FA1 Activity Report – American Sign Language (ASL) Fingerspelling Recognition System

1. Aim

To develop a real-time platform that captures American Sign Language (ASL) gestures via webcam, translates them into English text, and provides instant audio output—enabling seamless communication between deaf/hard-of-hearing individuals and those who do not understand sign language.

---

2. Objectives

The main objective of this project is to design and implement a comprehensive platform that facilitates the real-time translation of American Sign Language (ASL) into English text and speech. This solution aims to bridge the communication gap between sign language users and non-signers by leveraging advanced computer vision, natural language processing, and user-centered design principles.

Component Breakdown

1. Implement Real-Time ASL Recognition

Description:
Develop or integrate a robust computer vision system capable of detecting and classifying ASL signs through webcam input. The system should initially focus on recognizing individual alphabets in ASL and expand to handle complete words and phrases over time.

Technical Approach:

- Utilize machine learning models, such as convolutional neural networks (CNNs), for high-accuracy hand gesture recognition.
- Implement real-time processing to ensure responsiveness and reliability for users.
- Ensure the model is trained on a diverse dataset to account for different hand shapes, skin tones, and lighting conditions.
- Continuously improve recognition accuracy by utilizing user feedback and expanding the training dataset.

Expected Outcomes:

- High-precision detection and classification of ASL letters and words.
- Minimal latency in sign recognition and response.
- Seamless integration with standard webcam devices.

---

2. Translate ASL to English Text

Description:
Once ASL signs are recognized, the system should accurately convert these signs into English letters or words. The translation algorithm must display text with clarity and speed, serving as the foundation for communication between signers and non-signers.

Technical Approach:

- Leverage ASL grammar rules and context analysis for phrase and sentence-level translation.
- Incorporate error correction algorithms to refine output in cases of partial or ambiguous recognition.
- Provide a clear, readable output interface for displaying translated text to the user.
- Enable a real-time preview so users can follow the translation as they sign.

Expected Outcomes:

- Accurate conversion of ASL input to corresponding English text.
- Clear and prompt display of translated content.
- Support for expanding the range of recognized vocabulary beyond simple words.

---

3. Provide English Voice Output

Description:
Integrate a text-to-speech (TTS) module that reads the translated English text aloud. This feature is crucial for enabling communication with individuals who prefer auditory input or cannot view written text.

Technical Approach:

- Use state-of-the-art TTS engines to ensure natural and intelligible voice output.
- Allow customization of voice parameters (pitch, speed, gender) to meet user preferences.
- Support instant audio playback for translated text, with toggles for repeat or pause functions.

Expected Outcomes:

- Accurate vocalization of translated text.
- Clear, natural-sounding voice output suitable for diverse environments.
- Enhanced accessibility for visually impaired users.

---

4. Design a User-Friendly Interface

Description:
Create a highly accessible, intuitive, and responsive platform that caters to the needs of all users, regardless of their technical proficiency or device.

Technical Approach:

- Employ modern UI/UX standards to ensure ease of use and accessibility.
- Make the platform responsive for desktops, tablets, and mobile devices.
- Provide interactive tutorials and help sections to guide new users.
- Ensure compliance with accessibility standards (e.g., WCAG) for users with disabilities.

Expected Outcomes:

- An inclusive and engaging user experience.
- Minimal learning curve for first-time users.
- Full accessibility across various devices and assistive technologies.

---

5. Facilitate Inclusive Communication

Description:
Bridge the communication gap between sign language users and non-signers by enabling immediate translation and audio feedback, promoting equal opportunities for interaction.

Technical Approach:

- Enable one-on-one and group conversations using the platform.
- Support integration with existing communication tools (e.g., video conferencing platforms).
- Encourage real-time feedback and iterative improvement through user suggestions.

Expected Outcomes:

- Effective, inclusive communication in both personal and professional settings.

- Increased confidence and participation of sign language users in mixed-language environments.
- Strong community engagement and platform adoption.

3. Motivation

Sign language is the primary mode of communication for more than 70 million deaf people worldwide. Its importance extends beyond daily conversations—sign language empowers these individuals to learn, work, access vital services, and actively participate in their communities. By using sign languages, deaf people overcome barriers in society and ensure their inclusion despite hearing disabilities.

However, not everyone can learn or use sign language. This creates a significant gap in communication between deaf individuals and the larger population, often leading to feelings of isolation and limited access to opportunities. If all people with disabilities are to enjoy their rights equally, society must find better solutions to facilitate communication.

The challenge lies in making society truly inclusive, where communication is effortless—even when someone does not know sign language. The ambitious aim of this project is to create a human-computer interface (HCI) that understands American Sign Language (ASL) and serves as a bridge between deaf or mute individuals and those who cannot sign. By automating the recognition and translation of sign language, the system will help users communicate freely, access essential services, and participate more fully in society.

By focusing on user-friendly design, the project seeks to make advanced technology accessible to everyone, not just experts. Ultimately, this solution will improve the daily lives of deaf and mute people, making interactions easier and giving them greater independence and confidence in a world that often overlooks their needs.

4. Literature Survey

Title of Paper	Author & Year	Work Presented	Methodology Used	Accuracy / Performance	Key Findings
Using Deep Learning in Sign Language Translation to Text	Mary Jane C. Samonte et al. (2022)	Review of DL methods for sign language to text	CNN, CTC, DBN; literature review	High accuracy for DL methods	CNN most used; accuracy depends on data; research gaps remain

Title of Paper	Author & Year	Work Presented	Methodology Used	Accuracy / Performance	Key Findings
Real Time Conversion of Sign Language to Speech and Prediction of Gestures using ANN	Abey Abraham, Rohini V (2018)	Glove device converts signs to speech, predicts needs	Flex sensors, Arduino, ANN, Android app	Low mean square error, adaptive	Real-time gesture to speech, predictive user needs
Sign Language to Text Conversion	Deepika Patil et al. (2024)	Real-time ASL finger-spelling to text via neural networks	Camera images, LSTM, custom dataset	Up to 97% (P-Z), 96% (G-O); >90% overall	LSTM improves accuracy; confusion in similar signs; robust, user-friendly system
Sign Language to Text Conversion in Real Time using Transfer Learning	Shubham Thakar et al. (2024)	ASL sign-to-text app using transfer learning and CNN	CNN and VGG16 transfer learning, image preprocessing, large ASL dataset, web app	CNN: 94%, Transfer Learning (VGG16): 98.7%	Transfer learning boosts accuracy; robust app; easily extendable to other sign languages
Audio or Text to Sign Language Converter	Satvika Reddy Satti, P. Pavithra (2023)	Converts audio/text to Indian Sign Language (ISL) animations	NLP, Python, Django, NLTK, Blender 3D animation	High accuracy (qualitative)	Bridges communication gap for Indian users; 3D animated ISL output; improves education and inclusivity
Air Writing Word Recognition and Translation	Neha Ganesh Karande et al. (2024)	Recognizes air-written words with hand tracking and CNN	Webcam-based hand/finger tracking, CNN, translation module	Up to 98.3% (digits); ~75-80% (letters/words)	Real-time air-writing with basic setup; accurate recognition; future potential for accessibility & HCI
Air Writing Detection and Recognition	Amith K R, Nikhil Holla R, Prashanth J (2024)	Color-based air writing tracked by webcam, ML, virtual canvas	Computer vision object tracking, color	Up to 95% (digits); >90% (characters)	Hands-free, color-based air writing; robust ML; smart devices, HCI

Title of Paper	Author & Year	Work Presented	Methodology Used	Accuracy / Performance	Key Findings
			segmentation, CNN, ML		
American Sign Language Recognition Based on Transfer Learning Algorithms	Sanaa Mohsin et al. (2024)	Compares transfer learning models for ASL recognition	ASL dataset, transfer learning (VGG16, ResNet50, MobileNetV2, InceptionV3), CNN, data augmentation, Keras	InceptionV3: 96%, VGG16: 95%, MobileNetV2: 92%, ResNet50: 89%, CNN: 85%	Transfer learning (esp. InceptionV3) excels; robust ASL letter/number recognition; flexible for resources/applications
Intelligent Sign Language Recognition for Real-Time Text Conversion to Aid Speech and Hearing Impaired	Abhishek Tripathi et al. (2025)	Glove-based system converts sign language to text in real-time	Arduino UNO, IoT, flex sensors on glove, LCD display, Python data acquisition, custom gesture mappings, multi-user validation	85% (simple gestures), 75% (complex gestures)	Low-cost, portable glove system; good for simple gestures; robust across users; opportunities to improve complex gesture accuracy

5. Proposed Solution & Architecture Diagram

The project is designed to build a real-time ASL recognition system that transforms hand signs representing words into spoken English. The system employs computer vision and deep learning to capture, interpret, and convert ASL gestures to English words, followed by audio output for seamless communication.

Workflow

1. **Input Acquisition:**
   A webcam captures live video streams of the user's hand gestures.
2. **Hand Detection & Landmark Extraction:**
   Advanced libraries like MediaPipe and OpenCV are used to detect, track, and extract

precise hand landmarks (positions and features such as finger bends or hand orientation). This ensures stable and robust feature acquisition crucial for gesture recognition.

3. **Gesture Recognition:**
   The extracted hand features are processed by a deep learning model based on CNN (Convolutional Neural Network) or LSTM (Long Short-Term Memory) architectures, built with frameworks such as Keras or TensorFlow. This model learns spatial and temporal patterns in hand movements to classify the sign directly as an ASL word.
4. **Word Mapping:**
   Recognized gestures are mapped to corresponding English words using a pre-defined dictionary, providing real-time translation.
5. **Text-to-Speech (TTS):**
   The final mapped word is then converted into audible speech using a TTS module, allowing non-signers to instantly understand what the user intended to communicate.

Key Components

- **Camera Module:** Captures video frames for live gesture input.
- **Hand Detection & Landmark Extraction:** Uses MediaPipe/OpenCV for accurate hand tracking.
- **Gesture Recognition Model:** Deep learning models (CNN/LSTM with TensorFlow/Keras) for sign classification.
- **Word Mapping:** Translates detected gesture to its English meaning.
- **Text-to-Speech:** Provides audio feedback for the translated text.

Project Requirements

Hardware

- **Webcam:**
  Essential for capturing real-time video of the user's hand gestures.

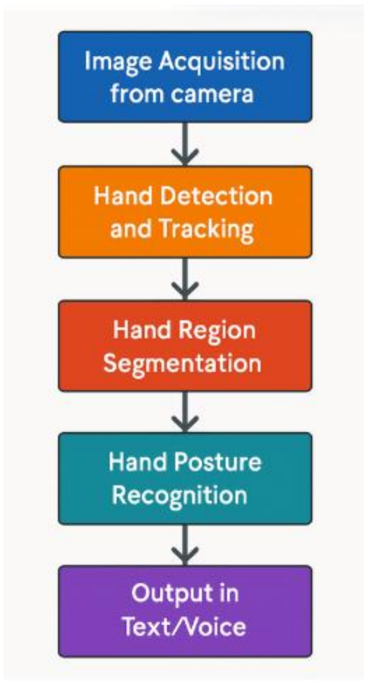
Software

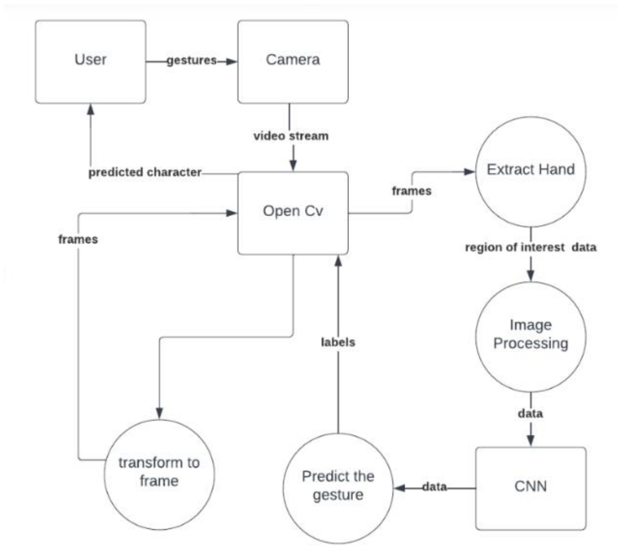
- **Operating System:**
  Windows 8 or above (ensures compatibility and performance).
- **IDE:**
  PyCharm (recommended for Python development).

- **Programming Language:**
  Python 3.9 (or later versions).

Python Libraries

- **OpenCV (cv2):**
  Used for video capture, preprocessing frames, drawing bounding boxes/landmarks, and general image processing.
- **NumPy (np):**
  Handles numerical computations, array manipulations, image conversions, and supports advanced mathematical operations required for image and model processing.
- **MediaPipe (mp):**
  Efficiently detects and tracks hand landmarks in real time, simplifying feature extraction essential for recognizing ASL signs.
- **TensorFlow (tf):**
  Powers deep learning operations—building, training, and deploying neural networks for gesture classification.
- **Keras:**
  Provides a high-level interface for model design, training, and evaluation using TensorFlow as the backend. Makes deep learning accessible and robust.





6. Methodology

Step 1: Data Collection

A robust data collection plan is essential for building an accurate and inclusive ASL recognition system. Below is a detailed breakdown of each component in the plan:

1. Creating Our Own Dataset

- **Approach:**
  The project will involve recording hand gestures for each letter of the ASL alphabet using a diverse group of volunteers.
- **Process:**
  Multiple samples (repetitions) for each letter will be captured to account for natural variation in signing style, hand movement, and position.
- **Goal:**
  To ensure a comprehensive dataset that represents real-world diversity and supports model generalization.

2. Using Existing Datasets

- **Approach:**
  The initial phase will leverage publicly available ASL datasets for model training.
- **Purpose:**
  These existing datasets serve as the foundation to quickly develop and validate the base recognition model before collecting custom data.
- **Benefit:**
  Saves time and provides a proven starting point for further improvements.

3. Devices

- **Devices Used:**
  - Webcams
  - Smartphone cameras
- **Reason:**
  These devices are easily accessible, affordable, and widely used, making it

straightforward for participants to contribute data and for deployment in real-world settings.

---

4. Recording Conditions

- **Initial Setup:**
  Data collection will start in well-lit environments with simple, non-distracting backgrounds.
- **Expansion:**
  Recordings will gradually include varied settings to improve the robustness of the recognition system for real-world scenarios (different lighting, cluttered backgrounds, etc.).
- **Impact:**
  This ensures the model performs reliably, regardless of user surroundings.

---

5. Participant Diversity

- **Inclusivity:**
  The data collection will deliberately include:
  - Different ages
  - Various skin tones
  - Hand sizes
  - Multiple signing styles
- **Purpose:**
  This diversity helps the system learn variations and avoid bias, making the model more accurate and fair for all potential users.

---

6. Data Labeling & Organization

- **Labeling:**
  Each recorded sample will be explicitly labeled by:
  - The ASL letter
  - The signer identity (anonymous ID or demographic category)
- **Organization:**
  Data will be sorted in folders by label for efficient access and management during model training.

- **Outcome:**
  Well-organized data accelerates model development and reduces errors.

---

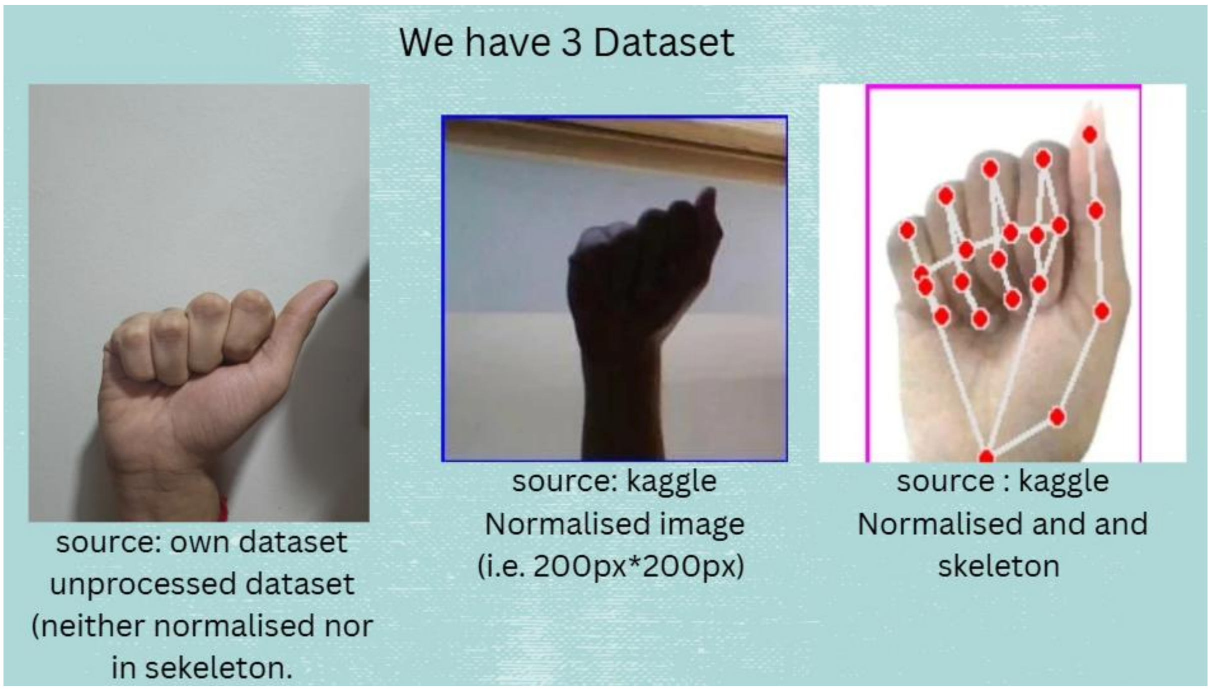
7. Data Enhancement

- **Hand Landmark Extraction:**
  Tools like MediaPipe will be used to generate “skeleton” images of hand gestures, which are less susceptible to noise and more useful for machine learning.
- **Augmentation Techniques:**
  - Image flipping
  - Brightness adjustments
- **Objective:**
  These methods increase the variety and volume of data, helping the model learn to recognize gestures under different conditions.

---

8. Quality Control & Ethics

- **Quality Review:**
  All collected and sourced data will be checked for clarity, accuracy, and usefulness.
- **Ethics:**
  Informed consent will be obtained from all participants, respecting their privacy and rights.
- **Hybrid Data Approach:**
  By combining existing public datasets and newly collected, annotated data, the system is designed to be:
  - Accurate
  - Reliable
  - Adaptable to changing needs



We have 3 Dataset

source: own dataset
unprocessed dataset
(neither normalised nor
in sekeleton.

source: kaggle
Normalised image
(i.e. 200px*200px)

source : kaggle
Normalised and and
skeleton

Step 2: Preprocessing

Effective preprocessing ensures that data fed into an ASL recognition system is clean, meaningful, and suitable for robust model learning. Below is a comprehensive explanation of each step involved in preprocessing for this project:

---

1. Hand Detection & Segmentation

- **Process:**
  The hand region is detected within each video frame or image using MediaPipe.
- **Purpose:**
  Focuses only on relevant hand gestures while ignoring unnecessary background elements.
- **Impact:**
  Reduces noise, which helps the recognition model concentrate on gesture details rather than distractions.

---

2. Landmark Extraction

- **Process:**
  MediaPipe extracts detailed key points (landmarks) on the fingers and palm from the isolated hand image.

- **Purpose:**
  Creates a “skeleton” representation of the hand, making the features robust to lighting changes and background variations.
- **Impact:**
  This precise extraction is crucial for reliable deep learning-based gesture recognition.

---

3. Skeleton Image Generation

- **Process:**
  Extracted landmarks are plotted onto a plain white background.
- **Purpose:**
  Removes complexity and color from scenes, distilling focus to the hand structure.
- **Impact:**
  Improves recognition accuracy by simplifying input data.

---

4. Image Normalization and Resizing

- **Process:**
  All images or skeleton plots are standardized to a fixed size (e.g., 64x64 pixels).
- **Pixel Value Normalization:**
  Pixel values are normalized (e.g., scaled between 0 and 1) to ensure uniform brightness and contrast.
- **Purpose:**
  Streamlines processing by making all data uniform for the model, and aids efficient learning.

---

5. Data Augmentation

- **Process:**
  Increases the diversity of training data by:
  - Flipping images horizontally or vertically
  - Rotating images at various angles
  - Adjusting brightness levels
- **Purpose:**
  Simulates real-world variability in gestures and recording conditions, improving model robustness.

---

6. Label Encoding

- **Process:**
  Each image is labeled according to the letter or gesture it represents.
- **Purpose:**
  Facilitates supervised learning, allowing the model to associate processed inputs with output classes.

---

7. Data Splitting into Training & Testing

- **Process:**
  After preprocessing, the dataset is divided:
  - Training Set: ~70% (used for model learning)
  - Validation Set: ~15% (used to tune model parameters and prevent overfitting)
  - Test Set: ~15% (used for unbiased evaluation of final model performance)
- **Purpose:**
  Ensures all classes (letters) are fairly represented and the model is properly validated on unseen data.

---

8. Quality Control

- **Process:**
  All samples are carefully reviewed to remove:
  - Blurry images
  - Mislabeled samples
  - Incomplete or low-quality data
- **Purpose:**
  Maintains rigorous data standards and maximizes training success.
- **Impact:**
  The model is trained on high-quality, diverse, and representative data.

Visual Example

The right side of the presentation visually demonstrates each preprocessing step:

- Raw hand images are detected and segmented.
- Landmarks are plotted to generate skeleton images.

- Images are resized and normalized.
- Data is labeled, augmented, split, and quality checked.

Step 3: Model Training

We use a **two-stage model**:
Visual Encoder (CNN) → Temporal Model (BiLSTM/Transformer) → CTC Decoder.

1. Visual Encoder

- **Backbone:** ResNet-34 or EfficientNet-B3
- Extracts **per-frame feature vectors** (e.g., 512-dim).
- Trained with spatial dropout to prevent overfitting.

2. Temporal Model

- **Option 1:** Bidirectional LSTM (BiLSTM) for sequential dependencies.
- **Option 2:** Transformer encoder for long-range dependencies.
- Input: sequence of frame features from CNN.

3. Decoder: CTC (Connectionist Temporal Classification)

- Outputs letter probabilities per time step without requiring manual alignment.
- Allows variable-length input sequences.

4. Training Configuration

- **Loss Function:** CTC loss.
- **Optimizer:** Adam (LR = 1e-4), cosine decay schedule.
- **Batch Size:** 16–32 sequences.
- **Regularization:** Weight decay, dropout (p = 0.3).
- **Training Epochs:** ~50–80 (with early stopping).

5. Semi-supervised Pretraining (optional)

- Pretrain encoder with autoencoders or contrastive learning using unlabeled video data.

Step 4: Evaluation

We evaluate using **signer-independent splits** — ensuring test signers are not in training data.

1. **Metrics**
   - **Character Error Rate (CER)** – Main metric for fingerspelling accuracy.
   - **Word Error Rate (WER)** – For sequences forming valid words.
   - **Frame-wise accuracy** – For intermediate model checks.
2. **Benchmarking**
   - Compare with baseline models from literature (e.g., Shi et al., 2018; 2019).
   - Test in different environmental conditions (lighting, camera angles).

Step 5: Deployment

We optimize for **real-time performance** on commodity hardware.

1. **Integration with OpenCV**
   - Capture live webcam feed.
   - Pass frames through preprocessing and model inference pipeline.
2. **Model Optimization**
   - Convert trained model to **ONNX** or **TensorRT** for speed-up.
   - Reduce model size via quantization (FP16 or INT8).
3. **Output Interface**
   - Display recognized text in real-time.
   - Optionally pass text to **Text-to-Speech (TTS)** engine for audio output.
4. **Latency Target**
   - Aim for <200 ms end-to-end processing per frame sequence.

---

7. Conclusion

Our system is designed to transform sign language, specifically the letters of the American Sign Language (ASL) alphabet, into meaningful words and spoken speech. This functionality significantly enhances communication for the deaf and hard-of-hearing community by providing them with a seamless way to express themselves in English, both visually and audibly.

At the core of the system is an advanced hand detection mechanism that accurately captures the user's hand gestures in real time. Utilizing state-of-the-art computer vision techniques and MediaPipe technology, the system isolates and analyzes the hand's position and movements, focusing precisely on the relevant gestures without distractions from the background. This precise detection is critical to ensuring that the signs are correctly interpreted.

To achieve reliable and accurate recognition, the system incorporates deep learning models—specifically convolutional neural networks (CNN) and long short-term memory (LSTM) networks—that have been trained on a comprehensive dataset of sign language gestures. This dataset is a hybrid of publicly available ASL sign data combined with newly collected, diverse samples. By blending existing and custom data, the model achieves robustness and generalizes well across different users, hand shapes, skin tones, and environments, ensuring inclusivity and reducing bias.

Once the system recognizes the individual sign language letters, it assembles them into words and translates these into textual form. To make communication even more natural and accessible, the translated text is passed through a text-to-speech (TTS) engine. This generates clear, human-like spoken English audio output, enabling non-signers to understand the message effortlessly. This auditory feedback also helps deaf users verify that their intended message is being correctly conveyed.

Looking ahead, the system's development roadmap includes the addition of air-written word recognition capabilities. This feature will enable users to "write" words in the air using continuous hand gestures, making the system even more user-friendly and versatile. By recognizing whole words or phrases formed dynamically in space, the platform will reduce the need for spelling out each letter, thereby speeding up communication and making it more fluid.

In summary, this system integrates intelligent hand detection, advanced deep learning, comprehensive datasets, and natural text-to-speech to bridge communication gaps effectively. Its evolving capabilities aim to empower the deaf community with a practical, user-centric tool for clear, fast, and natural interaction with hearing individuals, enhancing accessibility and inclusivity in everyday life.

8. References

No.	Reference	Paper Title / Topic
1	Samonte, M.J.C., et al. (2022)	Using Deep Learning in Sign Language Translation to Text
2	Abey Abraham, Rohini V (2018)	Real time conversion of sign language to speech...

No.	Reference	Paper Title / Topic
3	Deepika Patil et al (2024)	Sign Language to Text Conversion
4	Shubham Thakar et al (2024)	Sign Language to Text Conversion in Real Time using Transfer Learning
5	Satvika Reddy Satti, Pavithra (2023)	AUDIO OR TEXT TO SIGN LANGUAGE CONVERTER
6	Neha Ganesh Karande et al (2024)	Air Writing Word Recognition and Translation
7	Amith K R et al (2024)	Air Writing Detection and Recognition
8	Sanaa Mohsin et al (2024)	American Sign Language Recognition Based on Transfer Learning Algorithms
9	Abhishek Tripathi et al (2025)	Intelligent Sign Language Recognition for Real-Time Text Conversion...