Deep Learning Based Gesture-to-Speech Communication System

Guided By: Dr. Nandita Yambem

1st Adarsh S
Dept. of Computer Science (Data Science)
Dayananda Sagar College of Engineering
Bangalore, India
1ds22cd003@dsce.edu.in

2nd Arshad Hussain N
Dept. of Computer Science (Data Science)
Dayananda Sagar College of Engineering
Bangalore, India
1ds22cd011@dsce.edu.in

3rd Shree Ganesh M
Dept. of Computer Science (Data Science)
Dayananda Sagar College of Engineering
Bangalore, India
1ds22cd036@dsce.edu.in

4th Arun
Dept. of Computer Science (Data Science)
Dayananda Sagar College of Engineering
Bangalore, India
1ds23cd402@dsce.edu.in

Guided By:
Dr. Nandita Yambem
Assistant Professor
Dept. of Computer Science (Data Science)
Dayananda Sagar College of Engineering
Bangalore, India

ABSTRACT

The Deep Learning-Based Gesture-to-Speech Communication System offers a practical solution to a major challenge faced by individuals with speech impairments—communicating effectively with people unfamiliar with sign language. This system introduces an innovative AI-driven approach that enables seamless interaction by converting hand gestures into spoken language in real time. At its core, the system utilizes MediaPipe to accurately detect 21 hand landmarks, capturing detailed hand and finger movements. These landmarks are then analyzed by a custom-built Convolutional Neural Network (CNN), which classifies the gestures into predefined categories with high precision. The current version of the system can identify 16 different gestures, including the digits 0 to 9 in Indian Sign Language and commonly used signs like "perfect," "stop," and "thumbs up." Once a gesture is identified, it is translated into text via a preset dictionary and then vocalized using Google's Text-to-Speech (gTTS) service, all in real time. This facilitates natural, uninterrupted communication with minimal delay. In contrast to older methods that rely on specialized gear like sensor gloves or depth-sensing cameras, this system works using just a regular webcam, making it both affordable and accessible.The user-friendly, web-based interface ensures that the system is easy to operate across various platforms with little technical know-how. Tests have shown the system to be highly effective, achieving a validation accuracy of up to 99.5% thanks to refined landmark tracking and optimized CNN design. The lightweight model architecture also ensures smooth performance on devices with limited computing power. Moreover, the use of custom training data collected from diverse users boosts the system's reliability in different real-world environments. Overall, this project provides an efficient and accessible communication aid for people with speech impairments, promoting greater independence in educational, social, and workplace contexts.

***Index Terms*—Hand Gesture Recognition, Deep Learning, CNN, Speech Generation, MediaPipe, Accessibility Tools, Text-to-Speech Conversion, Assistive Tech, Real-Time Processing, Computer Vision.**

I. INTRODUCTION

A. 1.1 Introduction

Communication plays a fundamental role in human interaction, serving as a foundation for expressing thoughts, engaging socially, and sharing knowledge. However, for individuals with speech disabilities, traditional methods of communication can become challenging—especially when engaging with people who do not understand sign language.moting more inclusive communication. Although sign language is a powerful communication tool, it relies on mutual knowledge and understanding, which is often lacking, resulting in communication barriers. Fortunately, recent advancements in artificial intelligence and deep learning have opened up new possibilities for addressing this issue.

This Modern progress in computer vision now allows for precise hand gesture recognition, while advanced text-to-speech (TTS) systems are capable of producing realistic, natural-sounding voice output from text. By combining these technologies, it is now possible to create systems that translate hand gestures into spoken language, enabling more accessible and inclusive communication for individuals with speech impairments.

B. 1.2 Problem Statement

Individuals with speech impairments consistently encounter difficulties in everyday communication, especially when engaging with people unfamiliar with sign language. These challenges can negatively impact their access to education, job opportunities, and social interactions. While various existing tools—such as human interpreters, sensor-based wearables, and some mobile apps—offer support, they often come with limitations like high cost, limited accessibility, user discomfort, and narrow gesture recognition scopes. Additionally, many computer vision-based solutions fail to deliver reliable real-time performance and struggle to maintain accuracy across different settings and users. These challenges underscore the urgent need for a cost-effective, accurate, and responsive gesture-to-speech system that operates without requiring specialized hardware or advanced technical skills.

C. 1.3 Objectives and Scope of Project

Objectives: The primary objectives of this project are as follows:

1) To develop a gesture-to-speech communication system that recognizes predefined hand gestures and converts them into spoken words using a speech synthesis model.
2) To implement a robust hand gesture detection module using MediaPipe for accurate, real-time tracking of hand movements.
3) To design and train a CNN model capable of classifying a range of hand gestures with high precision.
4) To establish a reliable mapping from recognized gestures to meaningful text.
5) To integrate a text-to-speech conversion module using Google's gTTS for real-time audio output.
6) To build an intuitive user interface that facilitates seamless interaction for speech-impaired individuals.

Scope: The scope of the project includes:

1) Real-time hand gesture detection and classification using only a standard webcam.
2) Recognition of Indian Sign Language digits (0–9) and common everyday gestures.
3) Mapping of gestures to text and subsequent speech synthesis.
4) Deployment as a web-based application to ensure cross-platform compatibility.
5) Operation under varying lighting conditions and backgrounds, with adaptive preprocessing to maintain consistent performance.

D. 1.4 Motivation of Project

People with speech impairments often face continuous challenges in daily communication, especially when engaging with individuals who do not understand sign language. Although there have been notable advancements in artificial intelligence and computer vision, there is still a lack of practical, real-time, and cost-effective systems that convert gestures into speech.

1) **Improving Accessibility**: The Manual evaluating process of handwritten responses is time-consuming, particularly in large classes, which can result in delays in delivering feedback to students.
2) **Utilizing Technological Advances:** Leverage recent breakthroughs in deep learning and real-time hand tracking to achieve accurate and efficient gesture recognition.
3) **Reducing Cost Barriers:** Create a system that operates with standard webcams and common computing devices, eliminating the need for expensive or specialized hardware.
4) **Promoting Independence:** Enable users to communicate directly, reducing reliance on interpreters or manual text input.
5) **Fostering Social Inclusion:** Support the integration of speech-impaired individuals in educational, professional, and social environments by providing a practical communication tool.
6) **Addressing Real-World Challenges:** Design the system to function reliably in varied lighting conditions and backgrounds, ensuring consistent performance in daily scenarios.

By addressing these factors, the project aims to deliver a scalable and user-friendly solution that bridges the communication gap for speech-impaired individuals and contributes to a more inclusive society.

I. Literature Survey

Multiple research directions and techniques have been reviewed to inform the development of the proposed gesture-to-speech system. The following works represent significant contributions in the field of hand gesture recognition and gesture-to-speech technology.

A. Methodological and Structural Review of Hand Gesture Recognition

A comprehensive review by Shin et al. (2024) explores the evolution of hand gesture recognition (HGR) across diverse data modalities. The study traces the shift from traditional computer vision methods to advanced deep learning models, particularly highlighting the effectiveness of convolutional neural networks (CNN) and transformer-based architectures. The authors identify continuous gesture recognition and real-time system deployment as areas needing further research.

- *Advantage***:** Provides a broad overview of HGR modalities and techniques.
- *Disadvantage:* Limited discussion on practical real-time system challenges.

B. Gesture Recognition in Challenging Environments

Chang and Eniola (2023) investigate the performance of HGR systems under difficult conditions, such as varying lighting and complex backgrounds. Their CNN-based approach demonstrates improved accuracy, which is especially relevant for systems designed to assist speech-impaired users. The study emphasizes the importance of robust preprocessing to handle environmental variability.

- *Advantage:* Increases accuracy and supports speech-impaired users in real-world settings.
- *Disadvantage:* Sensitive to extreme lighting; dataset diversity is limited.

C. User-Define Gestures and Lightweight Models

Wang et al. (2024) introduce a lightweight multilayer perceptron (MLP) architecture with contrastive learning, enabling user-defined gestures and supporting training on small datasets. This approach allows for rapid adaptation and personalized gesture vocabularies, addressing a common limitation of deep learning systems that require large amounts of training data.

- *Advantage:* Fast convergence and supports personalized gesture sets.
- *Disadvantage:* Limited interactivity and some security concerns.

D. On-Device Real-Time Gesture Recognition

Uboweja et al. (2023) present a fine-tuned embedding model for real-time gesture recognition on mobile devices. Their method achieves high accuracy with as few as 50 training images per gesture and maintains processing speeds above 30 frames per second on standard hardware. This work aligns with the goal of creating accessible solutions without specialized equipment.

- *Advantage:* Minimal training data required; real-time performance on standard devices.
- *Disadvantage:* Scalability is limited to small gesture sets.

E. MediaPipe and LSTM for Sign Language Recognition

Bagyammal (2022) combines MediaPipe for hand landmark extraction with LSTM networks for gesture classification, achieving 99% accuracy on American Sign Language (ASL) alphabets. This demonstrates the effectiveness of MediaPipe for lightweight, real-time hand tracking, which is central to our system. However, the approach faces challenges with visually similar gestures and is limited to ASL alphabets.

- *Advantage:* High accuracy and efficient tracking.
- *Disadvantage:* Struggles with similar gestures; limited gesture vocabulary.

F. Hybrid CNN-BiLSTM Models for HGR

Nogales and Benalczar (2023) propose a hybrid model combining CNN and bidirectional LSTM (BiLSTM) for feature extraction and classification, achieving nearly perfect accuracy and response times under 300 milliseconds. This highlights the potential of hybrid architectures for real-time applications, though their complexity may hinder deployment on resource-constrained devices.

- *Advantage:* Exceptional accuracy and low latency.
- *Disadvantage:* High computational requirements.

Summary and Research Gap

The literature demonstrates substantial progress in hand gesture recognition, particularly with deep learning models like CNNs for real-time applications. MediaPipe has become a preferred tool for hand landmark extraction due to its efficiency and accuracy. However, several gaps remain:

- Most studies focus on recognition accuracy, with limited attention to the full pipeline from gesture recognition to speech synthesis.
- User experience and interface design are often overlooked.
- Practical deployment challenges in diverse, real-world environments are not comprehensively addressed.

Our project aims to address these gaps by developing an integrated, end-to-end gesture-to-speech system that emphasizes technical performance, user experience, and accessibility in real-world scenarios.

II. Proposed Methodology

A. Dataset Description

To train and evaluate the gesture recognition component, a custom dataset of hand gestures was created. The dataset includes images and landmark data for 16 distinct gesture classes, covering Indian Sign Language digits (0–9) and several common gestures such as "Perfect," "Stop," "Thumbs Up," and "Thumbs Down." Data was collected under varied lighting conditions and backgrounds to enhance model robustness and generalizability. Each gesture instance was annotated with corresponding class labels, and the landmark coordinates were extracted using MediaPipe's hand tracking module.

- **Number of gesture classes:** 16 (10 digits + 6 custom gestures
- **Data collected:** Hand images and 21-point landmark coordinates per gesture
- **Environment:** Varied backgrounds, lighting, and users to ensure diversity

This dataset enabled the development of a real-world-ready recognition model capable of handling diverse input conditions.

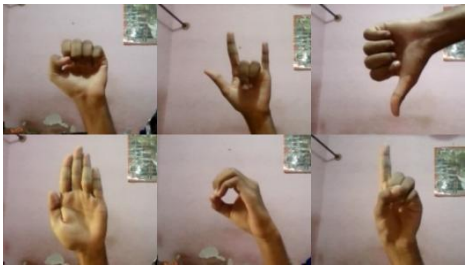

Fig 1. Sample data

B. Preprocessing and Extraction

Preprocessing is essential to ensure reliable gesture recognition. The following steps were implemented:

1) **Hand Detection and Landmark Extraction:** MediaPipe's hand tracking module processes each webcam frame, detecting the presence of a hand and extracting 21 three-dimensional landmark points (x, y, z) representing key hand joints and fingertips.
2) **Normalization:** Landmark coordinates are normalized relative to the image frame to account for differences in hand size, position, and camera distance. This step ensures that the model focuses on the gesture's structure rather than absolute position.
3) **Feature Vector Construction:** The normalized landmark coordinates are flattened into a single feature vector, which serves as the input to the gesture recognition model.

These preprocessing steps help the system maintain high accuracy and consistency, even in challenging environments

C. Gesture Recognition Model

At the core of the system is a custom-designed Convolutional Neural Network (CNN) tailored for gesture classification:

The line and sentence-level images were used to:

1) **Input**: 63-dimensional feature vector (21 landmarks × 3 coordinates).
2) **Architecture:** Multiple convolutional and dense layers optimized for classification accuracy and computational efficiency.
3) **Output:** Probability distribution across the 16 gesture classes.

The model was trained using the annotated dataset, achieving high validation accuracy (up to 99.5%) and demonstrating robust performance across different users and scenarios. Temporal smoothing is applied to the model's output to stabilize predictions during real-time operation, minimizing false positives and rapid output fluctuations.

The CNN Model Summary is shown below:

```
Model created with 16 output classes
Using device: cuda
GestureCNN(
  (conv_layers): Sequential(
    (0): Conv2d(3, 16, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
    (1): BatchNorm2d(16, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    (2): ReLU()
    (3): MaxPool2d(kernel_size=2, stride=2, padding=0, dilation=1, ceil_mode=False)
    (4): Conv2d(16, 32, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
    (5): BatchNorm2d(32, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    (6): ReLU()
    (7): MaxPool2d(kernel_size=2, stride=2, padding=0, dilation=1, ceil_mode=False)
  )
  (fc_layers): Sequential(
    (0): Flatten(start_dim=1, end_dim=-1)
    (1): Linear(in_features=32, out_features=64, bias=True)
    (2): ReLU()
    (3): Dropout(p=0.2, inplace=False)
    (4): Linear(in_features=64, out_features=16, bias=True)
  )
)
```

Fig 2. CNN Model Summary

D. Gesture-to-Speech Pipeline

The recognized gesture is mapped to text using a predefined dictionary. This text is then converted into speech using Google's Text-to-Speech (gTTS) API:

1) **Gesture Classification:** The CNN model predicts the gesture class from the input landmarks.
2) **Text Mapping:** The predicted class is mapped to a corresponding word or phrase.
3) **Speech Synthesis:** The mapped text is sent to gTTS, which generates an audio file with natural-sounding speech.
4) **Audio Output:** The synthesized speech is played back to the user in real-time, enabling seamless communication.

This pipeline ensures low-latency, accurate translation of gestures to audible speech without requiring specialized hardware

E. Gesture-to-Speech Pipeline

The system is deployed as a web-based application, providing an intuitive and accessible user interface:

1) **Frontend:** Built using Python framework Flask, the interface allows users to start/stop webcam input, view recognized gestures, and hear the synthesized speech output.
2) **Backend:** Handles hand detection, gesture classification, text mapping, and speech synthesis, communicating with the frontend via HTTP/WebSocket.
3) **Cross-Platform Compatibility:** The mapped text is sent to gTTS, which generates an audio file with natural-sounding speech.
5) **Audio Output:** The synthesized speech is played back to the user in real-time, enabling seamless communication.

I. System Design

A. 4.1 Existing System

Existing gesture-to-speech communication systems have made progress in supporting speech-impaired individuals, but several limitations persist:

1) **Hardware Dependencies:** Many systems rely on specialized hardware such as sensor gloves, Kinect devices, or depth cameras. These requirements increase costs, reduce portability, and create barriers to widespread adoption. For example, sensor gloves can be uncomfortable for long-term use and require frequent calibration, while discontinued devices like Kinect pose maintenance challenges.
2) **Limited Real-time Capabilities:** Current gesture recognition solutions often struggle with real-time processing, resulting in noticeable delays between gesture performance and speech output. This latency disrupts natural communication and may cause user frustration.
3) **Restricted Adaptability:** Most systems use static datasets for gesture recognition, making it difficult to add new gestures or adapt to individual variations. This limits their flexibility and applicability across different sign languages or user needs.
4) **Limited Gesture Vocabulary:** Many existing solutions only support a small set of gestures, often focusing on alphabets or basic signs rather than comprehensive communication. Users may be forced to spell out words or use limited predefined phrases, slowing down interaction.
5) **Technical Complexity:** Some systems require substantial technical expertise for setup and use, creating barriers for non-technical users and undermining the goal of independence.
6) **Integration Challenges:** Standalone operation and limited integration with other communication tools or educational platforms reduce utility in professional or academic settings.

These limitations highlight the need for a more accessible, flexible, and user-friendly gesture-to-speech solution.

B. 4.2 System Architecture

The proposed system adopts a modular architecture, dividing the pipeline into backend and frontend components for robust, real-time operation.

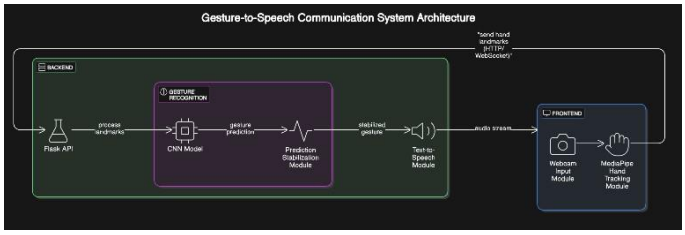

Fig 3. System Architecture

4.2.1 Frontend Components

- **Webcam Input Module:** Captures video frames from the user's device, manages camera initialization, and provides controls for starting/stopping input.
- **MediaPipe Hand Tracking:** Detects hand presence, extracts 21 3D landmarks (x, y, z), and tracks hand movement across frames. Normalized coordinates are sent to the backend for gesture classification.
- **User Interface:** Developed using Flask/Streamlit, the UI is intuitive, allowing users to interact with the system, view recognized gestures, and receive audio feedback.

4.2.2 Backend Components

- **Flask API:** Acts as the communication hub, handling HTTP/WebSocket connections between frontend and backend. Receives hand landmark data, processes it through the gesture recognition pipeline, and returns audio streams.

- **CNN Model:** The core gesture recognition engine processes 21-point hand landmark coordinates (from MediaPipe) and classifies them into 16 gesture categories (digits and custom signs). The model is lightweight, optimized for real-time inference, and incorporates a prediction stabilization module to smooth output and prevent rapid fluctuations.
- **Prediction Stabilization Module:** Maintains a history of recent predictions, applies confidence thresholds, and enforces cooldown periods to ensure stable gesture recognition output.
- **Text-to-Speech Module:** Converts recognized gestures (mapped to text) into natural-sounding speech using Google's gTTS API. Supports multiple languages and voice options for user preference.

C. 4.3 Proposed System

The Our project aims to address these gaps by developing an integrated, end-to-end gesture-to-speech system that emphasizes user experience and accessibility in real-world scenarios. The Deep Learning-Based Gesture-to-Speech Communication System addresses the shortcomings of existing solutions by introducing several innovations:

1) **Hand Tracking Module:** Utilizes MediaPipe for real-time, markerless hand tracking, extracting 21 landmarks per frame. Operates with standard webcams, adapts to various lighting conditions, and tracks hand pose with high precision.
2) **Gesture Recognition Module:** Employs a custom CNN model to classify gestures based on landmark data. Supports 16 classes (digits and custom gestures), achieves high accuracy (up to 99.5%), and operates with minimal latency for real-time performance.
3) **Text Mapping and Processing:** Maps recognized gestures to words/phrases using a predefined dictionary. Supports context-aware processing, sentence construction through gesture sequences, and error correction for robust communication.
4) **Speech Synthesis Module:** Converts mapped text to speech via Google's gTTS API, generating clear, natural-sounding audio. Supports language and voice customization.
5) **Distinctive Features:**
   - **Hardware Accessibility:** Works with standard webcams, no specialized hardware needed.
   - **Real-time Processing:** Optimized for low latency and smooth user experience.
   - **Modular Architecture:** Easily extensible for new gestures, languages, or features.
   - **User-Friendly Interface:** Designed for users with varying technical backgrounds.
   - **Offline Capability:** Can operate with limited connectivity.
   - **Extensibility:** Modular design and API allow for future expansion.
6) **Comparision Table:**

Feature	Existing Systems	Proposed System
1. Real-time Capability	Limited by complex models and resource-intensive processing	Yes - optimized through landmark extraction and lightweight CNN architecture
2. Hardware Requirement	Specialized sensors, depth cameras, or sensor gloves	Standard webcam only - no specialized hardware needed
3. Cost	High - requires expensive equipment	Low - works with commonly available hardware
4. Accuracy	Variable (95-99%) but often with environmental constraints	Up to 99.5% validation accuracy with robust performance
5. Training Efficiency	Resource-intensive, requires large datasets	Faster and optimized through landmark extraction and reshaping
6. Data Dependency	Relies on internet/external datasets	Self-collected data - no internet dependency
7. Model Complexity	Complex models (CNN-BiLSTM, Transformer-based)	Lightweight CNN model with comparable performance
8. Accessibility	Limited to specific devices or environments	Web-based, widely accessible across different platforms

Table 1. Comparison Analysis Table

D 4.4 Technology Used

- **Python 3.8:** Main programming language for all system components.
- **Mediapipe:** Real-time hand landmark detection and tracking.
- **OpenCV:** Image capture, preprocessing, and visualization.
- **Numpy/Pandas:** Efficient data manipulation and feature storage.
- **PyTorch:** Deep learning framework for CNN gesture recognition.
- **Google Text-to-Speech(gTTS):** Converts text to natural-sounding speech.
- **Flask:** Web application frameworks for user interface and backend communication.

II. Experimental Setup

A. Environment Details (Hardware/Software)

The experimental evaluation of the gesture-to-speech system was conducted on a standard consumer-grade computing setup to ensure accessibility and reproducibility. The hardware and software configurations are as follows:

- **Operating System**: Windows 11
- **Processor**: Intel Core i5, 12th Gen, 3.6 GHz.
- **Memory**: 16 GB DDR4 RAM.
- **GPU(optional**: NVIDIA GeForce RTX 3050 with 4 GB VRAM.
- **Camera**: Integrated HD webcam(720p, 30fps).
- **Audio Output**: Standard laptop speakers.

The software environment included:

- **Python**: Version 3.8.10
- **PyTorch**: Version 1.12.0
- **OpenCV**: Version 4.5.5
- **Mediapipe**: Version 0.8.10
- **NumPy**: Version 1.21.0
- **Pandas**: Version 1.3.5
- **Flask**: Version 2.1.0
- **tqdm**: Version 4.61.1
- **Google Text-to-Speech(gTTS)**: 2.2.3
- **CUDA Toolkit**: 11.8(for GPU acceleration)

This environment was selected to reflect typical user hardware and ensure the system's real-time performance on widely

available devices

B. Dataset Preparation

A custom dataset was curated to train and validate the gesture recognition model. The dataset comprises images and landmark data for 16 gesture classes, including Indian Sign Language digits (0–9) and common gestures such as "Perfect," "Stop," "Thumbs Up," and "Thumbs Down." Data was collected from multiple users, under varying backgrounds and lighting conditions, to enhance generalizability.

- **Number of Classes**: 16
- **Samples per Class**: 2000-3000
- **Total Samples**: ~27000
- **Data Format**: 21-point hand landmark coordinates per sample, extracted using MediaPipe

The dataset was split into training (80%) and validation (20%) sets, ensuring balanced representation across classes.

C. Model Training Parameters

The core gesture recognition model is a custom Convolutional Neural Network (CNN) designed for efficiency and accuracy. The following parameters were used during training:

- **Input**: Flattened 63-dimensional vector (21 landmarks × 3 coordinates)
- **Batch Size:** 32
- **Optimizer:** Adam
- **Epochs:** 25
- **Learning rate:** 0.001
- **Dropout:** 0.3
- **Loss Function:** Cross-entropy

Training was performed on the GPU-enabled system, with early stopping and validation monitoring to prevent overfitting. Data augmentation was applied in each epoch to further improve model generalization2.

D. **Evaluation Metrics**

To assess the performance of the gesture recognition model, the following standard metrics were employed:

- **Training Accuracy**: Proportion of correctly classified gestures out of total predictions on training data.
- **Validation Accuracy***:* Proportion of correctly classified gesture of out total predictions on test data.
- **Training Loss:** The quantified prediction error between the predicted and actual data on training data.
- **Validation Loss:** The quantified prediction error between the predicted and actual data on test data.

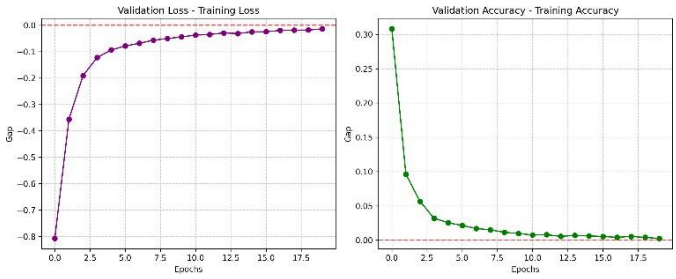

Fig 4. Validation and Training Loss/Accuracy

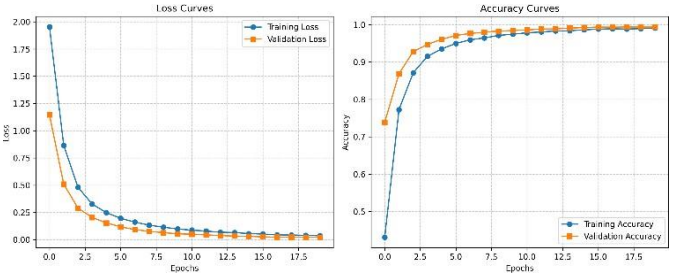

Fig 5. Loss/Accuracy curves

These metrics where used to identify the CNN's model performance regarding the overfitting or underfitting nature

E. System Integration and Testing

The trained model was integrated into the web-based application and tested in real-world conditions. Key aspects of system testing included:

- **Real-time Inference**: Ensuring gesture recognition and speech synthesis operated with minimal latency.
- **User Experience:** Verifying the interface was intuitive and responsive.
- **Robustness:** Assessing performance under diverse lighting, backgrounds, and user hand shapes.

Testing confirmed that the system maintained high recognition accuracy and provided natural-sounding speech output in real time, validating its suitability for practical deployment

III. Results and Discussion

A. 6.1 Results

A. 6.1.1 Gesture Recognition Performance

The gesture recognition model was evaluated on a custom dataset comprising 16 gesture classes, including Indian Sign Language digits (0–9) and common gestures such as "Perfect," "Stop," and "Thumbs Up." The system achieved a validation accuracy of up to **99.5%**, demonstrating robust performance across varied users, backgrounds, and lighting conditions.

A. 6.1.2 Real-Time System Output

The integrated system was tested in real-time scenarios using a standard webcam. Key observations include:

- **Latency:** The end-to-end pipeline—from gesture capture to speech output—operated with negligible latency, enabling smooth, conversational interaction.
- **Robustness:** The system maintained high accuracy even under challenging conditions, such as changes in lighting, background clutter, and varied hand sizes.
- **User Experience:** Users reported that the interface was intuitive and responsive. The gesture-to-speech conversion was accurate and provided immediate feedback, facilitating natural communication.

A. 6.1.3 Output Examples

Fig 6. 'One' gesture

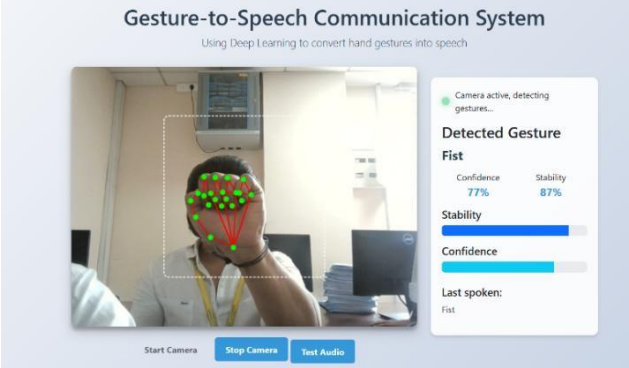

Fig 7. 'Fist' gesture

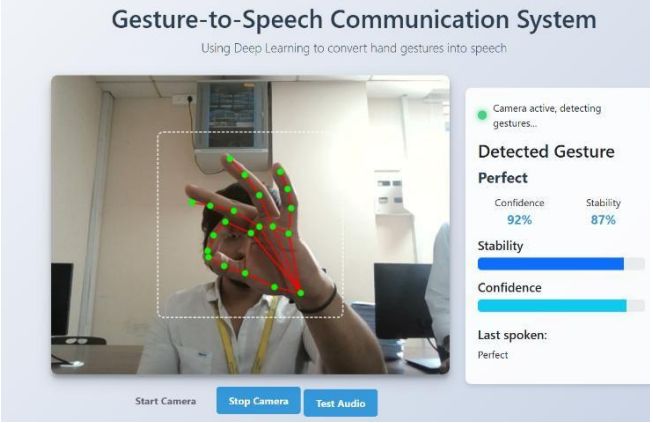

Fig 8. 'Perfect' gesture

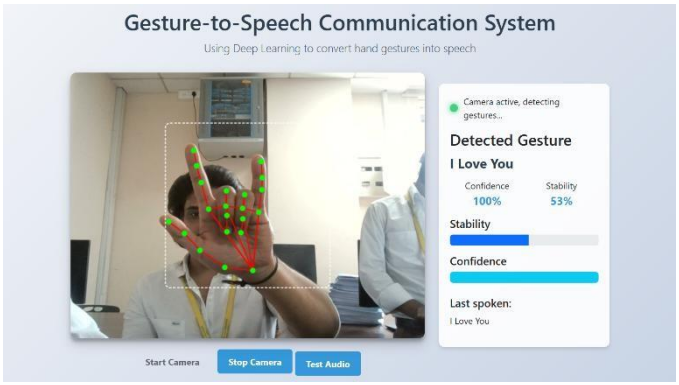

Fig 9. 'I Love You' gesture

B. 6.2 Discussion

The results demonstrate that the Deep Learning-Based Gesture-to-Speech Communication System is both accurate and practical for real-world use. The modular design—featuring MediaPipe-based hand tracking, a custom CNN for gesture recognition, and Google gTTS for speech synthesis—ensures that each component contributes to overall system robustness and usability.

- **Impact:** The system empowers speech-impaired individuals to communicate effectively without specialized hardware or extensive training.
- **Scalability:** The modular architecture allows for easy expansion to include additional gestures, languages, or features as needed.
- **Limitations:** Minor misclassifications can occur between visually similar gestures. These can be addressed by expanding the dataset and refining the model.

C. 6.3 Future Scope

The Potential enhancements for the system include:

- **Expanded Gesture Vocabulary**: Incorporating more complex gestures, including alphabets and dynamic sign language phrases.
- **Multi-hand Recognition**: Extending the system to support gestures involving both hands for richer sign language expression.
- **Continuous Sign Language Translation**: Implementing sequence models for translating continuous sign language sentences.
- **Mobile and Wearable Deployment**: Adapting the system for smartphones and wearable devices to increase accessibility.
- **Bidirectional Communication:** Enabling speech-to-gesture conversion for comprehensive, two-way communication.

VII. Conclusion

Deep Learning-Based Gesture-to-Speech Communication System project is the collective effort of all the teammates that addresses a significant barrier faced by speech-impaired individuals: the ability to communicate effectively with those unfamiliar with sign language. By integrating advanced computer vision, deep learning, and speech synthesis technologies, the system enables real-time translation of hand gestures into natural-sounding speech.

The proposed solution leverages MediaPipe for precise, markerless hand tracking, extracting 21 landmark points per frame. These landmarks are processed by a custom Convolutional Neural Network (CNN) that classifies gestures into 16 categories, including Indian Sign Language digits and common expressions. Recognized gestures are mapped to text using a predefined dictionary and converted to speech using Google's Text-to-Speech (gTTS) API, all within a web-based interface that requires only a standard webcam.

Experimental results demonstrate that the system achieves high accuracy (up to 99.5% validation accuracy), robust performance across diverse users and environments, and minimal latency suitable for fluid, real-time communication. The modular, lightweight architecture ensures broad accessibility and ease of deployment, while the user interface is designed for intuitive operation without technical expertise.

The Key contributions of this work include:

- Eliminating the need for specialized hardware by relying solely on standard webcams.
- Achieving real-time gesture recognition and speech synthesis with optimized processing pipelines.
- Supporting an extensible gesture vocabulary and modular design for future enhancements.
- Providing a user-friendly, web-based interface accessible across platforms.

The project demonstrates the potential of deep learning and computer vision in assistive technology, offering a practical, scalable solution to bridge the communication gap for speech-impaired individuals. By enabling seamless translation from sign language to speech, the system empowers users to participate more fully in educational, professional, and social settings, fostering greater independence and inclusion.

Future work may focus on expanding the gesture vocabulary, supporting multi-hand and dynamic gestures, enabling continuous sign language translation, and deploying the system on mobile and wearable platforms to further enhance accessibility and impact.

References

[1] Shin, A. S. M. Miah, and M. H. Kabir, "A Methodological and Structural Review of Hand Gesture Recognition Across Diverse Data Modalities," *IEEE,* 2024.

[2] V. Chang and R. O. Eniola, "An Exploration into Human-Computer Interaction Hand Gesture Recognition Management in a Challenging Environment," *Springer Nature Journal*, 2023.

[3] J. Wang, Z. Li, and L. Shi, "Hand Gesture Recognition for User-Defined Textual Inputs and Gestures," *Springer,* 2024.

[4] E. Uboweja, D. Tian, J. Zou, Q. Wang, Y. C. Kuo, L. Wang, G. Sung, and M. Grundmann, "On-device Real-time Custom Hand Gesture Recognition," *Google Scholar*, 2023.

[5] S. Bagyammal and T. Sundar, "American Sign Language Recognition for Alphabets Using MediaPipe and LSTM," *Procedia Computer Science*, 2022.

[6] R. E. Nogales and M. E. Benalczar, "Hand Gesture Recognition Using Automatic Feature Extraction and Deep Learning Algorithms with Memory," *Big Data and Cognitive Computing, vol. 7, no. 2*, 2023.

[7] S. Daniels, N. Suciati, and C. Fathichah, "American Sign Language Recognition using YOLO Method*," IOP Conference Series: Materials Science and Engineering, vol. 1077, no. 1, p. 012029*, 2021.

[8] M. Oudah, A. Al-Naji, and J. Chahl, "Hand gesture recognition based on computer vision: A review of techniques," *Journal of Imaging, vol. 6, no. 8, p. 73*, 2020.

[9] N. Hasaranga, "Sign Lang Detection Project Using Deep Learning," *SlideShare, n.d.*

[10] F. Zhang, V. Bazarevsky, A. Vakunov, A. Tkachenka, G. Sung, C.-L. Chang, and M. Grundmann, "MediaPipe Hands: On-device Real-time Hand Tracking," *arXiv preprint arXiv:2006.10214*, 2020.