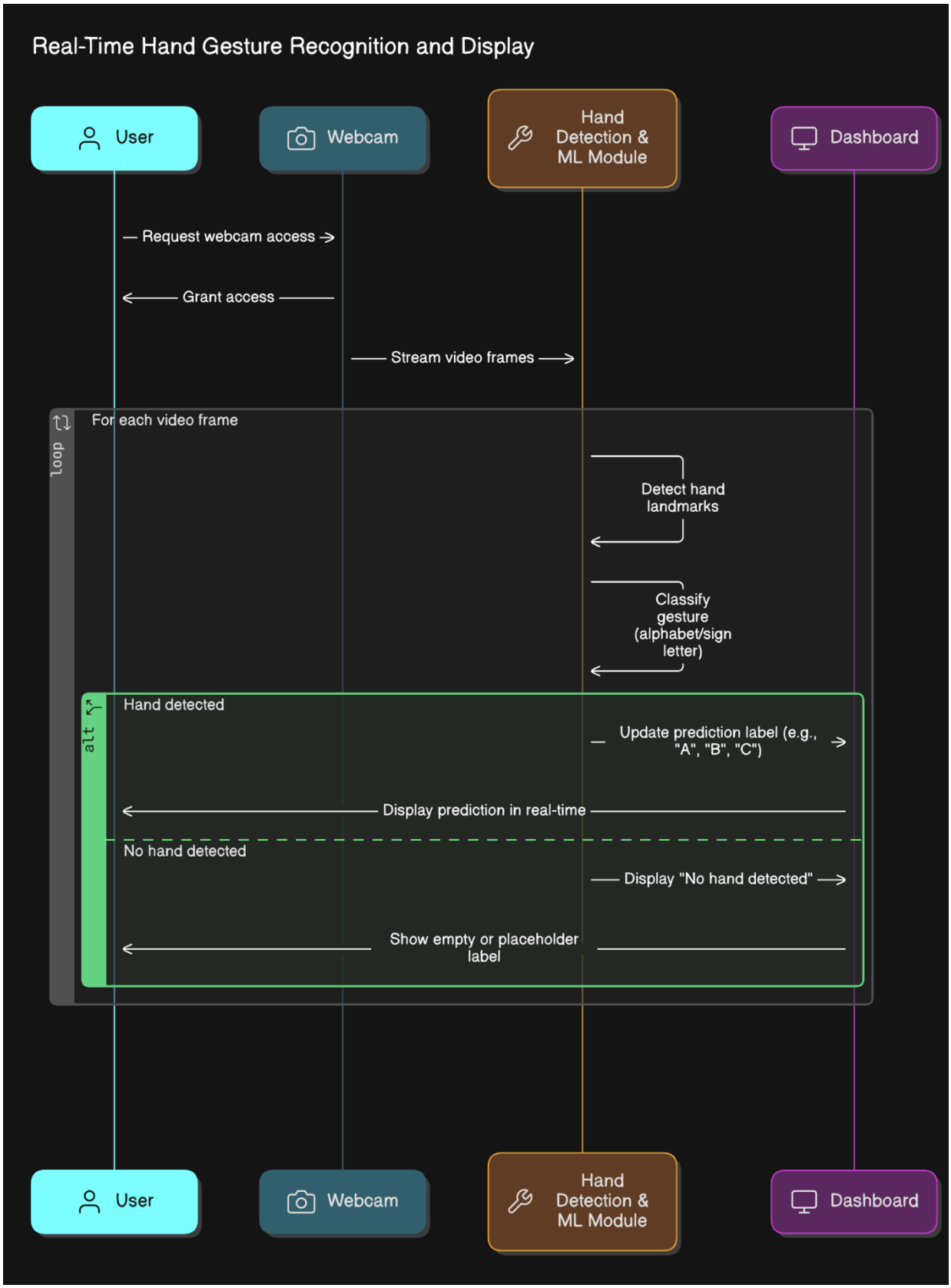

Fig 4.1: SEQUENCE DIAGRAM

4.1.2 BACK END INFRASTRUCTURE

The backend is developed using Python (Flask) and is responsible for real-time data handling and inference.

Key Components:

- WebSocket Integration: Facilitates real-time communication between the frontend and backend.
- MediaPipe Module: Extracts 21 hand landmarks from the webcam stream.
- ML Inference Engine: Predicts gestures from processed landmarks.
- Database (MongoDB): Stores user session data and recognition logs.

API Integration:

- Flask REST APIs handle recognition requests, user sessions, and data fetching for analytics.

4.2 DATA COLLECTION AND PREPROCESSING

4.2.1 Dataset and Data Labelling

The system is trained on a labeled dataset containing hand gesture images/videos corresponding to American Sign Language (ASL) alphabets. Each sample is tagged with its correct class (A–Z) for supervised training.

4.2.2. Data Preprocessing

Ensures the input fed into the model is clean and consistent:

- **Landmark Extraction:** Converts raw images into numerical vectors using MediaPipe.
- **Normalization:** Scales hand coordinates for model consistency.
- **Noise Removal:** Filters out frames with incomplete hand landmarks.

4.2.3 Feature Selection

The model uses:

- X and Y coordinates of 21 hand landmarks.
- Relative distances and angles between keypoints (as needed for accuracy).

4.2.4 Classification and Strategy Recommendation

Classification is performed using ML algorithms like:

- **K-Nearest Neighbors (KNN)**
- **Support Vector Machines (SVM)**
- **CNN (Future Upgrade)**

Future Strategy: Suggest the best model based on user hand patterns and environment (e.g., lighting conditions).

4.2.5 Performance Evaluation and Optimization

System reliability is ensured through:

- Accuracy evaluation using cross-validation.
- Real-time latency measurement for prediction.
- Performance tuning based on a feedback loop from user input.

4.2.6 Model Deployment(Future Enhancement)

The model is containerized and can be deployed on cloud servers for scalability. Future versions may:

- Support mobile app integration.
- Continuously retrain on new gestures uploaded by users.

4.2.7 Centralized Server

Flask server: Manages model inference and data communication.

4.3 SYSTEM WORK FLOW

4.3.1 User Interaction:

Users interact through a simple UI:

- Start/Stop Webcam.
- View gesture predictions in real-time.

4.3.2 Gesture Recognition and Response Display:

- Captures webcam input → extracts landmarks → feeds into ML model → displays recognized letter or word on screen.

4.3.3 Continuous Learning & Improvement:

Upcoming enhancements include:

- Model retraining using user-submitted gestures.
- Improved suggestions under varied lighting and backgrounds.
- An adaptive learning engine that improves predictions over time.

CHAPTER 5

IMPLEMENTATION AND RESULTS

5.1 IMPLEMENTATION

The **SignAlpha** project is developed and deployed using a reliable and efficient technology stack. The backend is powered by **Python Flask**, which handles gesture recognition logic and model inference. CSS3 and modern design frameworks ensure a clean, accessible, and user-friendly experience across different screen sizes and devices.

To facilitate gesture recognition, the system utilizes **MediaPipe** for extracting 21 hand landmarks from the webcam feed, which are then processed and classified using a trained **machine learning model**. This allows for accurate detection of American Sign Language (ASL) alphabet and potential expansion into word-level recognition. A centralized Flask server processes incoming landmark data, performs real-time predictions, and sends responses back to the React interface through **WebSocket** communication, ensuring minimal latency and smooth user interaction.

The implementation also includes:

- A live gesture prediction dashboard with visual feedback.
- A user profile and session tracking module (for future updates).

- A modular design that supports seamless integration of future features such as **multi-language support** and **adaptive learning** based on user behavior.

The **SignAlpha** platform is engineered for extensibility and performance, setting the foundation for intelligent, real-time communication support for the hearing and speech-impaired community.

5.2 OUTPUT SCREENSHOTS

Fig 5.1 shows the start of capturing of image for user's own dataset as they see fit.

Fig 5.2 illustrates what the captured dataset looks like.

Fig 5.3 shows the output of running the SignAlpha as the user makes the hand sign of alphabet A, it displays the appropriate alphabet corresponding to it

Fig 5.4 shows the output of running the SignAlpha as the user makes the hand sign of alphabet A, it displays the appropriate alphabet corresponding to it

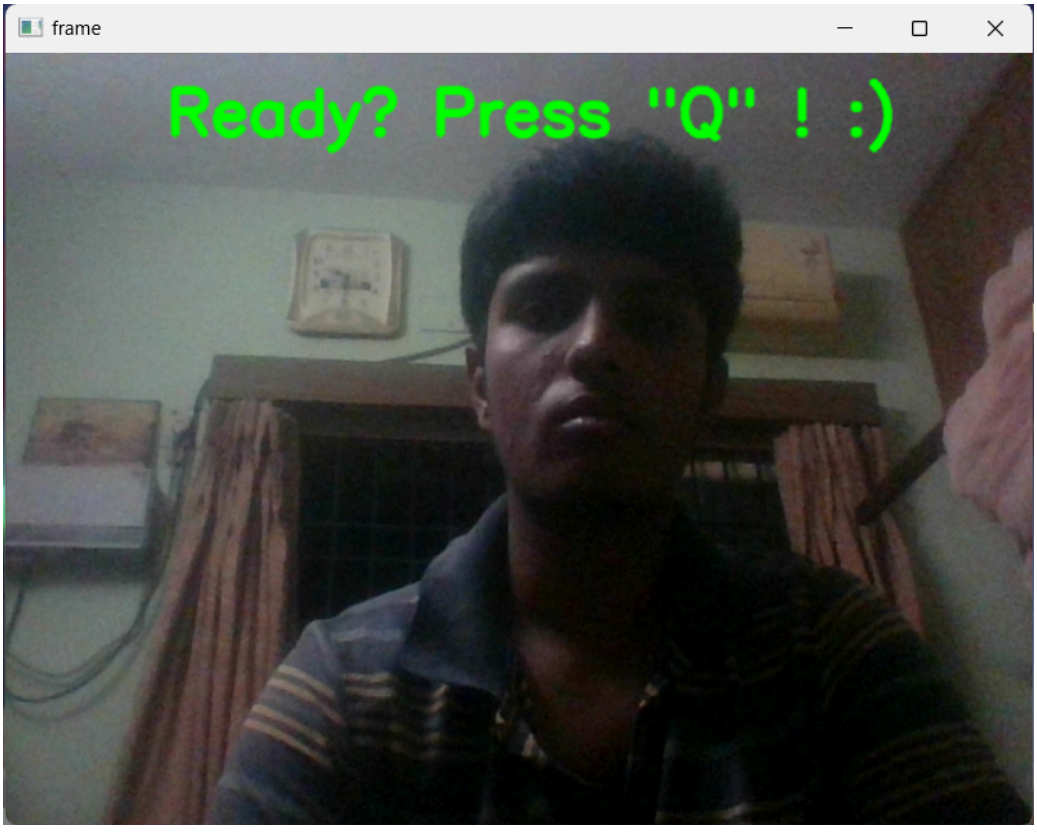

Fig 5.1 Starting of Dataset Capture

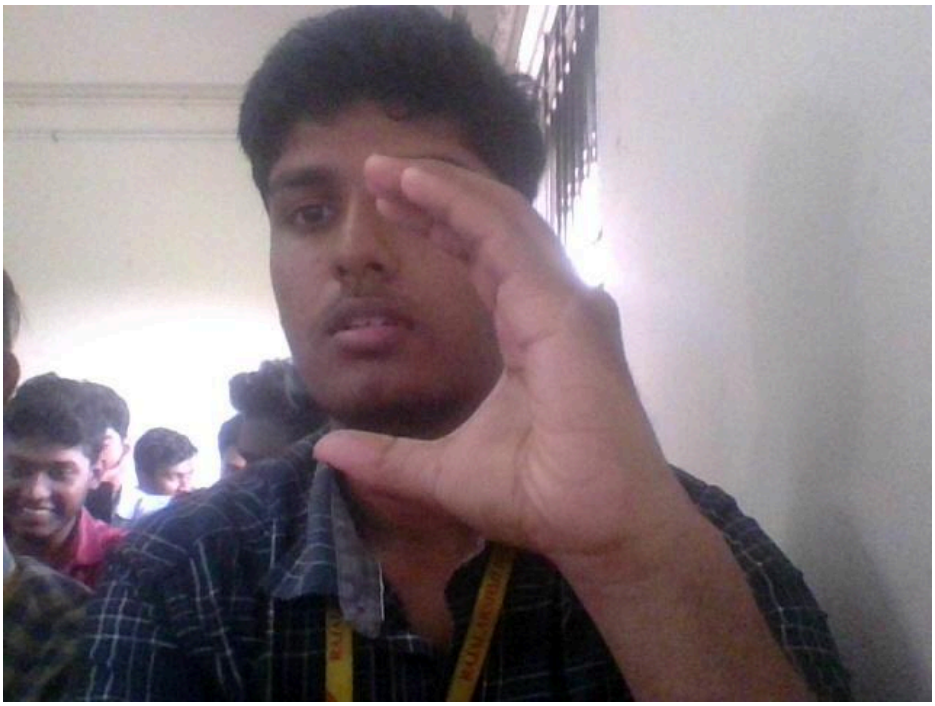

Fig 5.2 Captured dataset image

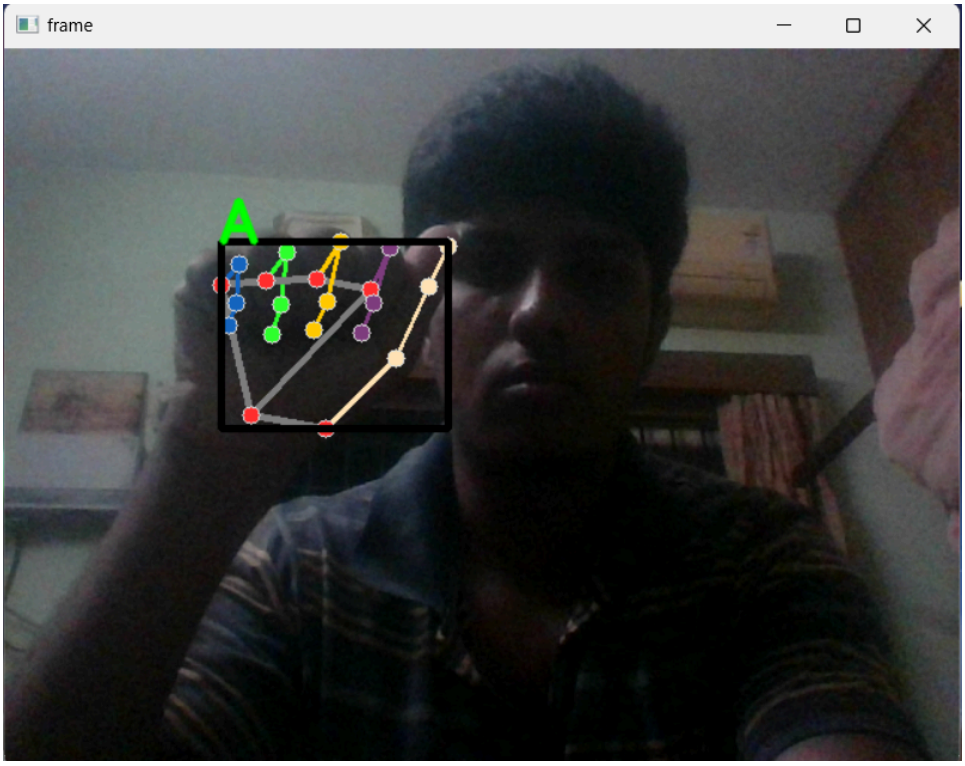

Fig 5.3 Sign Language Alphabet Detection of A

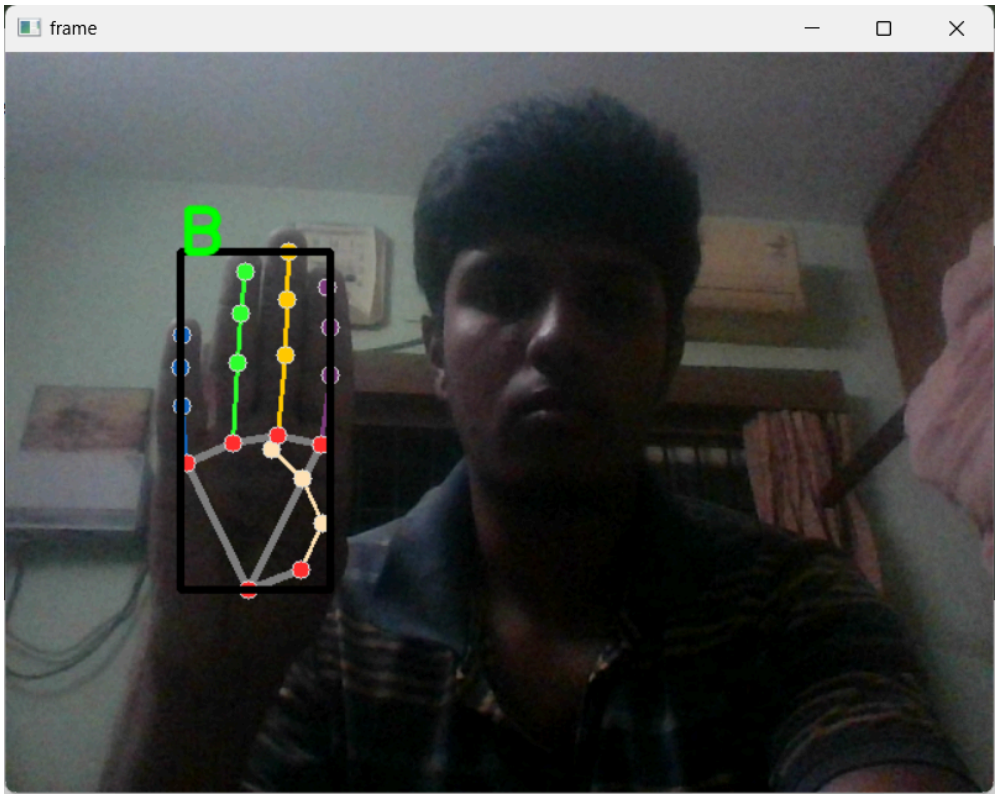

Fig 5.4 Sign Language Alphabet Detection of B

CHAPTER 6

CONCLUSION AND FUTURE ENHANCEMENT

6.1 CONCLUSION

The SignAlpha system presents an innovative and practical approach to bridging the communication gap between hearing-impaired individuals and the wider community. By leveraging computer vision, machine learning, and real-time data processing, the platform successfully recognizes American Sign Language (ASL) hand gestures and translates them into readable text. The seamless integration of a Flask-based backend, ReactJS frontend, and MediaPipe for hand landmark detection allows for a responsive and accessible user experience. SignAlpha not only promotes inclusivity but also demonstrates the potential of AI in enhancing assistive communication technologies. The project's modular design and secure architecture make it scalable and adaptable for broader use cases, such as education, customer service, and public communication.

6.2 FUTURE ENHANCEMENT

To further improve the capabilities and impact of **SignAlpha**, the following future enhancements are proposed:

1. **Word and Sentence Level Recognition**
   Extend recognition beyond individual alphabets to full words and sentences using LSTM or Transformer-based deep learning models.

2. **Gesture-to-Speech Integration**
   Enable real-time conversion of recognized signs into synthesized speech, providing instant voice feedback for seamless communication.

3. **User-Specific Learning Model**

Incorporate adaptive learning to personalize predictions based on a specific user's gesture style, improving accuracy over time.

4. **Multilingual Sign Language Support**
   Expand the system to recognize signs from other sign languages like BSL (British Sign Language) and ISL (Indian Sign Language).

5. **Mobile Application Deployment**
   Develop Android/iOS versions to make SignAlpha portable and more accessible in real-world scenarios.

6. **Offline Functionality**
   Implement offline gesture recognition by embedding the trained model on edge devices using TensorFlow Lite or ONNX.

7. **Augmented Reality (AR) Integration**
   Use AR interfaces (e.g., smart glasses) for real-time sign recognition and overlay translations in the user's field of vision.

8. **Enhanced Dataset and Model Accuracy**
   Continuously expand the gesture dataset and refine the classification model to improve prediction precision and reduce false positives.

REFERENCES

1. K. S. Sindhu, Mehnaaz, B. Nikitha, P. L. Varma and C. Uddagiri, "Sign Language Recognition and Translation Systems for Enhanced Communication for the Hearing Impaired," *2024 1st International Conference on Cognitive, Green and Ubiquitous Computing (IC-CGU)*, Bhubaneswar, India, 2024, pp. 1-6, doi: 10.1109/IC-CGU58078.2024.10530832.

2. B. A, L. J. J, C. Charan Kesava Reddy, C. A. Reddy and C. Bala Venkata Sai Rohith, "Real-Time Sign Language and Audio Conversion Using AI," 2024 International Conference on Communication, Control, and Intelligent Systems (CCIS), Mathura, India, 2024, pp. 1-6, doi: 10.1109/CCIS63231.2024.10932061.

3. G. Sharma, P. Gusain, A. Verma, H. Saini, R. Kumar and G. P. M. S, "Real-Time Sign Language Recognition and Translation Using MediaPipe and Random Forests for Inclusive Communication," *2025 2nd International Conference on Computational Intelligence, Communication Technology and Networking (CICTN)*, Ghaziabad, India, 2025, pp. 886-890, doi: 10.1109/CICTN64563.2025.10932602.

4. S. M. Antad, S. Chakrabarty, S. Bhat, S. Bisen and S. Jain, "Sign Language Translation Across Multiple Languages," *2024 International Conference on Emerging Systems and Intelligent Computing (ESIC)*, Bhubaneswar, India, 2024, pp. 741-746, doi: 10.1109/ESIC60604.2024.10481626.

5. G. L. P and R. Francis, "Sign2Text: Deep Learning-based Sign Language Translation System Using Vision Transformers and PHI-1.5B," *2024 IEEE*

International Conference on Artificial Intelligence in Engineering and

6. J. Zou, J. Li, J. Tang, Y. Huang, S. Ding and X. Xu, "Sign Language Recognition and Translation Methods Promote Sign Language Education: A Review," *2024 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, Kuching, Malaysia, 2024, pp. 3479-3484, doi: 10.1109/SMC54092.2024.10831194.

7. V. Sharma, A. Singh and S. Gaito, "Indian Sign Language recognition and translation: An Encoder-Decoder Approach," *2025 17th International Conference on Communication Systems and Networks (COMSNETS)*, Bengaluru, India, 2025, pp. 802-804, doi: 10.1109/COMSNETS63942.2025.10885623.

8. R. Thakkar, P. Kittur and A. Munshi, "Deep Learning Based Sign Language Gesture Recognition with Multilingual Translation," *2024 International Conference on Computer, Electronics, Electrical Engineering & their Applications (IC2E3)*, Srinagar Garhwal, Uttarakhand, India, 2024, pp. 1-6, doi: 10.1109/IC2E362166.2024.10826694.