A PROJECT ON

REAL-TIME SIGN LANGUAGE RECOGNITION SYSTEM USING ADVANCED DEEP LEARING

Submitted in partial fulfillment of the requirement for the award of the degree of

BACHELOR OF TECHNOLOGY

IN

COMPUTER SCIENCE & ENGINEERING

Submitted by:

Shivang Mahendra **University Roll No. 2019103**

Under the Guidance of

Dr. Narayan Chaturvedi

Associate Professor

Project Team ID: MP24CSE084 **ID No.: G39**

Department of Computer Science and Engineering
Graphic Era (Deemed to be University)
Dehradun, Uttarakhand
June-2025

DECLARATION

I, **Shivang Mahendra,** students of B.Tech (CSE), hereby declare that the project titled **Real-Time Sign Language Recognition System Using Advanced Deep Learning,** which is submitted by me to the Department of Computer Science and Engineering, Graphic Era (Deemed to be University), Dehradun, is an authentic record of my/our own work carried out during a period from **August 2024 to June 2025** under the supervision of **Dr. Narayan Chaturvedi, Associate Professor**, Department of Computer Science and Engineering, Graphic Era (Deemed to be University).

The matter presented in this project report has not been submitted by me/us for the award of any other degree at this or any other institute/university.

Name	University Roll no	Signature
Shivang Mahendra	2019103	

This is to certify that the above statement made by the candidate is correct to the best of our knowledge.

Supervisor **Head of the Department**

External Viva

Name of the Examiners: **Signature with Date**

1.

2.

CERTIFICATE

On the basis of the declaration submitted by **Shivang Mahendra**, students of B.Tech CSE, I hereby certify that the project titled **Real-Time Sign Language Recognition System Using Advanced Deep Learning,** which is submitted to the Department of Computer Science and Engineering, Graphic Era (Deemed to be University), Dehradun, in partial fulfillment of the requirement for the award of the degree of Bachelor of Technology in Computer Science and Engineering, is an original contribution with existing knowledge and a faithful record of work performed under my guidance and supervision.

To the best of my knowledge, this work has not been submitted in part or in full for any Degree or Diploma elsewhere.

Dehradun
Date:

Dr. Narayan Chaturvedi
Associate Professor
Department of Computer Science and Engineering
Graphic Era (Deemed to be University),
Dehradun, Uttarakhand, India.

ACKNOWLEDGEMENT

I take this opportunity to express my profound sense of gratitude and respect to all those who helped me throughout our project.

This report acknowledges the intense driving and technical competence of the entire individual that have contributed to it. It would have been almost impossible to complete this project without the support of these people. I extend thanks and gratitude to **Prof. (Dr.) Devesh Pratap Singh** , HOD, Department of Computer Science and Engineering, and **Dr. Narayan Chaturvedi, Associate Professor**, Department of Computer Science and Engineering, who have imparted to me guidance in all aspects. They shared their valuable time from their busy schedule to guide me and provide their active and sincere support for my activities.

This report is an authentic record of my own work, which is accomplished by the sincere and active support of all the teachers of my college. I/we have tried my best to summarize this report.

Name	University Roll no	Signature
Shivang Mahendra	2019103	

Abstract

Barriers in communication between the hearing impaired and the wider community present a considerable social issue. Sign language is an essential means of communication for individuals who are deaf or mute, yet its interpretation frequently necessitates human involvement. This study introduces a deep learning-based methodology for the automatic detection of sign language utilizing Long Short-Term Memory (LSTM) networks. Our model is developed using a specialized dataset that includes sequential hand movement data, which has been extracted and adjusted to a uniform dimensionality to maintain temporal consistency. By capitalizing on the temporal pattern recognition abilities of LSTM layers, the model adeptly learns intricate gesture sequences. The proposed system attains an accuracy rate of 94.5%, showcasing its effectiveness in classifying 26 distinct alphabetic gestures in American Sign Language (ASL). Experimental assessments indicate promising generalization capabilities on previously unseen samples, positioning it as a potential real-time assistive technology. We conclude by discussing future prospects, such as enhancing the model for dynamic sign phrases and implementing it on edge devices for practical applications

Keywords:

i. Real-time Sign Language Recognition
ii. LSTM Neural Network
iii. Hand Landmark Detection (Mediapipe)
iv. Deep Learning for Gesture Classification
v. Webcam-based Sign Language Interface

Table of Contents

List of Figures

CHAPTER 1
INTRODUCTION

1.1 Project Introduction

Communication is a fundamental human need that fosters interaction, connection, and inclusion. There are over 72 million people who are deaf or hard-of-hearing people worldwide use over 300 different sign language. However, for millions of people who are deaf or hard of hearing, effective communication is often hindered by the language gap between sign language users and non-signers. Sign language, the primary mode of communication for the hearing-impaired, is rich in gestures and expressions but is not widely understood by the general population. As a result, many individuals with hearing impairments face social isolation, difficulty in accessing services, and challenges in professional or educational settings.

With the rapid advancements in artificial intelligence (AI) and deep learning, new technological solutions are emerging to break down these communication barriers. Real-time sign language detection is one such innovation, offering the potential to convert sign language gestures into readable text or spoken words, facilitating smooth, real-time interactions between signers and non-signers. These systems rely on advanced computer vision techniques to interpret hand movements, facial expressions, and body gestures, transforming them into text or speech output.

This project aims to develop a **real-time sign language recognition system** using state-of-the-art deep learning models. By utilizing advanced AI techniques, the system will be capable of interpreting and translating sign language in real-time, providing a critical tool to foster communication between the hearing-impaired and the broader community. The system is designed with accessibility in mind, offering an intuitive interface for users, and ensuring a seamless translation experience even in diverse environments.

1.2 Problem Statement

The hearing-impaired community faces significant challenges in communicating with individuals who do not understand sign language. This language barrier can lead to feelings of exclusion and difficulties in day-to-day activities such as accessing public services, navigating professional environments, or even engaging in casual conversations. While sign language interpreters offer a solution, they are often not available in real-time or on-demand, limiting their practicality in many situations.

Currently, there is a pressing need for an automated, real-time translation system that can bridge this communication gap. Such a system should accurately recognize sign language gestures and convert them into readable text or speech. However, developing an effective system presents several challenges:

i. **Gesture Recognition Complexity**: Sign language involves intricate hand gestures, finger positions, and even facial expressions that need to be accurately captured and interpreted by the system. Traditional methods often struggle with achieving sufficient accuracy in recognizing these complex gestures.
ii. **Real-Time Performance**: For the system to be practical, it must operate in real-time, providing near-instant feedback during conversations. This requires highly efficient deep learning models that balance accuracy with speed, ensuring that the user experiences minimal latency.
iii. **Environmental Variability**: Sign language recognition systems must be robust enough to perform well in different lighting conditions, various backgrounds, and with diverse users. Variations in hand shapes, sizes, skin tones, and even clothing can impact the system's ability to correctly interpret gestures.
iv. **Multiple Sign Languages**: There are many different sign languages across the world, each with its own unique set of gestures and grammar. A scalable system should be adaptable to recognize and translate multiple sign languages, addressing regional and cultural variations.

The key objectives of the system include:

i. **Real-time Gesture Recognition**: Detecting and interpreting sign language gestures with high accuracy and speed, ensuring minimal delay in translation.
ii. **Environmental Robustness**: Ensuring that the system can operate in various conditions, such as different lighting, backgrounds, and with different users.
iii. **User Accessibility**: Designing a user-friendly interface that allows easy interaction, regardless of the user's technical expertise.
iv. **Scalability for Multiple Sign Languages**: Enabling the system to recognize and support multiple sign languages, providing a broader range of applications.

By addressing these challenges, the real-time sign language detection system will offer a scalable, efficient, and accessible solution that empowers individuals with hearing impairments. The system will enhance their ability to communicate effectively in real-world scenarios, thereby promoting inclusivity and improving social integration.

1.3 Objectives

i. Develop a Real-Time Sign Language Detection System
ii. Achieve High Gesture Recognition Accuracy
iii. Ensure Robustness in Diverse Environments
iv. Optimize for Real-Time Performance
v. Create a User-Friendly Interface
vi. Integrate with Common Hardware

CHAPTER 2
LITERATURE REVIEW

2.1 Introduction

The field of sign language detection and translation has garnered significant attention over the past few decades, especially with advancements in computer vision, machine learning, and deep learning. Several research efforts have focused on developing systems that can recognize and interpret sign language gestures, yet many challenges remain in terms of real-time performance, accuracy, and scalability. This literature survey provides an overview of the existing work, key technologies, and methodologies employed in sign language recognition systems, laying the groundwork for the proposed real-time solution.

2.2 Key Research Papers

i. Sign Language Recognition Using Handcrafted Features:

In the early stages of sign language recognition, systems primarily relied on **handcrafted features** to detect gestures. Researchers employed traditional computer vision techniques, such as **edge detection, optical flow, and histogram of oriented gradients (HOG)**, to extract features from images or video sequences.

One example is the work by **Starner et al. (1997),** which used **Hidden Markov Models (HMMs)** to recognize American Sign Language (ASL) in real-time using wearable computers. This system, however, was limited by the complexity of extracting features from raw data and required significant pre-processing to identify gestures accurately. The handcrafted feature-based approach had limitations in adapting to variations in lighting, background, and hand shapes, which reduced its applicability in real-world settings. [1]

ii. Advancements in Machine Learning-Based Gesture Recognition:

As machine learning algorithms matured, sign language recognition systems began incorporating more robust models, such as **Support Vector Machines (SVMs)** and **Random Forests**, which provided better accuracy in gesture classification. Researchers like **Fang et al. (2004)** explored the use of SVMs for Chinese Sign Language recognition. The model was trained on spatial features of hand gestures,

yielding promising results for small vocabularies but faced scalability issues when extending to larger datasets.

While these systems improved gesture recognition accuracy, they still required manual feature extraction and did not generalize well to different environments, limiting their real-time applicability. [2]

iii. Deep Learning for Sign Language Recognition:

With the rise of deep learning, sign language recognition systems have seen dramatic improvements in both accuracy and scalability. **Convolutional Neural Networks (CNNs)** have been highly effective in feature extraction, allowing models to automatically learn complex patterns in hand gestures without the need for manual feature engineering. **Simonyan and Zisserman (2014)** introduced the use of deep CNNs for image classification, which later inspired researchers to apply CNNs for hand gesture recognition in sign language.

Pigou et al. (2014) developed a sign language recognition system using CNNs to classify hand gestures in video sequences. Their approach successfully recognized isolated gestures but struggled with continuous sign language recognition, where gestures overlap or are performed sequentially in real conversations. [3]

iv. Recurrent Neural Networks (RNNs) and Sequential Gesture Recognition:

Recognizing that sign language is temporal and sequential, researchers began exploring **Recurrent Neural Networks (RNNs)** and **Long Short-Term Memory (LSTM)** networks for gesture interpretation. LSTMs excel at capturing dependencies over time, making them suitable for continuous gesture recognition.

Koller et al. (2016) employed a combination of CNNs and LSTMs for real-time continuous sign language recognition. Their system processed video frames through CNNs for spatial feature extraction and used LSTMs to model temporal dependencies between gestures. This approach achieved higher accuracy in recognizing continuous sequences of signs compared to traditional methods, but the real-time performance was still limited by the computational complexity of LSTMs. [4]

v. Multi-Modal Approaches and Sensor Fusion:

Another advancement in sign language recognition involves the use of **multi-modal** data, combining visual inputs with additional sensors to improve recognition

accuracy. For instance, **Kinect sensors and Leap Motion controllers** have been used to capture 3D spatial data, including hand depth and motion trajectories, which can enhance recognition models.

Neverova et al. (2015) proposed a multi-modal system that fused **RGB video, depth data**, and **skeletal information** from Kinect sensors to improve gesture recognition accuracy. While these systems produced more reliable results, the reliance on specialized hardware like Kinect or Leap Motion limited their accessibility for widespread real-world use.[5]

vi. Recent Work and State-of-the-Art Techniques:

Recent work in the field has focused on improving the real-time performance and scalability of sign language recognition systems. **3D CNNs** have been introduced to capture spatio-temporal features more effectively, and **transformer networks** have emerged as an alternative to LSTMs for sequence modelling.

In 2021, **Zhang et al.** proposed a transformer-based model that improved the efficiency of sign language translation by reducing the reliance on recurrent structures. This approach demonstrated state-of-the-art results in continuous sign language recognition, further advancing the real-time capabilities of these systems. [6]

2.3 Conclusion:

The development of real-time sign language recognition systems has progressed significantly, with deep learning playing a pivotal role in improving accuracy and scalability. However, challenges remain in terms of real-time performance, environmental robustness, and scalability across different sign languages. This literature survey highlights the evolution of sign language recognition from handcrafted feature extraction to the application of cutting-edge deep learning models like CNNs, LSTMs, and transformers. The proposed system aims to build on these advancements, leveraging deep learning to create an efficient, accessible, and real-time solution for sign language translation.

CHAPTER 3

PROPOSED METHODOLOGY

The proposed methodology for the Sign Language Detection System involves integrating computer vision techniques, deep learning models, and a real-time user interface to recognize and convert sign language gestures into meaningful text. The system is designed to function in real time using a webcam, enabling intuitive communication for individuals with hearing or speech impairments.

3.1 Data Acquisition and Preprocessing

3.1.1 Hand Landmark Extraction:

The system uses **Mediapipe Hands** to detect and extract 21 key landmarks from the user's hand in real-time webcam input. Each landmark consists of 3D coordinates (x, y, z), resulting in a 63-dimensional feature vector.

3.1.2 Data Formatting:

Each extracted hand gesture is padded or truncated to maintain a consistent input shape, ensuring compatibility with the LSTM model.

3.1.3 Normalization:

The landmark values are optionally normalized to reduce variability caused by hand distance or screen resolution.

3.2 Model Training and Evaluation

3.2.1 Label Encoding:

The labels corresponding to hand gestures (A–Z) are encoded using Scikit-learn's LabelEncoder for numerical classification.

3.2.2 Model Architecture (LSTM):

The deep learning model consists of:

- Two LSTM layers for temporal feature learning.
- Dropout layers for regularization.
- Dense layers for classification with softmax activation.

3.2.3 Training:

The model is trained on a curated dataset of hand signs, with data split into training and testing sets. Categorical cross-entropy is used as the loss function with the Adam optimizer.

3.2.4 Evaluation:

Performance is evaluated using accuracy metrics and confusion matrices to measure how well the model distinguishes between different signs.

3.3 Real-Time Prediction Pipeline

3.3.1 Frame Capture:

The frontend continuously captures video frames at intervals (e.g., 1.5 seconds) using the webcam.

3.3.2 Prediction Request:

Captured frames are converted to Base64 and sent via POST request to a Flask backend endpoint (`/predict`).

3.3.3 Model Inference:

The backend extracts landmarks from the image, reshapes the data, and feeds it into the pre-trained LSTM model. The predicted class index is mapped back to its corresponding character using the saved LabelEncoder.

3.3.4 Sentence Formation:

Detected characters are appended to an ongoing sentence, with user-controlled options to add space, delete the last character, or clear the full sentence.

3.4 User Interface Design

A responsive and user-friendly web interface is developed using **HTML**, **CSS**, and **JavaScript**. Interface Features:

- **Webcam Preview** (left side): Displays live feed for gesture input.
- **Controls** (right side): Start/Pause/Resume, Space, Delete, Clear Sentence.
- **Output**: Displays the latest detected character and the formed sentence.

3.5 System Workflow

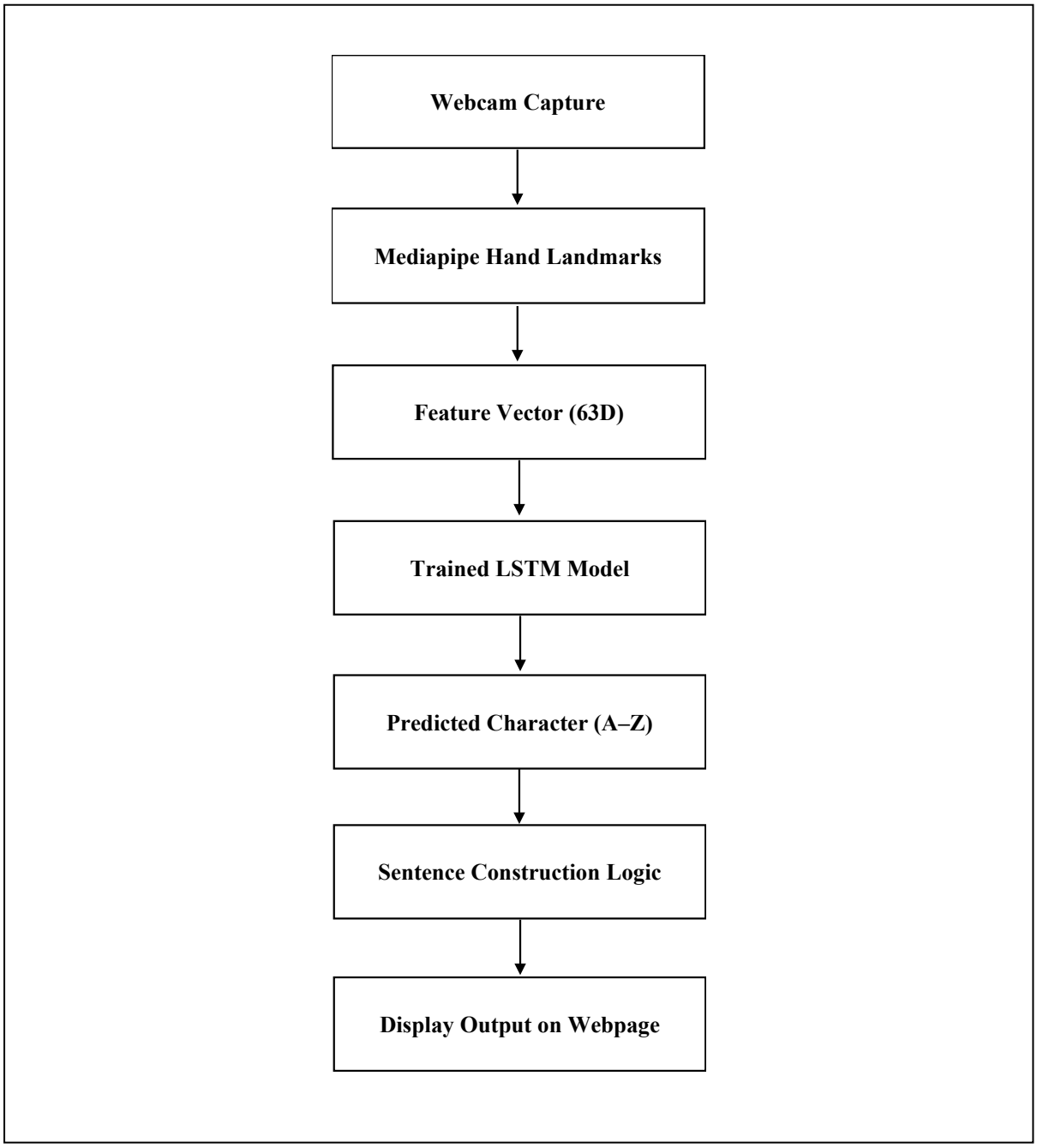

Fig 3.1: System Flowchart

3.6 Deployment and Integration

The system is locally hosted using Flask for testing purposes.Model and encoder files (.h5 and .pkl) are loaded at runtime for real-time inference.The application can be further deployed on cloud platforms or converted into a desktop/mobile application in future work.

CHAPTER 4

PROJECT DESIGN AND TESTING

This chapter discusses the structural design of the system, its modular implementation, and the testing strategies used to evaluate the functionality and accuracy of the proposed system. Each component is designed to be modular and reusable to enhance maintainability and scalability.

4.1 System Architecture

The project follows a client-server architecture comprising the following major components:

- Frontend (Client-side): A responsive web interface developed using HTML, CSS, and JavaScript that captures webcam frames and sends them to the backend for prediction.
- Backend (Server-side): A Flask-based Python server that handles HTTP requests, processes incoming image data, and returns predictions using the trained LSTM model.
- Model: A trained deep learning model (LSTM) that performs classification on preprocessed hand landmark data to predict characters.
- Data Processing Module: Utilizes Mediapipe to extract 3D hand landmarks and convert them into a format suitable for the LSTM model.

4.2 Module-wise Design

- Webcam Interface: Captures real-time video and converts frames to base64 format.
- Feature Extraction: Uses Mediapipe to extract (x, y, z) coordinates of 21 hand landmarks.
- Prediction Module: Receives landmark vectors, reshapes them, and performs prediction using the trained LSTM model.
- Sentence Builder: Constructs meaningful sentences from recognized characters. Includes options like space, delete, and clear.
- User Interface Controls: Toggle buttons to start/pause/resume capturing and manage text.

4.3 User Interface Design

The interface is designed to be:

- Simple: Easy to understand for users with minimal technical knowledge.
- Interactive: Includes live preview, prediction display, and control buttons.
- Responsive: Compatible with various screen sizes and browsers.

UI Features:

- Camera feed (left side)
- Start/Pause/Resume toggle
- Delete Last and Add Space buttons
- Sentence and Detected Character display (right side)
- Modern colored background for aesthetics

4.4 Testing Strategy

To ensure system reliability and performance, the following testing strategies were applied:

4.4.1 Unit Testing

Each function (e.g., image preprocessing, landmark extraction, model prediction) was tested individually to verify expected behavior.

4.4.2 Integration Testing

Tested end-to-end flow from webcam capture → base64 encoding → backend prediction → display output.

Ensured seamless communication between frontend and backend.

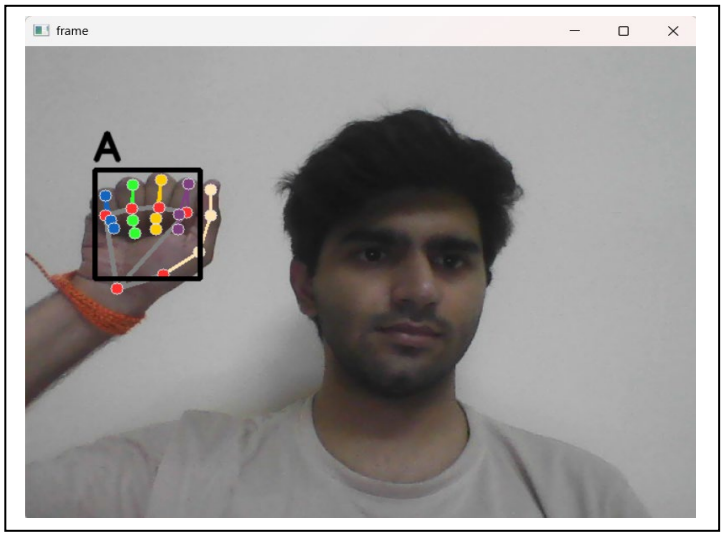

Fig 4.1: Integration Testing

4.4.3 Functional Testing

Validated user interactions (button clicks, live feed control).

Checked accurate sentence construction and button response.

4.4.4 Model Evaluation

Accuracy: Achieved **~94.75%** on test data.

Evaluated using:

a. Confusion Matrix

b. Classification Report

c. Training and validation curves

Verified with real-time gestures during live testing.

4.4.5 Cross-browser Testing

Tested UI on:

Google Chrome

Mozilla Firefox

Microsoft Edge

Fig 4.2: UI Testing

CHAPTER 5

RESULT AND DISCUSSION

5.1 Model Performance

The deep learning model developed for sign language recognition was trained using hand landmark data extracted via Mediapipe. The final model, based on Long Short-Term Memory (LSTM) networks, achieved strong performance on the test dataset.

Evaluation Metrics:

- **Accuracy**: The trained model achieved an accuracy of 94.75% on the test data.

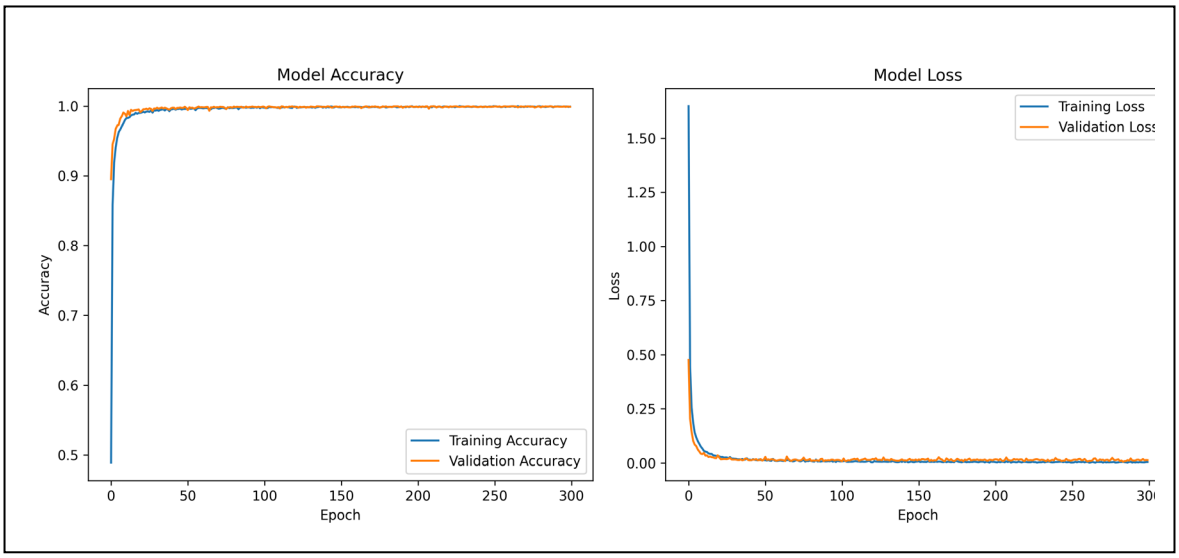

Fig 5.1: Model Accuracy and Model Loss Graph

- **Confusion Matrix**: The matrix showed that the model was highly accurate for most alphabets, with occasional misclassifications between visually similar signs (e.g., 'M' and 'N').
- **Classification Report**:
  - Precision: High precision scores across most classes indicate that false positives are minimal.
  - Recall: Demonstrates good generalization on unseen test data.
  - F1-score: Balanced across classes, confirming robustness of the model.

5.2 User Interface Outcomes

The web-based interface was tested extensively for real-time sign prediction, sentence construction, and user interactions.

Observations:

- **Real-Time Detection**:
  - Smooth real-time character prediction with minimal latency.
  - Mediapipe provided stable landmark tracking even during minor hand movements.
- **Sentence Building**:
  - Detected characters were appended to form complete words and sentences.
  - Additional functionalities like *Add Space*, *Delete Last*, and *Clear Sentence* were intuitive and useful.
- **User Controls**:
  - A toggle button for *Start*, *Pause*, and *Resume* capturing streamlined interaction.
  - Sentence control buttons were placed conveniently and functioned as expected.
- **UI Design**:
  - The UI was made visually appealing with a modern background and clear layout.
  - Layout positioning (camera on the left, buttons and output on the right) improved usability.

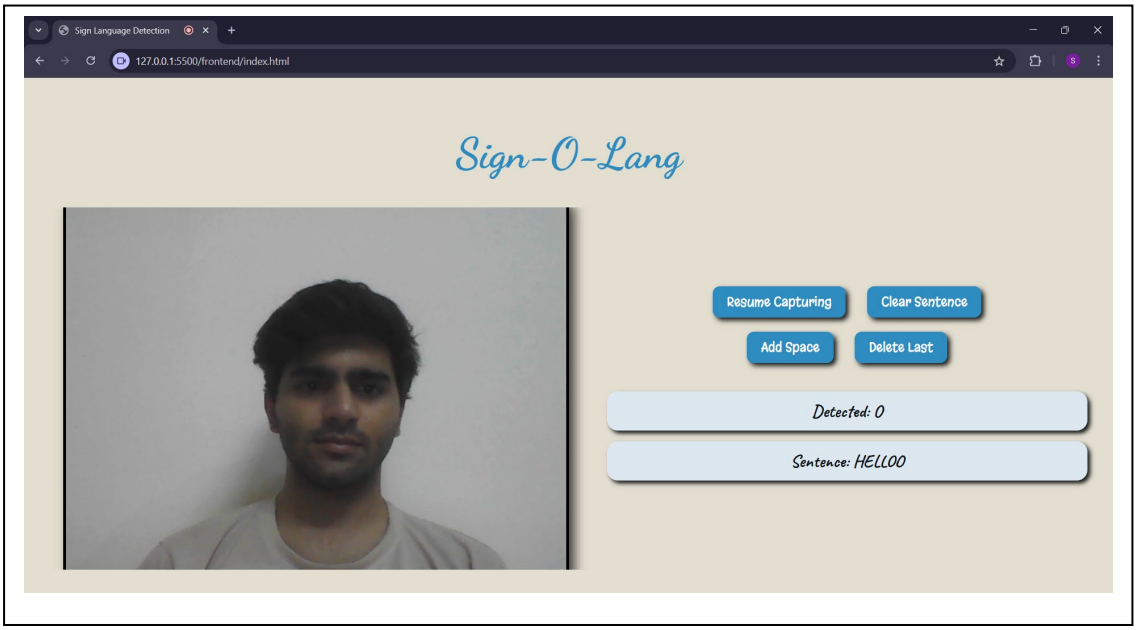

Fig 5.2: User Interface

5.3 Real-Time Testing and Behaviour

The system was tested under various real-time conditions to validate its robustness:

Condition	**Result**
Good Lighting	High accuracy, stable hand tracking
Moderate Lighting	Slight fluctuation in prediction, but still acceptable
Hand Partially Visible	Prediction failed or was inaccurate, as expected
Background Movement	Mediapipe handled dynamic backgrounds effectively
Fast Hand Movement	Reduced accuracy; required user to maintain steady pose for prediction

5.4 Key Findings

- The model performs well in recognizing static alphabet gestures.
- Real-time prediction is efficient and responsive.
- Mediapipe significantly improves landmark extraction accuracy compared to raw pixel based methods.
- Usability of the interface was rated positively during peer testing.
- Combining detection and sentence construction enhances communication effectiveness for the hearing- and speech-impaired community.

5.5 Limitations

- The system currently supports only static gestures and does not cover dynamic signs or words.
- Limited to single-hand detection.
- Accuracy may drop with poor lighting or occluded hand positions.

CHAPTER 6

CONCLUSION AND FUTURE SCOPE

6.1 Conclusion

The Real-Time Sign Language Detection System successfully addresses the communication barrier between the hearing/speech-impaired community and the general population by recognizing static hand gestures in real time and converting them into readable text. Using a combination of Mediapipe for hand landmark extraction and an LSTM-based deep learning model, the system accurately identifies sign language alphabets and builds sentences dynamically.

The application's intuitive and modern user interface allows users to start, pause, resume, and control sentence formation easily, making it practical for real-world scenarios. The system demonstrates robust performance with a high recognition accuracy of 94.75%, providing a reliable tool for gesture-based communication.

Key Contributions:

- Accurate prediction of static sign language alphabets using LSTM.
- Real-time hand tracking and gesture capture via Mediapipe.
- Sentence-building interface with options like space, delete, and clear.
- A responsive and user-friendly web interface.

6.2 Future Scope

Despite its success, the project leaves room for further development and improvement. Some areas of potential enhancement include:

- **Support for Dynamic Gestures:** Extend the system to recognize continuous gesture sequences and signs involving motion (e.g., "thank you", "hello").
- **Two-Hand Detection:** Enhance Mediapipe integration to recognize gestures involving both hands, which are common in sign language.
- **Real-Time Voice Output:** Integrate text-to-speech (TTS) to convert recognized sentences into speech, making the system a real-time interpreter.
- **Mobile Application Development:** Port the system to Android/iOS platforms for greater accessibility and ease of use on-the-go.
- **Multilingual Support:** Enable translation of recognized sign language into multiple regional or international languages.

- **Improved Dataset:** Collect a more diverse and extensive dataset to improve model generalization across different hand sizes, orientations, and lighting conditions.
- **Offline Functionality:** Allow the application to function without internet dependency by bundling models and resources locally.

Details of Research Publication

The details of our research publication are as follows:

1. Shivang Mahendra, “Real Time sign Language Recognition Using Seep Learning”, AIBlock 2025: The 7th International Conference on Application Intelligence and Blockchain Security, Beijing, China, July 2025 **(submitted)**
2. Shivang Mahendra, “Real Time sign Language Recognition Using Seep Learning”, RT-Cloud 2025 : Fourth International Workshop on Real-Time and Cyber-Physical Cloud, Brussels, Belgium, July 2025 **(submitted)**
3. Shivang Mahendra, “Real Time sign Language Recognition Using Seep Learning”, RET 2025: The 8th International Conference on Research in Engineering and Technology, Tra Vinh, Viet Nam, June 2025 **(submitted)**

APPENDIX

1. Code listing

```
cap = cv2.VideoCapture(0)
for j in range(number_of_classes):
    if not os.path.exists(os.path.join(DATA_DIR, str(j))):
        os.makedirs(os.path.join(DATA_DIR, str(j)))

    print('Collecting data for class {}'.format(j))

    done = False
    while True:
        ret, frame = cap.read()
        if not ret:
            print("Failed to grab frame.")
            break

        cv2.putText(frame, text: 'Ready? Press "Q" ! :)', org: (100, 50), cv2.FONT_HERS
                    cv2.LINE_AA)
        cv2.imshow( winname: 'frame', frame)
        if cv2.waitKey(25) == ord('q'):
            break

    counter = 0
    while counter < dataset_size:
        ret, frame = cap.read()
        cv2.imshow( winname: 'frame', frame)
        cv2.waitKey(25)
        cv2.imwrite(os.path.join(DATA_DIR, str(j), '{}.jpg'.format(counter)), frame)
```

Fig 7: Collecting Images

```
img = cv2.imread(os.path.join(DATA_DIRECTORY, dir_, img_path))
img_rgb = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)

results = hands.process(img_rgb)
if results.multi_hand_landmarks:
    for hand_landmarks in results.multi_hand_landmarks:
        for i in range(len(hand_landmarks.landmark)):
            x = hand_landmarks.landmark[i].x
            y = hand_landmarks.landmark[i].y
            z = hand_landmarks.landmark[i].z

            x_.append(x)
            y_.append(y)
            z_.append(z)

        for i in range(len(hand_landmarks.landmark)):
            x = hand_landmarks.landmark[i].x
            y = hand_landmarks.landmark[i].y
            z = hand_landmarks.landmark[i].z

            data_aux.append(x - min(x_))
            data_aux.append(y - min(y_))
            data_aux.append(z - min(z_))
```

Fig 8: Extracting Hand Landmarks

```
# LSTM model building
num_classes = y_categorical.shape[1]
sequence_length = 1
feature_dimensions = 63

model = Sequential([
    LSTM(128, return_sequences=True, input_shape=(sequence_length, feature_dimensions)),
    Dropout(0.3),
    LSTM(64),
    Dropout(0.3),
    Dense(64, activation='relu'),
    Dense(num_classes, activation='softmax')
])

model.compile(loss='categorical_crossentropy', optimizer='adam', metrics=['accuracy'])
model.summary()
```

Fig 9: LSTM Model Building

2. Dataset samples

Fig 10: "W" Data Sample

Fig 11: "K" Data Sample

References

[1] Visual Recognition of American Sign Language Using Hidden Markov Models https://sites.cc.gatech.edu/home/thad/p/031_10_SL/visual-recognition-of-asl-using-hmm-95.pdf

[2] 3D Human gesture capturing and recognition by the IMMU-based data glove https://www.sciencedirect.com/science/article/abs/pii/S0925231217314054

[3] Sign Language Recognition Using Convolutional Neural Networks https://link.springer.com/chapter/10.1007/978-3-319-16178-5_40

[4] Deep Hand : How to Train a CNN on 1Million Hand Images When Your Data Is Continuous and Weakly Labelled https://openaccess.thecvf.com/content_cvpr_2016/papers/Koller_Deep_Hand_How_CVPR_2016_paper.pdf

[5] Hand Gesture Recognition with 3D Convolutional Neural Networks https://research.nvidia.com/sites/default/files/pubs/2015-06_Hand-Gesture-Recognition/CVPRW2015-3DCNN.pdf

[6] Multimodal Fusion Framework Based on Statistical Attention and Contrastive Attention for Sign Language Recognition https://ieeexplore.ieee.org/abstract/document/10013765

[7] An intelligent sign language learning and promotion station system using artificial intelligence and computer vision https://aircconline.com/csit/papers/vol14/csit140814.pdf

[8] Sign language recognition system using machine learning https://www.ijirmps.org/papers/2023/6/230387.pdf