SIGNSYNC - SIGN LANGUAGE INTERPRETER FOR INCLUSIVE COMMUNICATION

CS6611 – CREATIVE AND INNOVATIVE PROJECT

Submitted by

Sangeetha S (2022103023)

Dhivya S (2022103029)

ArunKumar D (2022103001)

Venizha R (2022103530)

in partial fulfillment of the requirements for the award of the degree of

BACHELOR OF ENGINEERING

in

COMPUTER SCIENCE AND ENGINEERING

College of Engineering, Guindy

ANNA UNIVERSITY, CHENNAI - 600 025

APRIL 2025

ANNA UNIVERSITY, CHENNAI - 600 025

BONAFIDE CERTIFICATE

Certificate that this project request titled **SIGNSYNC - SIGN LANGUAGE INTERPRETER FOR INCLUSIVE COMMUNICATION** is the bonafide work of **Sangeetha S (2022103023), Dhivya S (2022103029), ArunKumar D (2022103001), Venizha R (2022103530)** who carried out the project work under my supervision, for the fulfillment of the requirements as part of the **CS6611 – Creative and Innovative Project**.

Dr. V. MARY ANITA RAJAM	**Dr. P.UMA MAHESWARI**	**Dr. P. GANESH KUMAR**
Professor & Head	**Professor**	**Associate Professor**
HEAD OF THE DEPARTMENT	**SUPERVISOR**	**SUPERVISOR**
Department of Computer Science and Engineering, Anna University, Chennai – 600025.	Department of Computer Science and Engineering, Anna University, Chennai – 600025.	Department of Computer Science and Engineering, Anna University, Chennai – 600025.

ABSTRACT

SignSync is a bidirectional communication system developed to bridge the gap between Indian Sign Language (ISL) users and non-signers, thereby enhancing accessibility for India's deaf and hard-of-hearing community. Leveraging advanced machine learning and computer vision technologies, SignSync features two core modules: **ISL to Text-and-Speech** and **Speech-to-Text-and-ISL**.

The ISL to Text-and-Speech module utilizes MobileNetV2, an efficient convolutional neural network, to recognize ISL gestures from video input. Trained on a custom dataset of 10,000 images representing 50 distinct gesture types, the model achieves a training accuracy of 99.60% and a test accuracy of 99.70%. Recognized gestures are translated into both text and synthesized speech, facilitating real-time communication for non-signers.

Conversely, the Speech-to-Text-and-ISL module employs the Web Speech Recognition API to transcribe spoken language into text, which is then mapped to a corresponding set of ISL gesture GIFs or letter images for visual representation. This enables non-signers to communicate effectively with ISL users through spoken input.

In addition to real-time translation, SignSync incorporates features such as image-based gesture recognition, live webcam input, and an educational interface that supports interactive ISL learning. Designed for accessibility and ease of use, the platform offers a seamless user experience while demonstrating robust performance across various use cases.

ACKNOWLEDGEMENT

Foremost, we would like to express our sincere gratitude to our project guide, **Dr. P. UMA MAHESWARI**, Professor and **Dr. P. GANESH KUMAR**, Associate Professor, Department of Computer Science and Engineering, College of Engineering Guindy, Chennai for their constant source of inspiration. We thank them for the continuous support and guidance which was instrumental in taking the project to successful completion.

We are grateful to **Dr. V. MARY ANITA RAJAM**, Professor and Head, Department of Computer Science and Engineering, College of Engineering Guindy, Chennai for her support and for providing necessary facilities to carry out for the project.

We would also like to thank our friends and family for their encouragement and continued support. We would also like to thank the Almighty for giving us the moral strength to accomplish our task.

Sangeetha S **Dhivya S** **ArunKumar D** **Venizha R**

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER 1
INTRODUCTION

Communication is an essential human requirement, but millions of deaf and hearing-impaired people face barriers in communicating with society at large. In India, Indian Sign Language (ISL) is used by most deaf individuals to communicate. Despite its critical importance, ISL remains inadequately acknowledged in education across all levels and in public services.

Due to this gap, deaf people often have limited opportunities and thus reduced access to education, employment, or social integration. **Sign language** facilitates communication among deaf individuals at their own level, but social exclusion caused by non-signers, due to lack of knowledge and sensitivity, contributes to social isolation and misunderstandings.

1.1. OBJECTIVE

The main purpose of this project is to create SignSync—a user-centered, web-based platform that is intended to:

1. Enable real-time **ISL to text and speech** translation using computer vision technology, allowing people unfamiliar with sign language to understand signed conversations instantly.
2. Provide **speech-to-ISL translation**, helping individuals who are deaf or hard of hearing understand spoken language—even in challenging environments like public spaces or long-distance communication.
3. Provide an interactive **ISL learning module** to foster self-directed learning and awareness.
4. The purpose is to empower the deaf community, promote learning ISL, and foster an inclusive environment between **signers and non-signers**.

1.2. PROBLEM STATEMENT

Deaf, Hard-of-Hearing, and speech-impaired individuals rely on sign language to communicate, but many people are unable to understand it, leading to significant issues. In **healthcare settings**, this results in misdiagnoses, delayed treatments, and medical errors. In **legal settings**, these individuals struggle to fully participate in trials, impacting their access to justice. Additionally, the lack of **real-time speech-to-sign language conversion** creates communication and inclusivity challenges during government functions and public events. These barriers highlight the urgent need for effective communication solutions that promote accessibility and equal participation.

Current systems usually focus on only one direction of communication—either converting sign language to **text or speech**, or converting speech to sign language. These solutions are often not real-time, lack accuracy, and do not provide any support for learning sign language.

There is a clear need for a single, integrated platform that can support both directions of **communication in real time**, while also helping users learn sign language. Such a solution would improve accessibility, promote inclusion, and make communication easier between signers and non-signers.

1.3. NEED FOR THE SYSTEM

There is a growing demand for a solution that not only translates sign language into text and speech but also converts spoken language into sign language—all within a single, accessible platform. Existing tools are limited in scope, often focusing on one-way communication or lacking real-time functionality. Furthermore, most platforms do not include educational support to help non-signers learn sign language, which is key to building a more inclusive environment.

SignSync is designed to meet this need by offering a comprehensive, real-

time, and user-friendly system that supports two-way communication and sign language learning. By integrating multiple features into one platform, SignSync empowers both signers and non-signers, promotes accessibility, and helps bridge the communication gap across various sectors such as healthcare, education, public services, and daily interactions.

1.4. CHALLENGES IN THE SYSTEM

While SignSync offers a comprehensive solution for bridging the communication gap between sign language users and non-signers, several challenges were encountered during its development and implementation:

1. **Limited Dataset Availability:** There is a lack of large, diverse, and publicly available Indian Sign Language (ISL) datasets. Creating a custom dataset was necessary, which required significant time and effort to ensure adequate representation of different gestures under varied conditions.

2. **Gesture Similarity and Ambiguity:** Many ISL gestures are visually similar, making it difficult for the model to distinguish between certain signs, especially when performed quickly or with slight variations by different users.

3. **Environmental Sensitivity:** Real-time sign recognition is sensitive to lighting, background noise, and camera positioning. Variations in these factors can affect the accuracy and reliability of gesture detection and prediction.

1.5. SCOPE OF THE PROJECT

The SignSync project aims to develop a web-based application that facilitates sign language communication and learning through real-time sign-to-text translation, speech-to-sign conversion, and an interactive educational

module. Its primary scope includes:

Core Functionalities:

1. **Real-Time Sign-to-Text and Speech:** Detect hand gestures via webcam using MediaPipe and a pre-trained model, display predicted signs as text, and vocalize them using the Web Speech API.
2. **Speech-to-Sign Translation:** Capture spoken input via the Web Speech API, convert it to text, and display corresponding sign language GIFs or letter images sourced from static resources.
3. **Learn Signs Module:** Provide an interactive interface for users to explore and learn sign language gestures through a grid of clickable signs, showing GIFs or letter images in a popup.
4. **Image Upload for Sign Prediction:** Allow users to upload images for static sign recognition, enhancing accessibility for non-real-time use cases.

Target Audience:

- **Non-Signers in Critical Fields:** Medical professionals (doctors, nurses) can use this platform for real-time sign-to-text and speech to understand deaf patients' signs and speech-to-sign to convey instructions visually; other professionals (educators, customer service) communicate with signers.
- **Signers:** Deaf or hearing-impaired individuals benefit from speech-to-sign translation at public events (e.g., conferences, lectures).
- **Learners & Enthusiasts:** Users learn and practice sign language gestures.
- **General Users:** Those interested in assistive communication technologies.

CHAPTER 2
LITERATURE SURVEY

2.1 LITERATURE REVIEW

This chapter explores the current research and innovations published in the domain of sign language recognition, real-time translation systems, and assistive technologies that are proposed to be incorporated in our Sign Language Interpreter project.

R. Brindha et al. [1] explored deep learning techniques for Indian Sign Language (ISL) recognition, using CNNs to identify and classify ISL gestures from video inputs, trained on a custom dataset. Their system converts gestures into text and speech, helping bridge communication between signers and non-signers. While promising for accessibility in areas like education and healthcare, the study notes challenges in handling variations in signing styles and environments.

V. Madhusudhana Reddy et al. [2] worked on speech-to-text and text-to-speech translation using deep learning. They combined MobileNetV2 and MediaPipe Hands for real-time gesture recognition focused on hand landmark detection. Although the system achieved high accuracy, adapting it to diverse lighting, hand shapes, and regional variations remains a hurdle. Their study highlights the need for robust vision systems for smooth sign language translation.

Kohsheen Tiku et al. [3] introduced a real-time sign language-to-text and speech converter, driven by deep learning. Prioritizing real-time performance, the system aims to lessen dependence on human interpreters at dynamic events like public gatherings. Their findings underline the necessity for broad vocabulary coverage and precise gesture detection to make such systems practical for everyday conversations.

Nihashree Sarma et al. [4] built a real-time ISL recognition system using YOLOv3, known for its object detection capabilities. Their system effectively processes video sequences to classify ISL gestures, performing well under controlled conditions. The study highlights the potential of using object detection models for sign language, although scalability across varied datasets and real-world conditions needs further research.

Jashwanth Peguda et al. [5] developed a system to translate speech into ISL gestures, shown through animations or visuals. This tool fosters inclusivity, allowing non-signers to engage with the Deaf community. However, building a comprehensive gesture database across different Indian languages remains a significant task for broader implementation.

Sruthi and Lijiya [6] proposed "Signet," a CNN-based ISL recognition system that processes video inputs for sign classification. Their work emphasizes the importance of large, diverse datasets to improve model generalization for different ISL variations, helping push forward scalable real-world applications.

Kishore et al. [7] investigated a video-audio interface for ISL gesture recognition, merging visual and auditory data to boost classification accuracy. Their system works well in controlled environments, but optimizing it for real-time adaptability remains a work in progress.

Redmon and Farhadi [8] introduced YOLOv3, which has since become a staple in real-time gesture recognition systems for its speed and accuracy, especially on devices with limited resources.

Howard et al. [9] created MobileNetV2, a lightweight CNN architecture perfect for mobile and embedded devices. Its use in gesture recognition systems shows that efficiency doesn’t have to sacrifice accuracy, making it a valuable option for accessible, low-cost interpreters.

Alawwad et al. [10] applied Faster R-CNN to Arabic Sign Language

recognition, achieving high accuracy. Their study reveals that while region-based CNNs adapt well to different sign languages, real-time application remains a challenge due to computational complexity.

Anjana Devi [11] developed a real-time ISL recognition system focused on aiding the physically challenged, translating video inputs into text. The study stresses the importance of low-latency processing to ensure smooth, real-world communication.

Sneha Varsha et al. [12] introduced "Sign Communication" an ISL recognition system using deep learning, demonstrating high accuracy in controlled environments and highlighting its potential for educational and public settings.

Lalitha [13] designed a machine learning-based framework for video-based sign language interpretation, extracting temporal and spatial gesture features to generate text. While effective under controlled conditions, real-time optimization is necessary for practical deployment.

Liao et al. [14] developed a dynamic sign language recognition system using BLSTM-3D residual networks, which better capture the flow of signing motions in videos. Though powerful, the system demands significant computational resources.

Mangairkarasi et al. [15] created a CNN-based ISL recognition system achieving high accuracy with a custom dataset. Their work emphasizes that diverse training data is crucial for models to adapt to regional ISL differences.

Priyanka and Sakthi Prabha [16] built a hand gesture-to-speech translation system for the speech-impaired, enhancing communication accessibility. Multimodal approaches show great promise, although improving real-time responsiveness remains a key challenge.

Najib [17] proposed a multilingual sign language recognition system using machine learning, aiming to support several sign languages. However,

expanding to diverse vocabularies demands substantial dataset development.

Kishore and Kumar [18] presented a system for segmenting, tracking, and recognizing ISL gestures, converting them into voice and text outputs. Although their multi-stage pipeline boosts accuracy, it also increases system complexity.

Nandy et al. [19] explored ISL gesture recognition for human-robot interaction, using machine learning to help robots interpret sign language. While promising for assistive robotics, real-time processing still needs improvement.

Azhar [20] introduced a bidirectional translation system for Yemeni Sign Language using fuzzy logic and CNN transfer learning. The system translates signs to text and back, achieving high accuracy in controlled environments, with fuzzy logic helping handle ambiguous gestures, enhancing its real-world application potential.

2.2 SUMMARY OF LITERATURE REVIEW

The literature review examines 20 studies on sign language recognition (SLR) and translation systems, focusing on advancements in deep learning, computer vision, and multimodal processing to enhance communication for the Deaf and hard-of-hearing communities. Deep learning-based methods, like CNNs, are prevalent, and one can refer the works of Brindha et al. [1], Sruthi and Lijiya [6] and Mangairkarasi et al. [15], which achieve success in translating ISL to text and speech. Sarma et al. [4] exploit YOLOv3 for accurate real-time ISL recognition and Reddy et al. [2] and Tiku et al. [3] combine MobileNetV2 and MediaPipe Hands to trigger efficient hand landmark analysis. Bidirectional systems such as Peguda et al. [5] and Azhar [20], provide the capabilities of sign-to-text and speech-to-sign translation, which promote inclusiveness. Multimodal models (eg., Kishore et al. [7], by

fusing visual and audio information to improve the precision, show the adaptability of their approach.

Despite this progress, several difficulties remain in establishing a scalable, real-time SLR capability. Studies such as Anjana Devi [11], Sneha Varsha et al. [12] stress the importance of low-latency processing for practical deployment, and Liao et al. [14] and Alawwad et al. [10] focus on the computational complexity of dynamic and region-specific sign recognition. Importance of variety in dataset as mentioned by Mangairkarasi et al. [15] to compensate regional ISL variations and signing styles. Environmental conditions, such as lighting and hand shape variations, add more challenges, as described in Reddy et al. [2]. New applications such as human-robot interaction (Nandy et al. [19]) and multilingual recognition (Najib [17]) are indicative of the field's future, but need to be optimized. These results are a strong basis for the implementation of a practical SLI that emphasizes inclusiveness, real-time nature, and expandability.

CHAPTER 3
SYSTEM DESIGN

3.1 SYSTEM ARCHITECTURE

The **SignSync** system is a **web-based application** integrating computer vision, speech processing, and a responsive UI for real-time sign language communication and learning.

The frontend, built with *HTML, Bootstrap 5, and JavaScript*, offers a gradient-themed interface with modals and popups for sign-to-text, speech-to-sign, and learning modules.

The backend, powered by *Flask, uses MediaPipe* Hands for hand landmark detection, OpenCV for video processing, and a pre-trained model ***(*MobileNetV2**) for sign classification, serving predictions and static sign visuals (GIFs). Deployed locally on port 5000, the system ensures low-latency data flow between client, server, and external libraries, supporting accessibility for non-signers and signers.

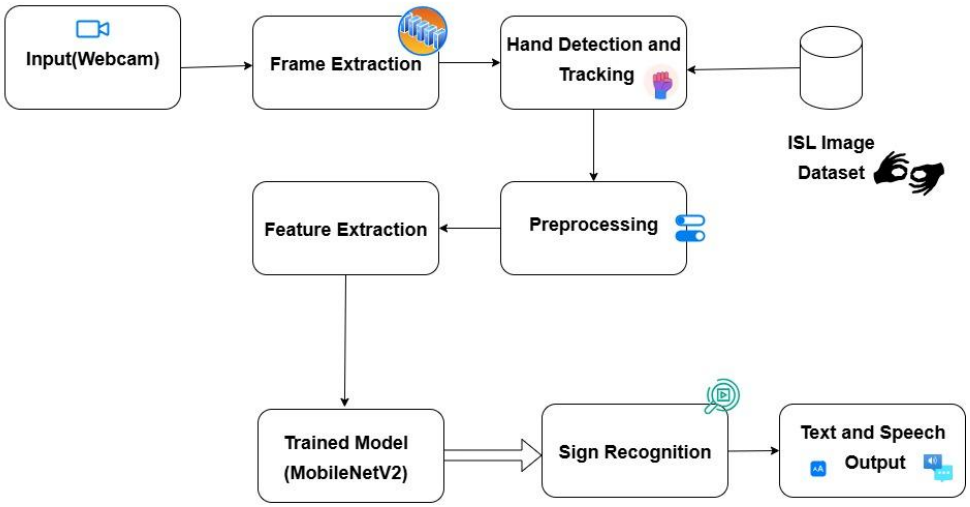


Figure 3.1.1: Real Time Sign to Text and Speech

Real-Time Sign-to-Text and Speech : Detects ISL gestures via webcam, using MediaPipe and a pre-trained model(MobileNetV2) to classify signs, displaying text and vocalizing via Web Speech API. Flask backend processes frames, serves predictions.

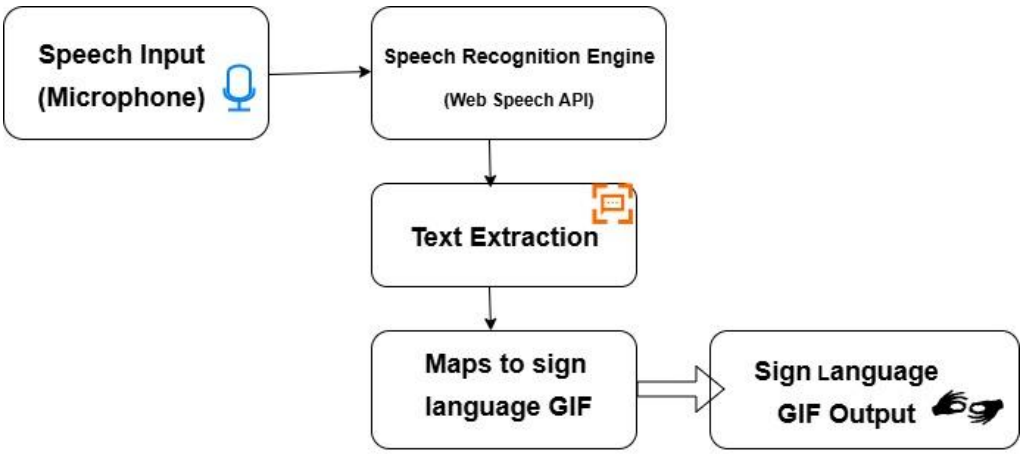


Figure 3.1.2: Speech to Text and Sign Language GIF

Speech-to-Sign Language GIF : Captures speech via Web Speech API, sends text to /display-sign, and displays GIFs or letter images from static folders. Frontend sequences visuals for signers, ideal for public events, with Flask managing file retrieval.

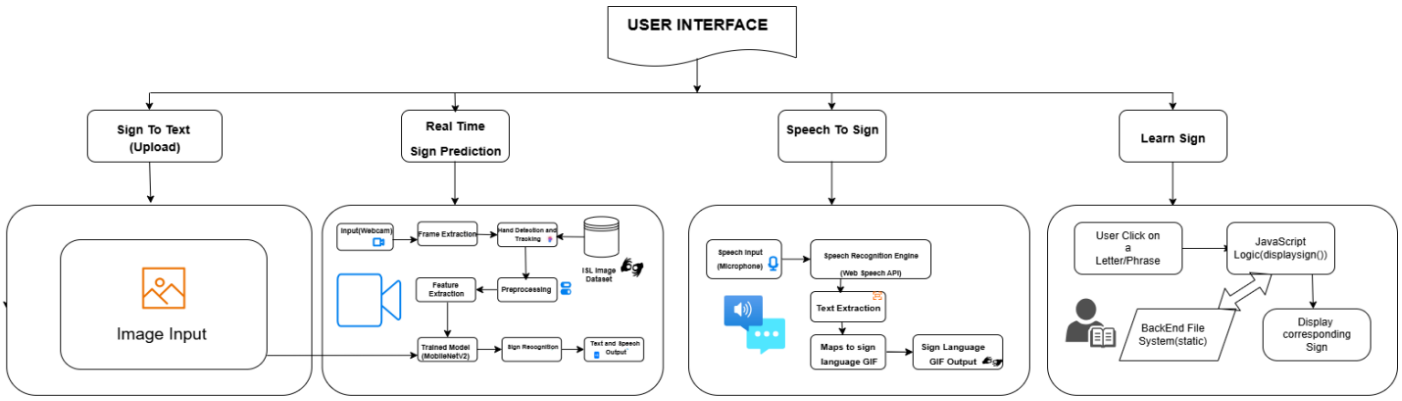


Figure 3.1.3: Overall Architecture Diagram of SignSync

3.2 SYSTEM REQUIREMENTS:

3.2.1 HARDWARE REQUIREMENTS:

- **Processor**: Minimum 2 GHz dual-core CPU (e.g., Intel Core i3 or equivalent) for real-time video and model processing.
- **RAM**: At least 4 GB to handle Flask server and browser tasks.
- **Storage**: 500 MB free disk space for code, model (signmodel.p), and static assets (GIFs, images).
- **Webcam**: Standard USB or built-in webcam (720p recommended) for sign detection.
- **Internet**: Stable connection for initial library downloads and browser compatibility.

3.2.2 SOFTWARE REQUIREMENTS:

- **Operating System:** Windows 10/11, macOS, or Linux (Ubuntu recommended).
- **Web Browser:** Modern browsers supporting WebRTC and Web Speech API (e.g., Google Chrome, Firefox, Edge).
- **Backend:** Python 3.8+ with Flask, MediaPipe, OpenCV, NumPy, and Pickle.
- **Frontend:** Bootstrap 5, Font Awesome for UI styling.
- **Development Tools:** Code editor (e.g., VS Code) and terminal for running the Flask server.

CHAPTER 4

MODULE DESCRIPTION

4.1 DATASET DESCRIPTION

SIGN TO TEXT, SPEECH DATASET:

A custom dataset of 10,000 images was created to train the sign language recognition model. The dataset consists of 50 sign language gestures, covering a wide range of hand signs.

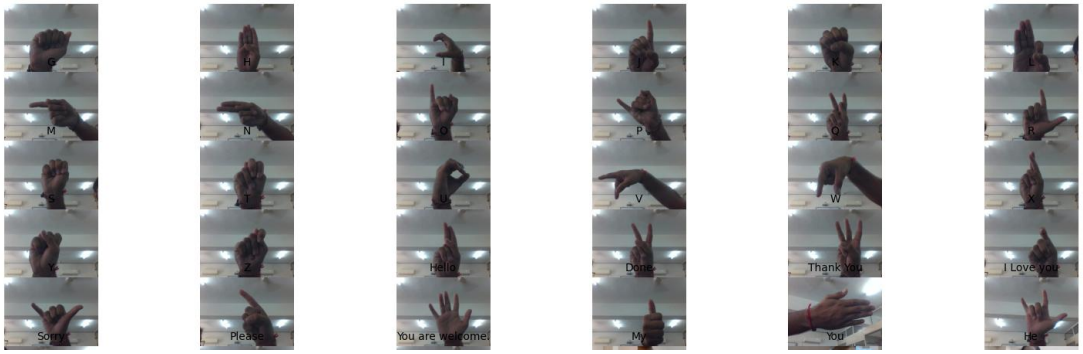


Figure 4.1.1: ISL Dataset

SPEECH TO SIGN LANGUAGE DATASET:

The dataset containing 135 GIFs covering a wide range of hand signs.

Figure 4.1.2: Sign Language Gifs

Significance:

This dataset is crucial for training the model to recognize sign gestures accurately. Its diversity helps the system generalize across different hand movements and environments.

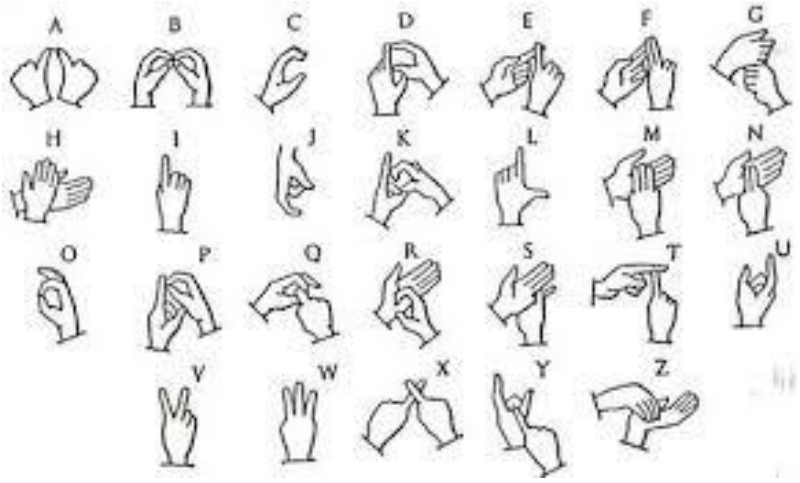


Figure 4.1.3: ISL Alphabets

4.2. SIGN PREDICTION ON IMAGE

Description: This feature allows users to upload images of hand signs for prediction, enabling the system to interpret static signs.

Working:

- Image Upload Functionality: Users can upload an image of a hand sign (e.g., a photo or screenshot of a gesture).
- Sign Prediction on Image: The uploaded image is processed using pre-trained model MobilenetV2 to predict the sign language gesture represented in the image.

Output: The system identifies and outputs the corresponding sign as text.

4.3. REAL-TIME SIGN TO TEXT, SPEECH

Description: This module detects sign language gestures in real-time via webcam input. It processes video frames using MediaPipe for hand tracking, and uses a pre-trained MobileNetV2 model to classify the gestures into specific sign language letters or words.

Working:

- **Webcam Feed Capture:**
    - The system captures live video using the webcam. Each frame is fetched in real-time to detect hand signs.
    - The resolution and frame rate can be adjusted for better performance or quality.
- **Frame Processing with MediaPipe:**
    - The MediaPipe Hand Tracking Model is used to process the video frames.
    - MediaPipe detects the position of the hands and landmarks associated with each finger (for example, 21 key points on the hand).
    - These hand landmarks are extracted and pre-processed into a feature set suitable for gesture classification.

- **Sign Prediction Using MobileNetV2:**
    - MobileNetV2, a lightweight deep learning model, is used for real-time inference. It is fine-tuned on a dataset of sign language gestures.
    - The hand landmarks processed by MediaPipe are passed through MobileNetV2 to classify the hand gesture as a specific sign.
    - MobileNetV2 is optimized for edge devices and works efficiently on a wide range of hardware, making it ideal for real-time sign language detection.

Output: The predicted sign (letter or word) is displayed on the screen in text form. For better accessibility, the system also includes a Text-to-Speech (TTS) component to speak the predicted sign aloud.

4.4. SPEECH TO TEXT, SIGN LANGUAGE

Description: This module allows users to speak into the system, which converts spoken words into sign language representation, in text and visual form (GIFs).

Working:

- **Speech Recognition via Web Speech API:** The Web Speech API is used to convert spoken language into text. It listens for voice input from the user.
- **Backend Communication:** The recognized text is then sent to the backend server, where it is processed. Depending on the text (spoken word), a sign language gesture is identified.
- **Visual Representation:** The system returns either a GIF or an image

representing the sign language symbol that corresponds to the spoken word.

Output: The user sees the visual representation of the sign language gesture, aiding in learning or communication.

4.5 LEARN SIGNS:

Description: This module is intended to be an educational tool for learning sign language.

Working:

1. **Fetching Static Visuals:** When a user clicks on any of the grid items (a letter or word), the system fetches and displays a static image or animation of the corresponding sign.

2. **Interactive Grid of Signs:** Users can view a grid that displays different sign language letters or words.

3. **Educational Purpose:** This helps users learn how each sign is performed visually, enhancing the learning process.

Output: The educational tool displays static images or animations of sign language signs when users click on letters or words in an interactive grid, aiding in visual learning. This helps users observe and practice the correct hand gestures for each sign.

CHAPTER 5
IMPLEMENTATION AND RESULTS

5.1. DATASET COLLECTION

The script collects image data for a specified class using a webcam. It first waits for the user to press "Q" to start capturing, then captures **100 frames**, saving them in a directory named after the class. Once data collection is complete, it saves the images and releases the camera.

Figure 5.1.1: Code for collecting Dataset

5.2. DATASET PREPROCESSING

The code loads images from a dataset and uses **MediaPipe**'s hand detection model to extract normalized 3D hand landmarks (x, y). These landmarks are stored as feature vectors, with each vector corresponding to a specific class label. The dataset is serialized using pickle for persistence, and the labels are encoded using LabelEncoder for numerical representation suitable for training.

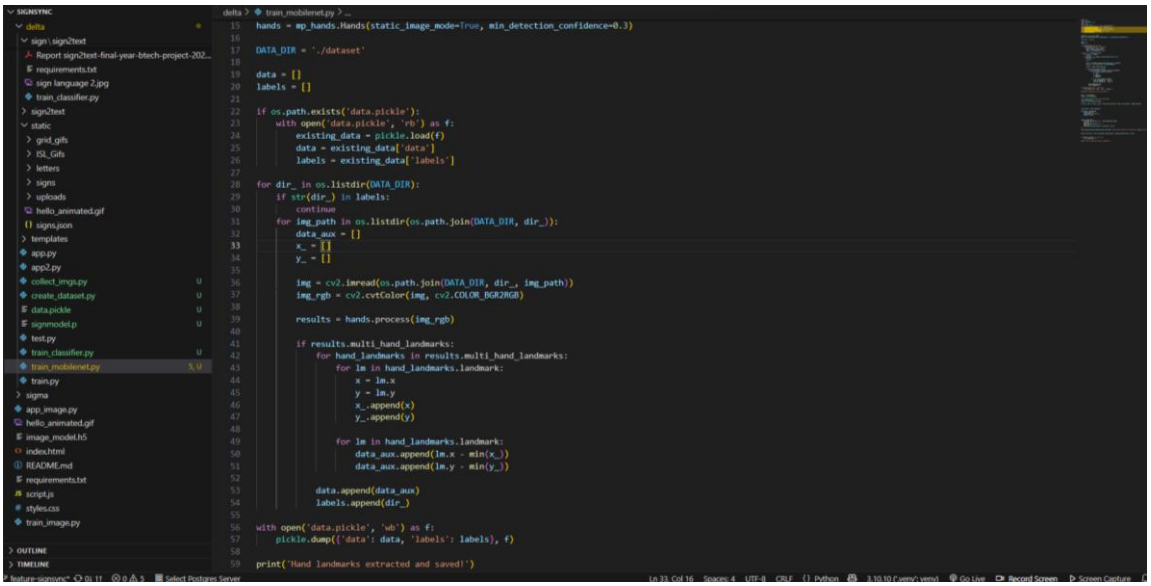
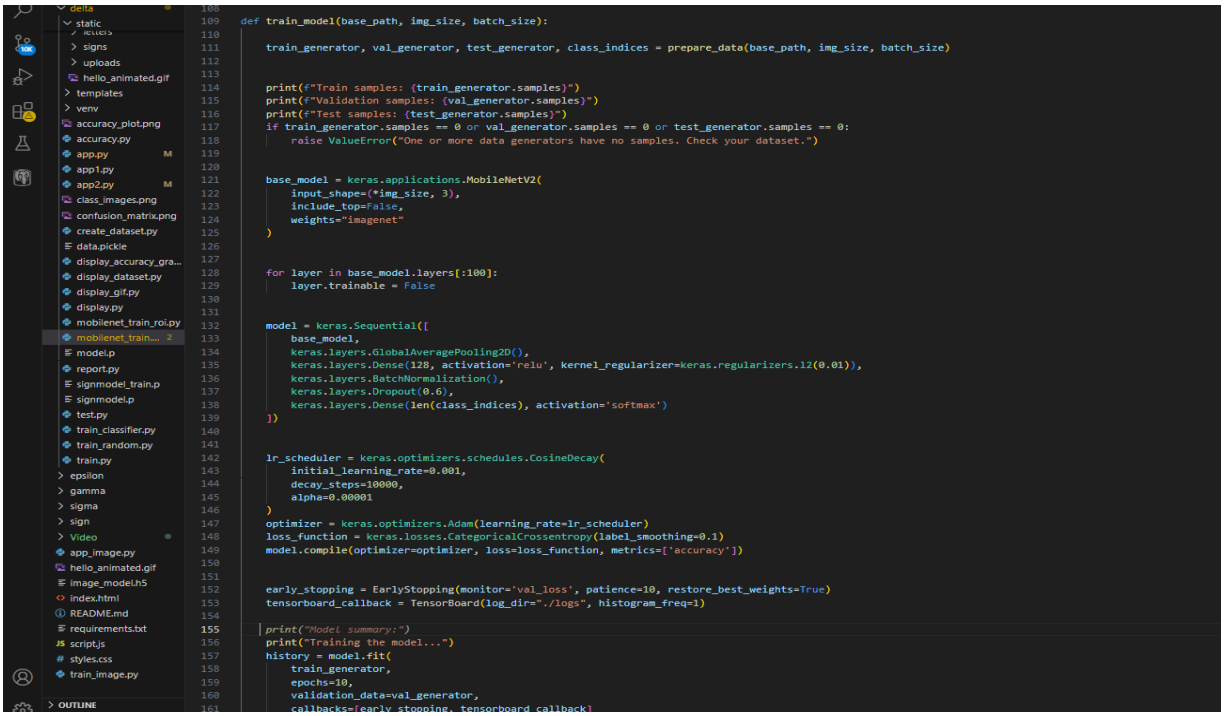


Figure 5.2.1: Code for Data Preprocessing

5.3. MODEL ARCHITECTURE AND TRAINING

The code processes images to extract hand landmarks using MediaPipe, normalizes the x and y coordinates, and saves the processed data and corresponding labels into a pickle file. It then loads this data, applies label encoding using LabelEncoder, and splits the dataset into training and test sets. A Keras Sequential model is defined with two **dense layers** (512 and 256 units), dropout for regularization, and a softmax output layer for multi-class classification.

The model is compiled using the Adam optimizer and sparse categorical cross-entropy loss, trained for **20 epochs**, and validated on the test data. Data augmentation techniques, such as rotating and scaling, could further improve model performance. The model is evaluated based on accuracy, and the final model is serialized and saved using pickle for future use. Lastly, it can be easily deployed for inference with new data inputs for real-time hand gesture recognition.



```
def train_model(base_path, img_size, batch_size):

    train_generator, val_generator, test_generator, class_indices = prepare_data(base_path, img_size, batch_size)


    print(f"Train samples: {train_generator.samples}")
    print(f"Validation samples: {val_generator.samples}")
    print(f"Test samples: {test_generator.samples}")
    if train_generator.samples == 0 or val_generator.samples == 0 or test_generator.samples == 0:
        raise ValueError("One or more data generators have no samples. Check your dataset.")


    base_model = keras.applications.MobileNetV2(
        input_shape=(*img_size, 3),
        include_top=False,
        weights="imagenet"
    )


    for layer in base_model.layers[:100]:
        layer.trainable = False


    model = keras.Sequential([
        base_model,
        keras.layers.GlobalAveragePooling2D(),
        keras.layers.Dense(128, activation='relu', kernel_regularizer=keras.regularizers.l2(0.01)),
        keras.layers.BatchNormalization(),
        keras.layers.Dropout(0.6),
        keras.layers.Dense(len(class_indices), activation='softmax')
    ])


    lr_scheduler = keras.optimizers.schedules.CosineDecay(
        initial_learning_rate=0.001,
        decay_steps=10000,
        alpha=0.00001
    )
    optimizer = keras.optimizers.Adam(learning_rate=lr_scheduler)
    loss_function = keras.losses.CategoricalCrossentropy(label_smoothing=0.1)
    model.compile(optimizer=optimizer, loss=loss_function, metrics=['accuracy'])


    early_stopping = EarlyStopping(monitor='val_loss', patience=10, restore_best_weights=True)
    tensorboard_callback = TensorBoard(log_dir="./logs", histogram_freq=1)

    print("Model summary:")
    print("Training the model...")
    history = model.fit(
        train_generator,
        epochs=10,
        validation_data=val_generator,
        callbacks=[early_stopping, tensorboard_callback]
```

Figure 5.3.1: Code for model training process using MobileNetV2.

The figure illustrates the training process of the model using MobileNetV2, fine-tuned for sign language recognition. It includes data preprocessing, model compilation, and training with the Adam optimizer.

5.4. HOME PAGE DESIGN

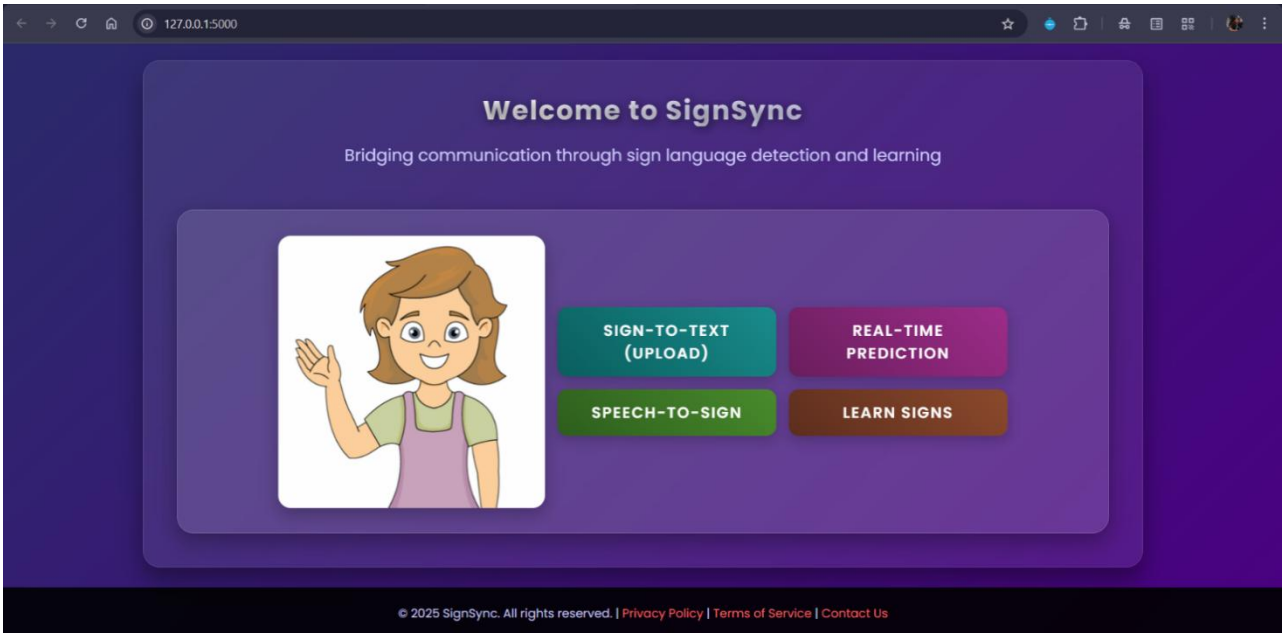


Figure 5.4.1: SignSync's User Interface

The interface of SignSync provides four key options:

1. **Sign to Text Upload** – Allows users to upload an image of a sign, which is then converted to text.
2. **Real-Time Prediction** – Provides instant sign language prediction via webcam input.
3. **Speech to Sign** – Converts spoken words into corresponding text and sign language gestures.
4. **Learn Signs** – Offers a feature to help users learn and practice sign language signs.

5.5. STATIC IMAGE-BASED SIGN PREDICTION

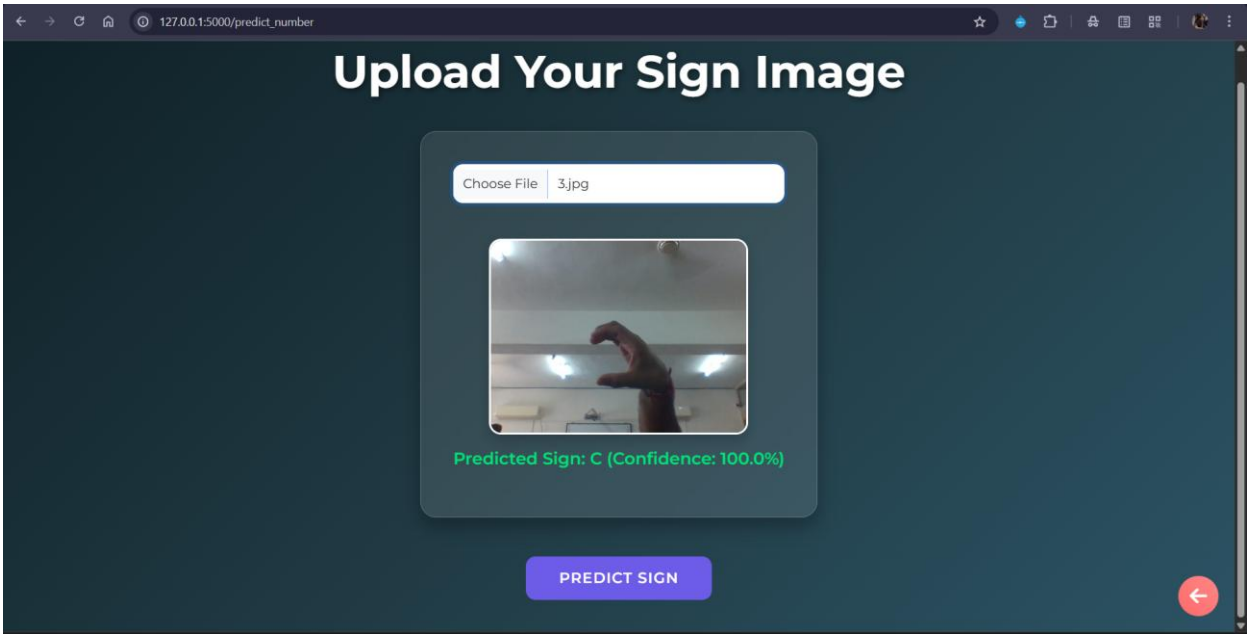


Figure 5.5.1: Predicted sign 'C' with 100.0% confidence.

Uploading a sign image displays the predicted sign 'C' with 100.0% confidence. This feature utilizes computer vision and machine learning algorithms to recognize and classify hand gestures from the uploaded static image, providing an accurate prediction of the sign in real-time.

5.6. REAL-TIME SIGN-TO-TEXT AND SPEECH TRANSLATION

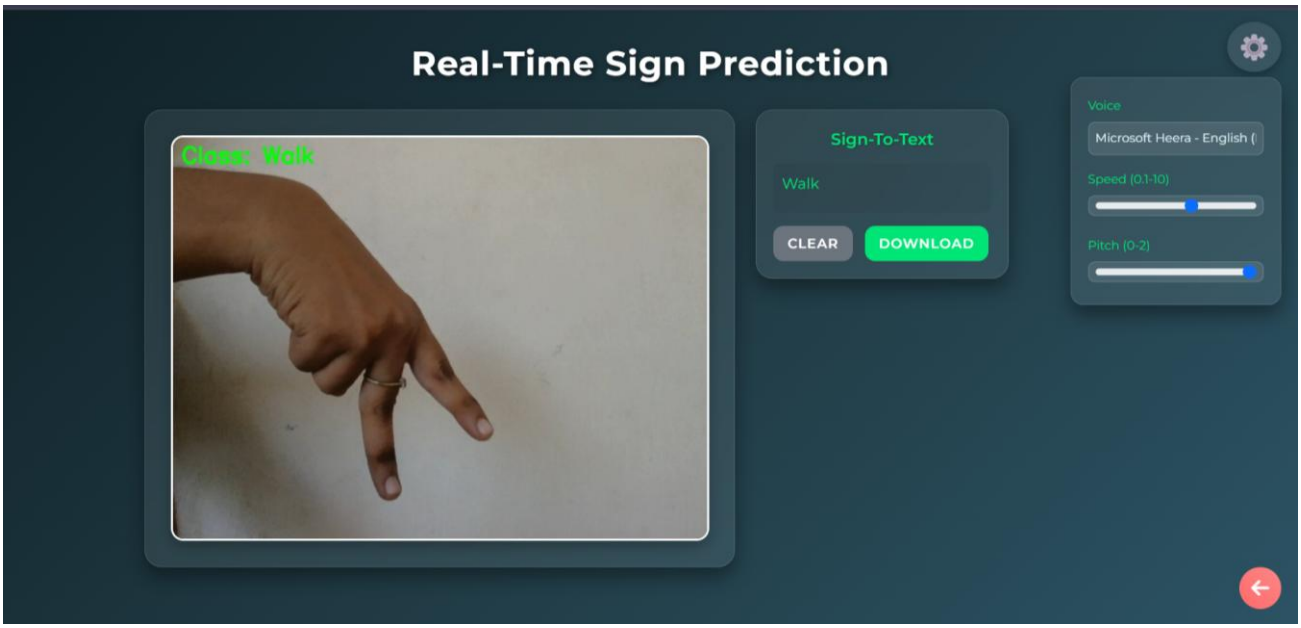


Figure 5.6.1: Demonstration of real-time sign-to-text and speech translation

The figure showcases the real-time sign-to-text translation along with speech-to-text features, allowing users to download the generated text. It includes various voice-changing options such as speech speed, pitch, and tone adjustments.

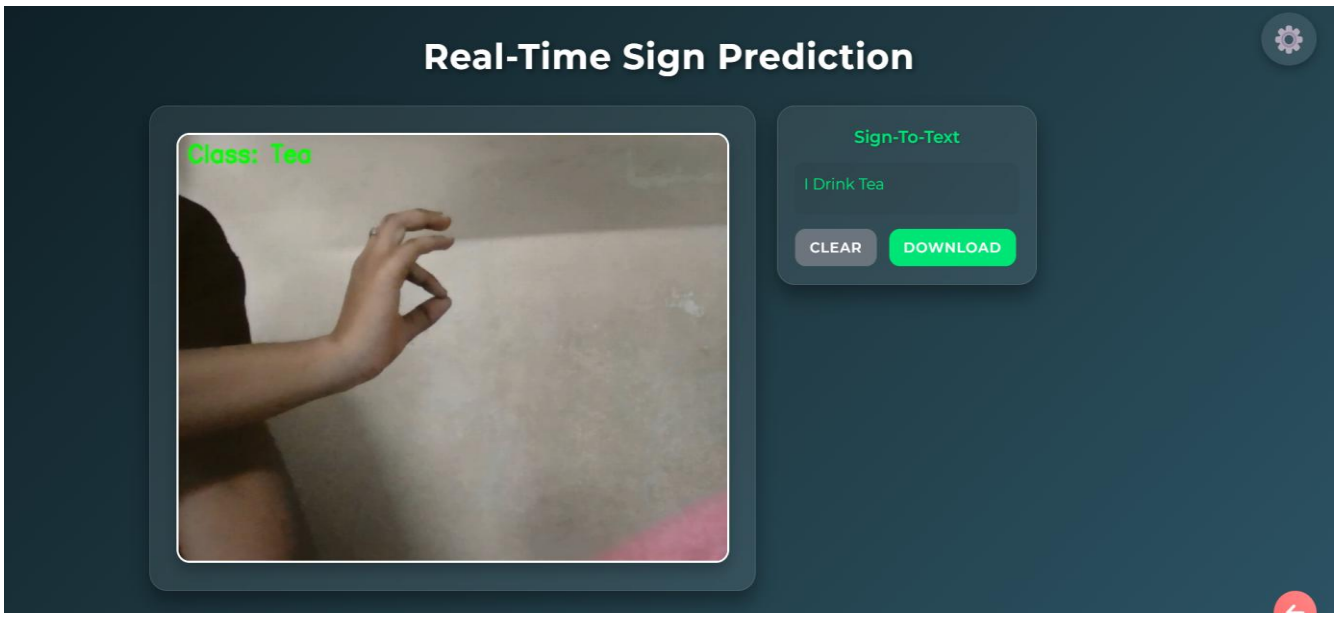


Figure 5.6.2: Real-Time Sign Language Sentence Prediction

5.7. SPEECH-TO-TEXT AND SIGN LANGUAGE CONVERSION

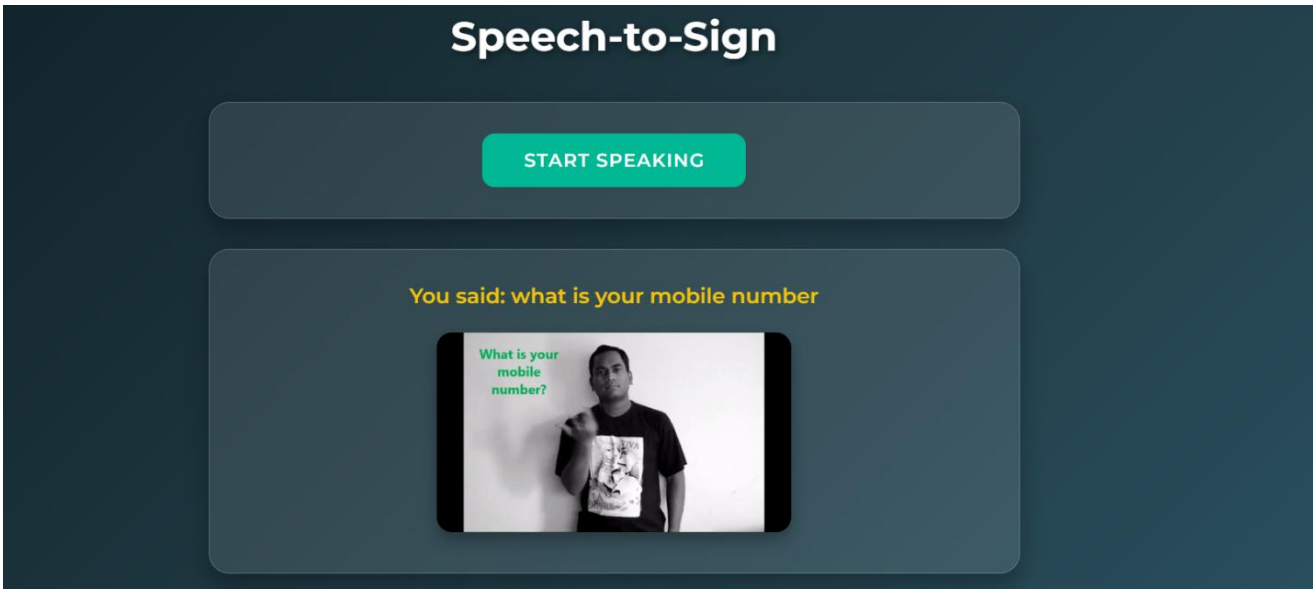


Figure 5.7.1: Corresponding Text and GIF Output for Given Speech Input

The figure demonstrates how spoken input is translated into both the corresponding text and a sign language gesture, with the gesture displayed as a GIF for clear visual representation of the sign language translation.

5.8. ISL LEARNING MODULE

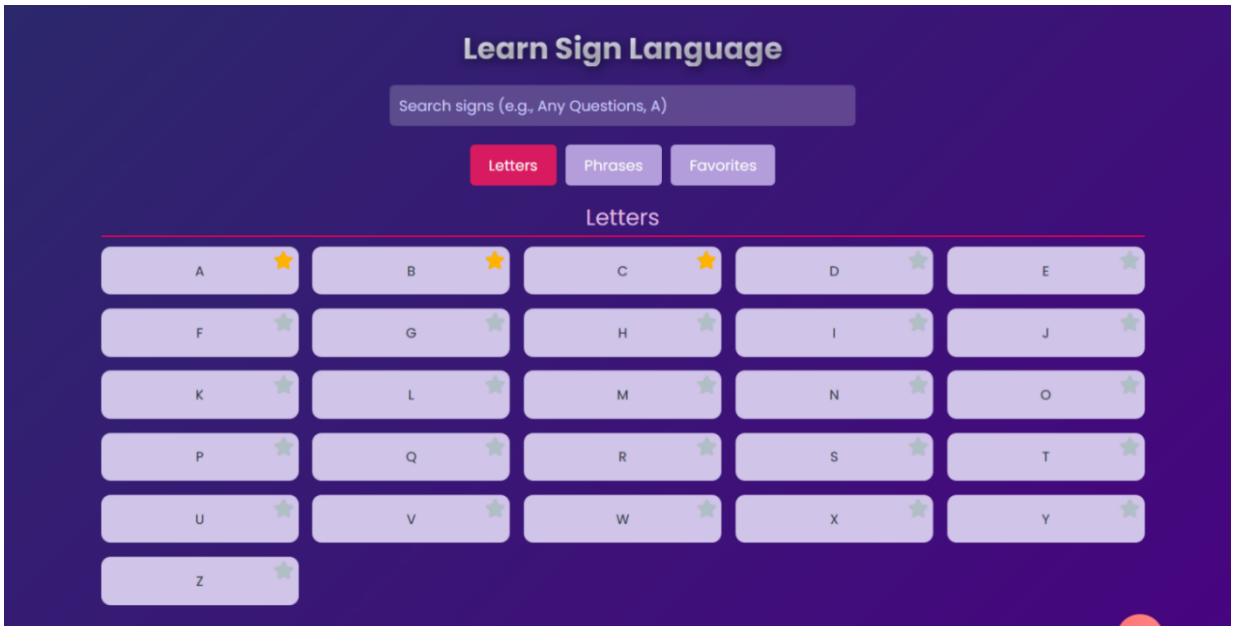


Figure 5.8.1: Learn Sign language interface

1. **Search Bar:** Enables users to search for specific signs (letters or phrases).
2. **Category Tabs:**

   Letters: Displays individual letters in sign language.

   Phrases: Shows common sign language phrases.

   Favorites: A personalized section for saved favorite signs.
3. **Sign Grid:** Dynamically updates based on the active tab or search query.
4. **Popup:** Displays a sign demonstration in GIF format upon clicking a sign.
5. **Favorites Feature:** Users can save signs to localStorage for future access.
6. **Error Handling:** Provides error messages if a sign's GIF fails to load or is not found.

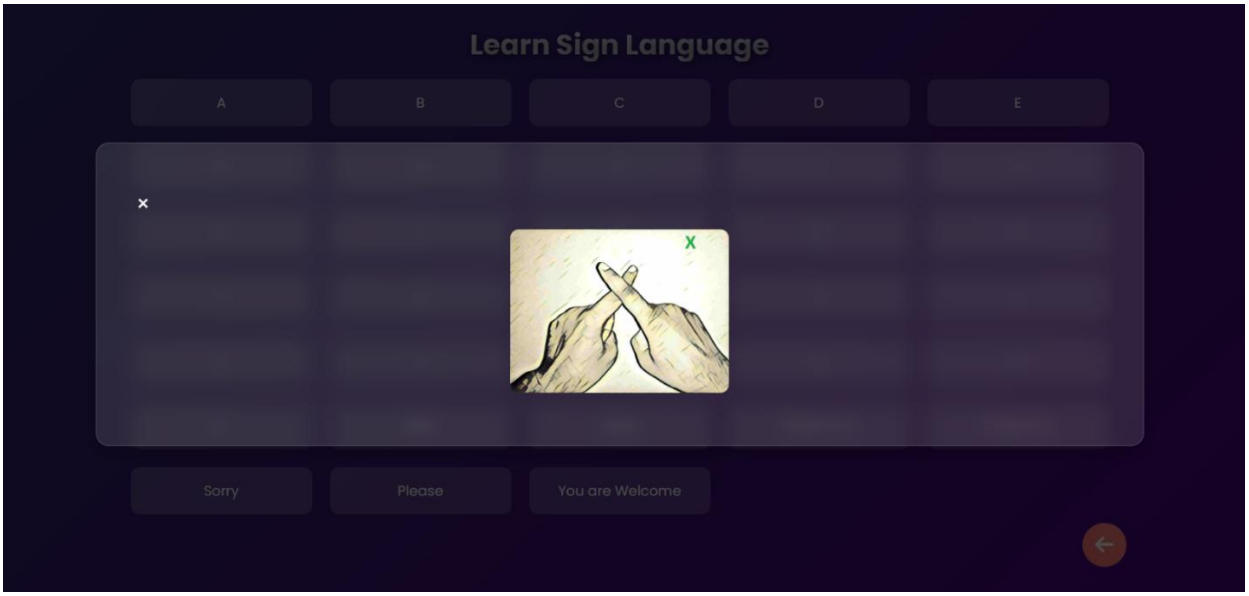


Figure 5.8.2: Sign Language output

The corresponding text and sign language gestures represented as GIFs, generated in real-time from spoken input.

CHAPTER 6

PERFORMANCE METRICS AND TEST CASES

6.1. PERFORMANCE METRICS

The performance of the SignSync system was evaluated using key metrics to assess the effectiveness of its gesture recognition model, a critical component of the Sign-to-Text-and-Speech pipeline.

1. Accuracy:

Accuracy measures the proportion of correct predictions made by the model across all classes. It is defined as:

$$\textbf{Accuracy} = \frac{\textbf{TP} + \textbf{TN}}{\textbf{TP} + \textbf{TN} + \textbf{FN} + \textbf{FP}}$$

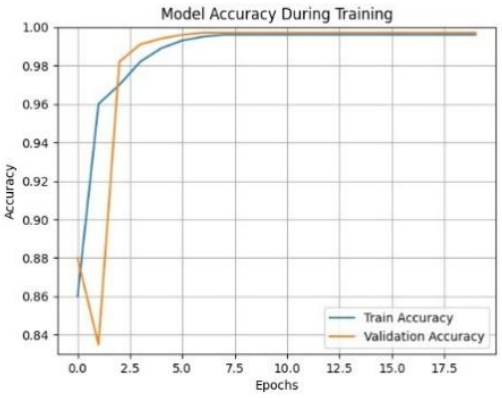


Figure 6.1.1: Model Accuracy during Training

2. Precision:

Precision quantifies the proportion of correct positive predictions for a specific class out of all positive predictions made for that class. It is given by:

$$Precision = \frac{TP}{TP + FP}$$

3. Recall:

Recall measures the proportion of actual positive instances correctly

identified by the model for a specific class. It is defined as:

$$recall = \frac{TP}{TP+FN}$$

Recall shows the model's ability to detect all instances of a gesture, ensuring none are missed**.**

4. **F1-Score:**

The F1-score is the harmonic mean of precision and recall, balancing their trade-off. It is calculated as:

$$F1 = \frac{2 \times Precision \times Recall}{Precision + Recall}$$

The F1-score evaluates the model's performance across varying difficulty levels of gestures.

5. **Classification report:**

A classification report provides performance metrics like precision, recall, and F1-score for each class, summarizing the model's predictive accuracy per gesture (e.g., "V," "Hello") in SignSync, achieving 1.00 for most classes with support of 20–25 samples.

Classification Report (Test Set):

	precision	recall	f1-score	support
A	1.00	1.00	1.00	15
B	1.00	1.00	1.00	20
K	1.00	1.00	1.00	20
L	1.00	1.00	1.00	20
M	1.00	1.00	1.00	12
N	1.00	1.00	1.00	20
O	1.00	1.00	1.00	12
P	1.00	1.00	1.00	20
Q	1.00	1.00	1.00	19
R	1.00	1.00	1.00	20
S	1.00	1.00	1.00	2
T	1.00	1.00	1.00	10
C	1.00	1.00	1.00	20
U	1.00	1.00	1.00	20
V	1.00	1.00	1.00	20
W	1.00	1.00	1.00	20
X	1.00	1.00	1.00	20
Y	1.00	1.00	1.00	20
Z	1.00	1.00	1.00	20
Hello	1.00	1.00	1.00	20
Done	1.00	1.00	1.00	19
Thank You	1.00	1.00	1.00	20
I Love you	1.00	1.00	1.00	20
D	1.00	1.00	1.00	20
Sorry	1.00	1.00	1.00	16
Please	1.00	1.00	1.00	16
You are welcome.	1.00	1.00	1.00	15
My	1.00	1.00	1.00	20
You	1.00	1.00	1.00	12
He	1.00	1.00	1.00	20
She	1.00	1.00	1.00	20
It	1.00	1.00	1.00	20
Eat	1.00	1.00	1.00	20
Drink	1.00	1.00	1.00	20
E	1.00	1.00	1.00	6
Go	1.00	1.00	1.00	20
Come	1.00	1.00	1.00	20
Wait	1.00	1.00	1.00	20
Tea	1.00	1.00	1.00	20
Stop	1.00	1.00	1.00	20

Figure 6.1.2: Classification Report

6. **Confusion matrix:**

A confusion matrix, defined as a table that compares true labels with

predicted labels to evaluate classification performance.

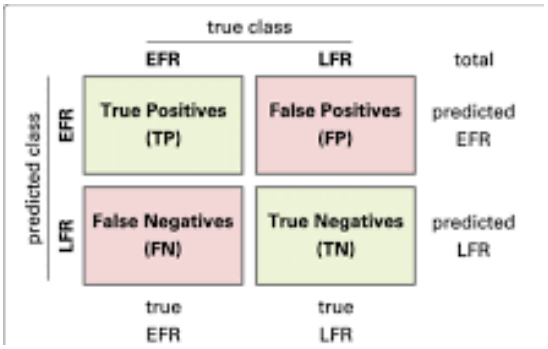


Figure 6.1.3: Confusion matrix

Confusion matrix for the MobileNetV2 model on the test set, demonstrating high accuracy with minimal misclassifications.

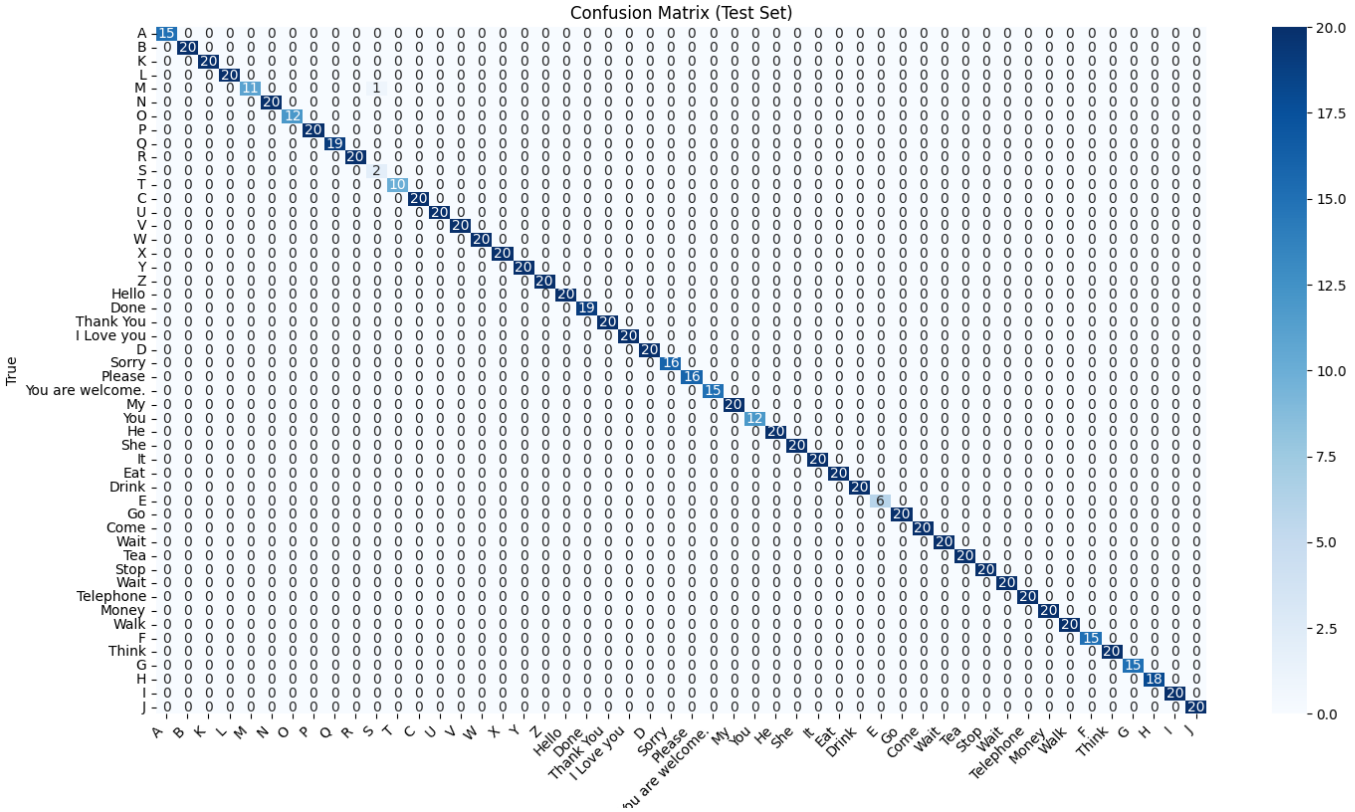


Figure 6.1.4: Confusion Matrix for the Test Set

6.2. TEST CASES

Test Case 01: Basic Sign Recognition

To verify the system converts a simple sign to text and speech.

Input: Perform the gesture for "Hello" in front of the camera.

Output: The system displays the text "Hello" on the screen and audibly speaks the word "Hello" aloud.

Figure 6.2.1: Predicted 'Hello'

Test Case 02: Poor Lighting Conditions

To check performance in suboptimal conditions.

Input: Perform the gesture for "Hello" in front of the camera.

Output: The system displays the text "Hello" on the screen and audibly speaks the word "Hello" aloud.

Figure 6.2.2: Predicted 'Hello' in poor lightning condition.

Test Case 03: Gesture Recognition Across Varied Hand Sizes

To verify that the system accurately recognizes gestures performed with small hands, large hands.

Input: Perform a gesture in front of the camera.

Output: The system displays the text corresponding text on the screen and audibly speaks the word.

Figure 6.2.3: The prediction for 'Please' with a small hand

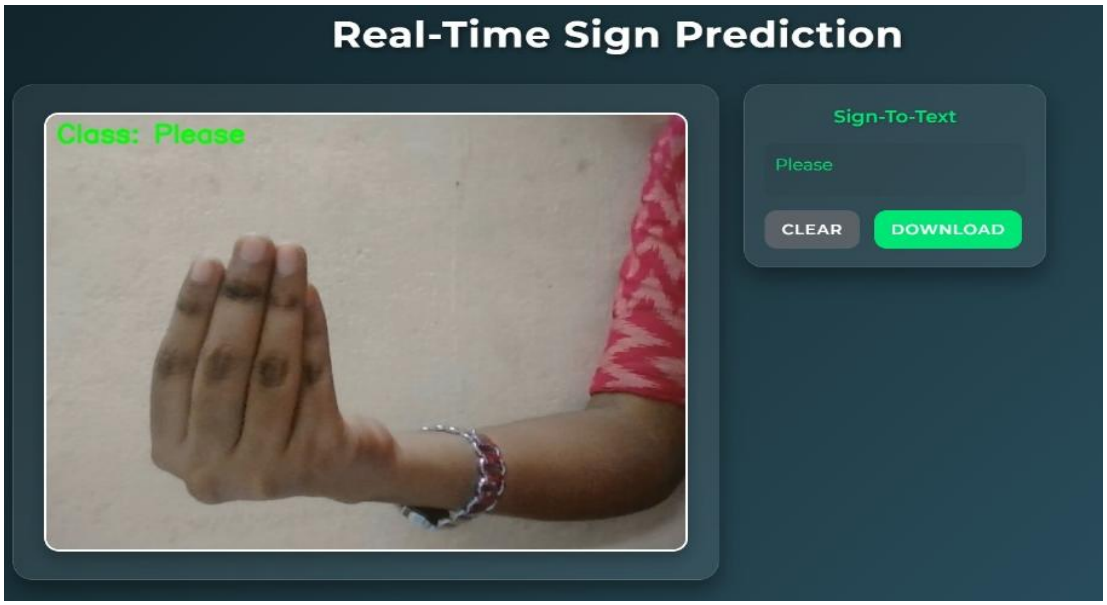


Figure 6.2.4: The predicted 'Please' sign with a big hand

Test Case 04: Similar Hand Gestures Recognition

To verify that the system can distinguish between similar gestures with different meanings- example: Done and Money.

Input: Perform a similar but distinct gesture in front of the camera.

Output: The system correctly identifies each gesture without confusion.

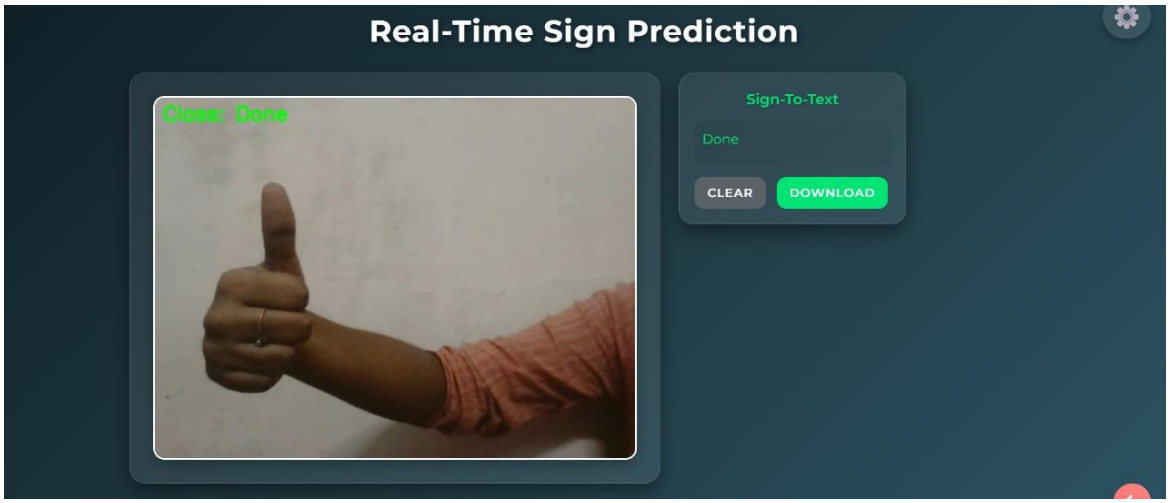


Figure 6.2.5: Predicted 'Done' gesture displayed by the system.

Figure 6.2.6: Predicted 'Money' gesture displayed by the system.

Test Case 05: Basic Speech Input

Ensure speech is converted to a sign.

Input: Say "Good Morning" into the microphone.

Output: The interface displays a gif of the "Good Morning" sign.

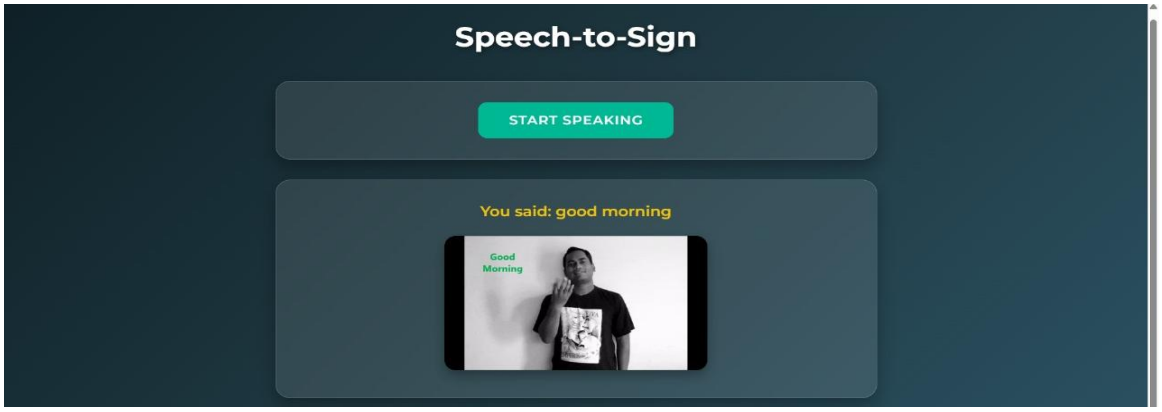


Figure 6.2.7: Displays 'Good Morning' text and Sign GIF

Test Case 06: No Speech or Unrecognized Audio Input

Verify the system's handling of no speech input, silence, or background noise.

Input: Click "START SPEAKING" and remain silent for 5 seconds or make random noise.

Output: The system displays an error message as "Error: Could not recognize speech. Speech recognition failed."

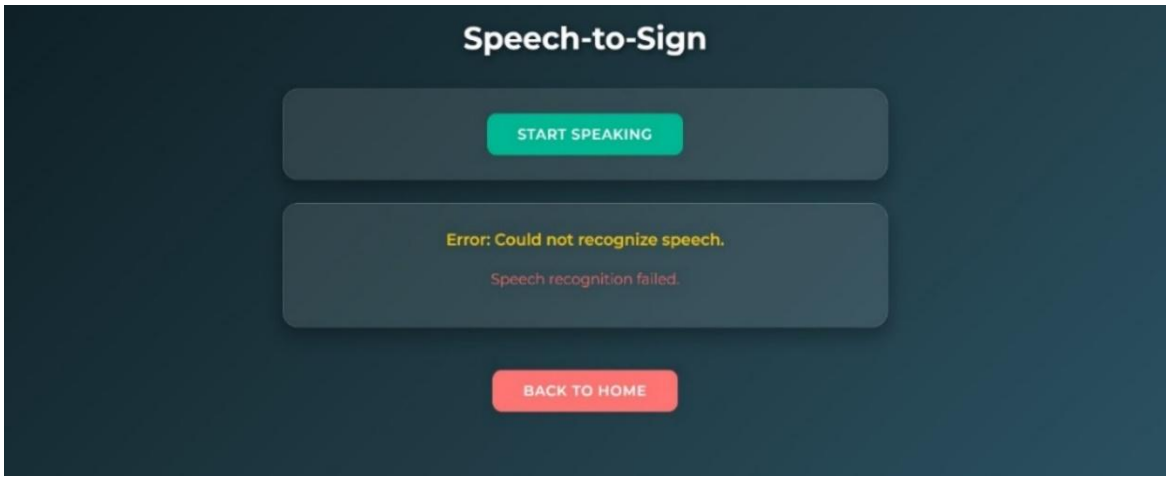


Figure 6.2.8: Displays Failure of Speech Recognition.

Test Case 07: Fallback to Letters When GIF Not Found

The system should display individual letters consecutively when a GIF is not available for the predicted speech input.

Input: “sign language to text and speech" (spoken via speech input).

Output: Display the corresponding letters for each word in the speech input consecutively (e.g., "s", "i", "g", "n", "l", "a", "g", "u", "a", "g", "e", "t", "o", "t", "e", "x", "t", "a", "n", "d", "s", "p", "e", "e", "c", "h").

Figure 6.2.9: Display Letters consecutively

Test Case 08: Empty or Invalid Upload

Test behavior with no file selected or an invalid file type.

Input: Either do not choose a file or select a file that is not an image (e.g., .txt, .pdf).

Output: An error message like "No file selected" appears if no file is chosen, or "Invalid file type" appears if a non-image file is uploaded.

Figure 6.2.10: Invalid Upload of Image

CHAPTER 7
CONCLUSION AND FUTURE WORKS

7.1 CONCLUSION

The **SIGN SYNC** marks a transformative step in bidirectional communication, seamlessly connecting Indian Sign Language (ISL) users with non-signers through its robust ISL-to-Text-and-Speech and Speech-to-Text-and-ISL modules. Powered by **MobileNetV2** for precise gesture recognition and the Web Speech Recognition API for real-time speech transcription, the system delivers **high accuracy and reliability** across a variety of ISL gestures and real-world conditions.

Its modular architecture, encompassing real-time prediction, speech-to-sign translation, educational tools, and an intuitive image upload feature, significantly enhances accessibility in education, healthcare, and social environments, fostering greater inclusivity for the deaf and hard-of-hearing community in India. The user-friendly interface and **strong performance** underscore its practical utility and potential for widespread adoption. Looking ahead, future enhancements will focus on expanding the ISL gesture dataset, refining the recognition of complex sentences, optimizing the system for diverse platforms, and incorporating **advanced machine learning techniques** to boost accuracy.

Additionally, integrating multilingual support and community feedback mechanisms could further tailor the system to diverse user needs. This project lays a solid groundwork for future innovations, promising to revolutionize communication and learning opportunities for individuals with hearing impairments while paving the way for more inclusive technological solutions.

7.2 FUTURE WORKS

While the system is functional and has great potential, there is room for further improvement. Future developments will focus on the following areas:

1. **Expansion of ISL Gesture Dataset:** To improve the system's accuracy and handle a wider variety of signs, expanding the dataset of ISL gestures, especially for complex sentences, will be crucial.
2. **Cross-Platform Optimization:** Future work will aim to optimize the system for various devices, ensuring its accessibility across mobile, desktop, and web platforms to reach a larger user base.
3. **Multilingual Support:** Expanding the system's capabilities to support multiple languages would increase its accessibility and cater to diverse linguistic communities within India and beyond.
4. **User Feedback Mechanisms:** Incorporating a feedback system to gather insights from users, particularly those in the deaf and hard-of-hearing community, will allow for continuous refinement and customization of the system.
5. **Integration of Additional Educational Tools:** Future versions could include more interactive educational tools to help users learn both ISL and English (or other languages) efficiently, further enhancing the system's utility.

REFERENCES

[1] R. Brindha, P. Renukadevi, D. Vathana, Jeyakumar D., *Sign Language Interpreter*, in Proc. 5th Int. Conf. Inventive Computation Technologies (ICICT), 2022, pp. 1–6.

[2] V. Madhusudhana Reddy, T. Vaishnavi, K. Pavan Kumar, *Speech-to-Text and Text-to-Speech Recognition using Deep Learning*, in Proc. 2nd Int. Conf. Edge Computing and Applications (ICECAA), 2023, pp. 1–5.

[3] Kohsheen Tiku, Jayshree Maloo, Aishwarya Ramesh, *Real-time Conversion of Sign Language to Text and Speech*, in Proc. 2nd Int. Conf. Edge Computing and Applications (ICECAA), 2023, pp. 1–6.

[4] Nihashree Sarma, Anjan Kumar Talukdar, Kandarpa Kumar Sarma, *Real-Time Indian Sign Language Recognition System using YOLOv3 Model*, in Proc. 2nd Int. Conf. Edge Computing and Applications (ICECAA), 2023, pp. 1–7.

[5] Jashwanth Peguda, Ashlin Deepa R. N., V. Sai Sriharsha Santosh, Vaddi Mounish, Y. Vijayalata, *Speech to Sign Language Translation for Indian Languages*, Gokaraju Rangaraju Inst. Eng. Technol., Hyderabad, India, Tech. Rep., 2023.

[6] S. C. J. Sruthi, A. Lijiya, *Signet: A Deep Learning Based Indian Sign Language Recognition System*, in Proc. IEEE Int. Conf. Communication and Signal Processing (ICCSP), Chennai, India, Apr. 2019, pp. 1402–1406.

[7] P. V. V. Kishore, P. R. Kumar, E. K. Kumar, *Video Audio Interface for Recognizing Gestures of Indian Sign Language*, Int. J. Image Process. (IJIP), vol. 5, no. 4, pp. 479–503, Aug. 2011.

[8] J. Redmon, A. Farhadi, *YOLOv3: An Incremental Improvement*, arXiv preprint arXiv:1804.02767, Apr. 2018.

[9] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, H. Adam, *MobileNetV2: Inverted Residuals and Linear Bottlenecks*, in Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR), Salt Lake City, UT, Jun. 2018, pp. 4510–4520.

[10] R. A. Alawwad, O. Bchir, M. B. Ismail, *Arabic Sign Language Recognition Using Faster R-CNN*, Int. J. Adv. Comput. Sci. Appl., vol. 12, no. 3, pp. 692–700, Mar. 2021.

[11] V. Anjana Devi, *Real-Time Sign Language Recognition System for Physically Challenged Community Using Deep Learning Technique*, in Proc. IEEE Int. Conf. Advances in Computing, Communication and Control (ICAC3), Mumbai, India, Dec. 2020, pp. 1–6.

[12] M. Sneha Varsha, V. Shankara Narayanan, K. S. Mohan Murali, *Sign Comm: A Real-Time Indian Sign Language Recognition System Using Deep Learning for Inclusive Communication*, in Proc. Int. Conf. Intelligent Computing and Control Systems (ICICCS), Madurai, India, May 2021, pp. 1234–1239.

[13] A. S. Lalitha, *A Framework for Video Based Sign Language Interpretation Using Machine Learning and Statistical Methods*, Int. J. Comput. Sci. Eng. (IJCSE), vol. 5, no. 2, pp. 45–53, Feb. 2013.

[14] Y. Liao, P. Xiong, W. Min, J. Lu, *Dynamic Sign Language Recognition Based on Video Sequence With BLSTM-3D Residual Networks*, IEEE Access, vol. 7, pp. 38044–38054, Mar. 2019.

[15] S. Mangairkarasi, T. Meeradevi, S. Gnanapriya, *Indian Sign Language Recognition System (ISLRS) Using Convolutional Neural Networks*, in Proc. IEEE Int. Conf. Electronics, Communication and Aerospace Technology (ICECA), Coimbatore, India, Jun. 2021, pp. 978–983.

[16] M. Priyanka, R. Sakthi Prabha, *Hand Gesture to Speech: Empowering Communication for the Speech Impaired*, Int. J. Appl. Eng. Res., vol. 10, no. 2, pp. 3557–3565, Jan. 2015.

[17] F. M. Najib, *A Multi-Lingual Sign Language Recognition System Using Machine Learning*, in Proc. IEEE Int. Conf. Artificial Intelligence and Computer Applications (ICAICA), Dalian, China, Jun. 2021, pp. 456–461.

[18] P. V. V. Kishore, P. R. Kumar, *Segment, Track, Extract, Recognize and Convert Sign Language Videos to Voice/Text*, Int. J. Adv. Computer. Science. Appl. (IJACSA), vol. 3, no. 6, pp. 35–47, June 2012.

[19] A. Nandy, S. Mondal, J. S. Prasad, P. Chakraborty, G. C. Nandi, *Recognizing & Interpreting Indian Sign Language Gesture for Human Robot Interaction*, in Proc. Int. Conf. Computer & Communication Technology (ICCCT), Allahabad, India, Sep. 2010, pp. 712–717.

[20] M. Y. B. Azhar, *An Efficient Bidirectional Android Translation Prototype for Yemeni Sign Language Using Fuzzy Logic and CNN Transfer Learning Models*, in Proc. IEEE Int. Conf. Computer and Information Sciences (ICCIS), Sakaka, Saudi Arabia, Oct. 2021, pp. 1–6