EXPLORE INNOVATE EXCEL

PUNJAB ENGINEERING COLLEGE
(DEEMED TO BE UNIVERSITY)

Department of Computer Science and Engineering

MINOR PROJECT
REPORT

Sign-Talk

Under the Guidance of Dr. Manish Kumar
Punjab Engineering College (Deemed to be University)

Submitted By:-

Yashaswi Khurana (22106023)
Chetanya Mahana (22106008)
Anubhav Pandey (22106009)
Akshat Goel (22106039)
Shikhar Keshav (22106034)

DECLARATION

We hereby declare that the project work entitled **"Sign-Talk"** is an authentic record of our own work carried out at Punjab Engineering College (Deemed to be University), Chandigarh as per the requirements of “Minor Project” for the award of the degree of B.Tech. Computer Science and Engineering, under the guidance and supervision of Dr Manish Kumar.

We further declare that the information has been collected from genuine & authentic sources and we have not submitted this project report to this or any other university for the award of diploma or degree of certificate examination.

Certified that the above statement made by the students is correct to the best of my knowledge and belief.

Dr. Manish Kumar (Assistant Professor, Punjab Engineering College)

Date: 04/12/2024

Place: Punjab Engineering College, Chandigarh

Yashaswi Khurana (22106023)
Chetanya Mahana (22106008)
Anubhav Pandey (22106009)
Akshat Goel (22106039)
Shikhar Keshav (22106034)

EXPLORE INNOVATE EXCEL

PUNJAB ENGINEERING COLLEGE
(DEEMED TO BE UNIVERSITY)

Department of Computer Science and Engineering

CERTIFICATE

It is Certified that the Project work entitled **Sign-Talk** submitted by **Yashaswi Khurana, Chetanya Mahana, Anubhav Pandey, Akshat Goel and Shikhar Keshav** for the fulfillment of Minor Project offered by Punjab Engineering College (Deemed to be University) during the academic year **2023-24** is an original work carried out by the students under my supervision and this work has not framed any basis for the award of and Degree, Diploma or such other titles. All the work related to the project is done by these candidates themselves. The approach towards the subject has been sincere and scientific.

Date: 04.12.2024

Dr. Manish Kumar
CSE Department,

Punjab Engineering College (Deemed to be University)

ACKNOWLEDGEMENT

We would like to take this opportunity to thank our college Punjab Engineering College (Deemed to be University), Chandigarh and Department of Computer Science and Engineering for giving us an opportunity to work on this project.

We are immensely grateful to our project mentor Dr. Manish Kumar (Assistant Professor, Department of Computer Science and Engineering) whose continuous guidance, technical support and moral support at times of difficulty helped us to achieve milestones in the given time. She has been a great source of knowledge.

We are also thankful to the Committee for evaluation who gave their valuable feedback and guidelines for the betterment and completion of this project.

We convey our deep sense of gratitude to all teaching and non-teaching staff for their constant encouragement, support and selfless help throughout the project work. It is a great pleasure to acknowledge the help and suggestion, which we received from the Department of Computer Science and Engineering.

We wish to express our profound thanks to all those who helped us in gathering information about the project. Our families too have provided moral support and encouragement several times.

Yashaswi Khurana (22106023)
Chetanya Mahana (22106008)
Anubhav Pandey (22106009)
Akshat Goel (22106039)
Shikhar Keshav (22106034)

ABSTRACT

Sign language plays a crucial role in enabling communication for individuals with hearing or speech impairments. The SignTalk project aims to bridge the communication gap between sign language users and those unfamiliar with it by developing an efficient and accessible hand gesture recognition system. The system utilizes advanced computer vision techniques and machine learning algorithms to accurately detect and interpret both static and dynamic hand gestures.

The SignTalk framework integrates three distinct models to provide comprehensive recognition capabilities. The first model focuses on alphabet recognition based on American Sign Language (ASL). The second model interprets commonly used static gestures and incorporates a workflow that converts recognized gestures into text and speech. The third model is dedicated to recognizing dynamic gestures and translating them into text in real time. By leveraging tools like OpenCV and Mediapipe for hand tracking, and algorithms such as KNN, Multilayer Perceptron, and LSTM, SignTalk achieves robust performance and real-time interaction through webcam input.

To enhance the system's scalability and user experience, we are developing a Flask-based web application that integrates the functionalities of all three models. Additionally, data augmentation techniques, including geometric transformations and noise additions, are applied to expand the training dataset and improve model robustness.

The project delivers a versatile solution designed for inclusivity, enabling seamless communication in diverse settings such as educational institutions, public services, and personal interactions. Future extensions of SignTalk aim to expand its gesture library and enhance its applicability across multiple languages.

EXPLORE INNOVATE EXCEL

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

KNN	K-Nearest neighbors
CNN	Convolutional Neural Network
MLP	Multi layered Perceptron
NN	Neural Network
NLP	Natural Language Processing
ISL	Indian Sign Language
DL	Deep Learning
ASL	American Sign Language
ANN	Artificial Neural Network
ML	Machine Learning
RNN	Recurrent Neural Network
TL	Transfer Learning
LSTM	Long Short-Term Memory

CHAPTER 1: INTRODUCTION

Sign Talk aims to bridge the communication gap between individuals with hearing or speech impairments and the general public. This system translates sign language gestures into text or speech using advanced machine learning models. The project integrates three specialized models for recognizing sign language alphabets, static hand gestures, and dynamic gestures. These models are developed using cutting-edge machine learning techniques, ensuring high accuracy and real-time performance.

1.1 Motivation

Sign language is the primary mode of communication for millions of individuals with hearing or speech disabilities. However, the lack of widespread knowledge and understanding of sign language creates barriers to effective communication. This results in social isolation and difficulty in accessing essential services for those who rely on sign language.

Our motivation stems from the need to create an inclusive society where communication is accessible to everyone. By developing a system that translates sign language into text and speech, Sign Talk empowers individuals with disabilities, facilitating seamless interaction and fostering inclusivity.

Figure 1: Deaf Community Advocacy and Success

1.2 Problem Statement

While traditional sign language translation tools exist, they often face limitations such as poor accuracy, the inability to recognize dynamic gestures, or the lack of real-time functionality. Additionally, many systems focus solely on static gesture recognition, ignoring the complexity of dynamic gestures.

Problem:
To design and implement a robust system that accurately recognizes and translates:

Sign language alphabets (e.g., A, B, C, etc.).
Static hand gestures (e.g., OK, Not OK, Be Quiet, I Love You).
Dynamic gestures (e.g., Peace, Home, Thank You, Help).
This system must achieve high accuracy, work in real-time, and support a wide range of gestures.

1.3 Methodology

The development of Sign Talk involved the following systematic approach:

Dataset Creation:

Collected and labeled datasets for sign language alphabets, static gestures, and dynamic gestures.
Used both publicly available datasets and custom data captured via a webcam.
Initial Model Implementation (KNN):

Implemented KNN (K-Nearest Neighbors) as a baseline model for static and alphabet gesture recognition.
Achieved moderate results, but the model was too simplistic for complex gesture differentiation.
Model Enhancement (RNN and ANN):

For static gestures and alphabets, switched to RNN (Recurrent Neural Network) and ANN (Artificial Neural Network), significantly improving accuracy to 100%.
Dynamic Gesture Recognition (LSTM):

Developed an LSTM (Long Short-Term Memory) model for dynamic gesture recognition, handling sequential data effectively.
Achieved an accuracy of 84%, providing reliable real-time predictions.
Real-Time Integration:

Combined all three models into a unified system capable of recognizing gestures and converting them to speech in real-time using Python frameworks like TensorFlow and Flask.

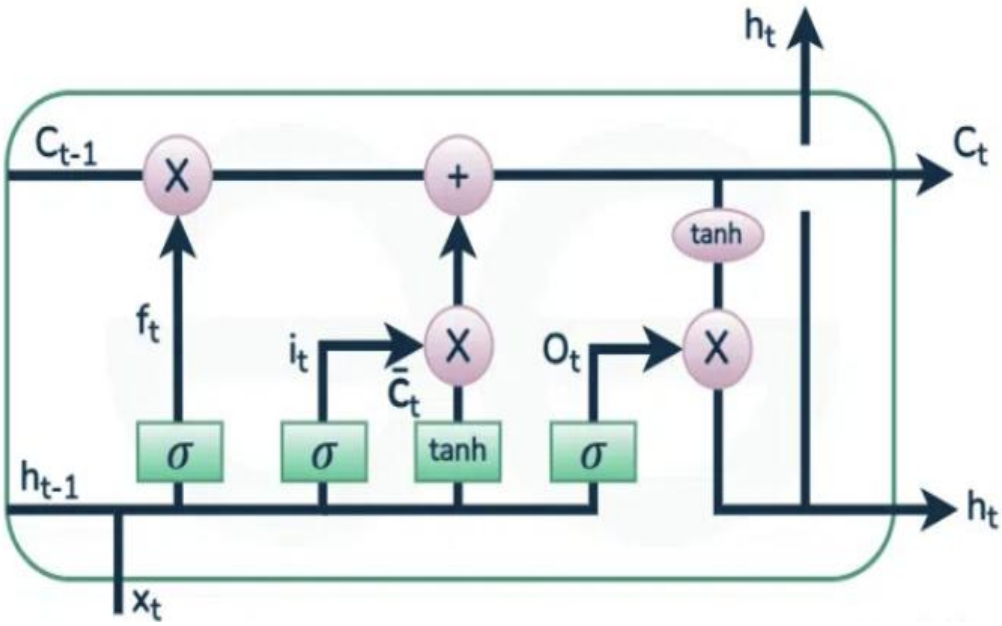

Figure 2: The LSTM Model in Gesture Recognition

CHAPTER 2: BACKGROUND

2.1 Overview of Sign Language Recognition

Sign language is a structured system of gestures, facial expressions, and body movements that serves as the primary mode of communication for individuals with hearing and speech impairments. While effective within communities fluent in sign language, communication barriers arise when interacting with individuals unfamiliar with the language. Over the years, researchers have sought to develop automated systems to translate sign language into text or speech, bridging this gap and enhancing inclusivity.

Modern advancements in machine learning and computer vision have enabled significant progress in this domain. Sign language recognition systems leverage powerful models to analyze gestures and translate them into meaningful outputs. These systems are broadly categorized into three components: recognition of individual alphabet signs, static gestures, and dynamic gestures. Each component presents unique challenges requiring tailored approaches for accurate and efficient interpretation.

Figure 3: American Sign Language (ASL) alphabet

2.2 Related Work

Existing research in the field of sign language recognition has primarily focused on specific subsets of gestures, such as static or alphabet-based recognition. Systems leveraging simpler models like KNN (K-Nearest Neighbors) have achieved moderate success but often fail to handle the complexity of real-world applications involving dynamic sequences. In contrast, advanced models like RNNs (Recurrent Neural Networks) and LSTMs (Long Short-Term Memory networks) have demonstrated superior performance in recognizing sequential and context-sensitive gestures.

However, limitations persist in terms of accuracy, scalability, and real-time applicability. Previous works have often relied on small datasets or datasets specific to a particular region, making these systems less adaptable to diverse linguistic and cultural sign languages. Additionally, many systems lack integration for comprehensive sign-to-speech translation, focusing instead on intermediate outputs like text-only translation.

2.3 Research Challenges

The development of a robust sign language recognition system poses several challenges:

Data Diversity: Sign languages vary significantly across regions and cultures, requiring extensive and diverse datasets to ensure inclusivity.
Static vs. Dynamic Gestures: While static gestures are relatively straightforward to recognize, dynamic gestures involve complex temporal patterns, necessitating sophisticated sequence modeling techniques.
Real-Time Performance: High accuracy must be achieved without compromising processing speed, particularly for dynamic gestures.
Generalization: The system should perform reliably across varying conditions, such as different lighting, backgrounds, and user variations in gesture execution.

2.4 Importance of the Project

Sign Talk addresses these challenges by integrating three distinct models tailored to different gesture types: alphabets, static gestures, and dynamic gestures. By achieving high accuracy and real-time performance, this project sets a new standard for assistive technology, empowering individuals with disabilities to communicate effectively and seamlessly in diverse environments.

The project's ability to translate sign language into both text and speech enhances accessibility, making it valuable not only for personal communication but also for professional, educational, and healthcare settings. This work lays the foundation for further innovations in inclusive technology, pushing the boundaries of what is possible in human-computer interaction.

CHAPTER 3: PROPOSED WORK

Our project, Sign Talk, integrates three distinct models to translate sign language gestures into text and speech. Each model is designed to handle specific gesture types—alphabet recognition, static gestures, and dynamic gestures—using tailored machine learning architectures. Below is a detailed explanation of each model and its implementation:

3.1 Alphabet Recognition and Static Gesture Model

3.1.1 Introduction to Alphabet and Static Gesture Recognition

Static gestures, such as individual sign language alphabets (A, B, C, etc.) and actions like "OK," "Not OK," and "Be Quiet," are characterized by stationary hand positions. These gestures are captured as images and processed using computer vision techniques to extract meaningful features.

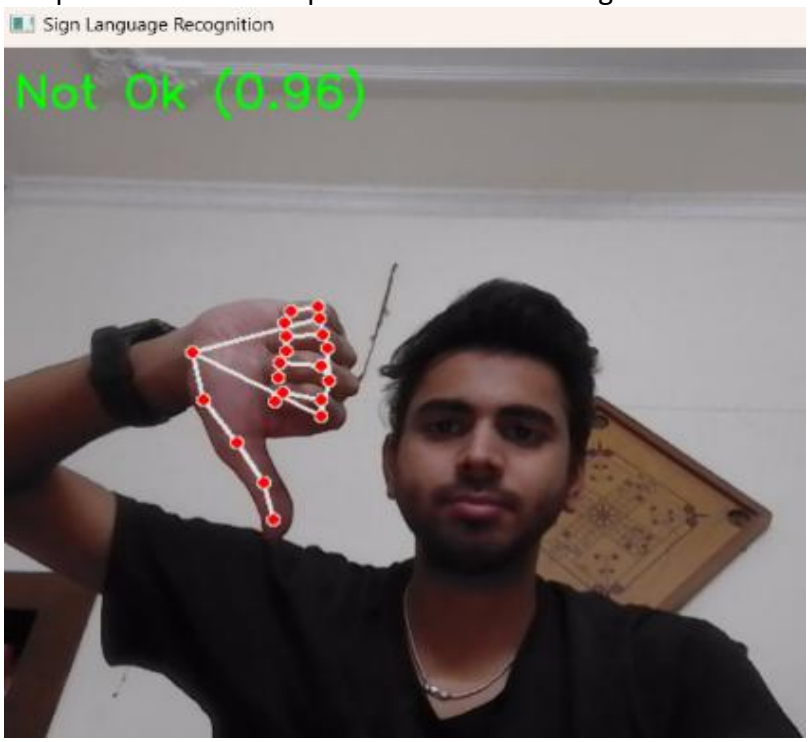

Figure 4: Gesture recognition

3.1.2 Data Preprocessing and Feature Extraction

Hand Landmark Detection:

MediaPipe Hands Library: We use MediaPipe to extract 21 key hand landmarks (e.g., joint positions) from the input video or images. These landmarks capture the structure and orientation of the hand.
The extracted landmarks include the x, y, and z coordinates of each key point, representing the spatial arrangement of the hand.
Normalization and Scaling:

The landmarks are flattened into a single vector for input into the classification model.
StandardScaler is applied to normalize the features, ensuring uniform scaling for efficient training.\

3.1.3 Model Implementation (ANN for Static Gestures)

Architecture:

A Multi-Layer Perceptron (MLP) is used, consisting of an input layer, two hidden layers (64 and 32 nodes), and an output layer.

Activation functions like ReLU are employed in the hidden layers, with a softmax output for classification.

Training and Inference:

The MLP is trained on a labeled dataset of static gestures, achieving 98% accuracy.

The model predicts gestures based on the input landmarks with real-time feedback displayed on the interface.

3.1.4 Customization and Flexibility

Users can add new gestures by capturing hand landmarks and labeling them with a gesture name.

The model retrains dynamically to incorporate new gestures, enhancing its adaptability.

3.2 Dynamic Gesture Recognition Model

3.2.1 Introduction to Dynamic Gestures

Dynamic gestures involve sequential movements, such as "Peace" or "Home." Recognizing these gestures requires understanding the temporal progression of hand positions over time.

3.2.2 Data Preprocessing and Temporal Feature Representation Sequence

Formation:

Input video frames are processed to extract hand landmarks for each frame. These landmarks are stored as a time-ordered sequence, representing the temporal evolution of the gesture.

Temporal Normalization:

To ensure uniformity, all gesture sequences are normalized to a fixed length using padding or interpolation techniques. This standardization is crucial for feeding consistent input into the LSTM model.

3.2.3 Model Implementation (LSTM for Dynamic Gestures):

Architecture:

The LSTM model consists of:

An input layer to accept sequential data (e.g., a sequence of hand landmarks).

Two LSTM layers with 128 and 64 units, enabling the model to capture temporal dependencies.

A dense output layer with softmax activation for gesture classification.

Training:

The LSTM model is trained on a dataset of dynamic gestures, achieving an accuracy of 84%.

The training process optimizes the model to understand gesture sequences and distinguish between similar gestures.

3.2.4 Real-Time Inference

The model processes input sequences from a webcam in real-time, ensuring minimal latency.

Recognized gestures are converted to text and speech using the Google Text-to-Speech (gTTS) API for user feedback.

3.3 Real-Time Integration and System Workflow

3.3.1 Real-Time Processing Video Input:

A webcam feed is processed frame by frame using OpenCV. Hand landmarks are detected for each frame and passed to the relevant model based on gesture type (static or dynamic). Classification and Feedback: For static gestures, the MLP classifier predicts the gesture from a single frame. For dynamic gestures, a sequence of frames is fed into the LSTM model for temporal analysis and classification. The recognized gesture is displayed on the screen, and its speech equivalent is generated using gTTS.

3.3.2 User Interaction

Users can interact with the system via a graphical interface that displays recognized gestures in text and provides audio feedback. New gestures can be added seamlessly, enhancing the system's adaptability to diverse user needs.

3.4 Tools and Technologies

- MediaPipe: For accurate hand landmark detection.
- OpenCV: For real-time video frame processing.
- MLP and LSTM Models: Built using Scikit-learn and TensorFlow/Keras libraries.
- gTTS (Google Text-to-Speech): For converting recognized gestures into speech.
- Python: The primary language for system integration and model implementation.

3.5 Summary of Model Performance

- Alphabet and Static Gesture Models (MLP):
- Accuracy: 98%, Key Strengths: High precision and flexibility to adapt to new gestures.
- Dynamic Gesture Model (LSTM):
- Accuracy: 84%, Key Strengths: Effective handling of temporal sequences and real-time processing.
- Sign Talk combines these models into a unified system, providing an inclusive and user-friendly solution for translating sign language into text and speech.

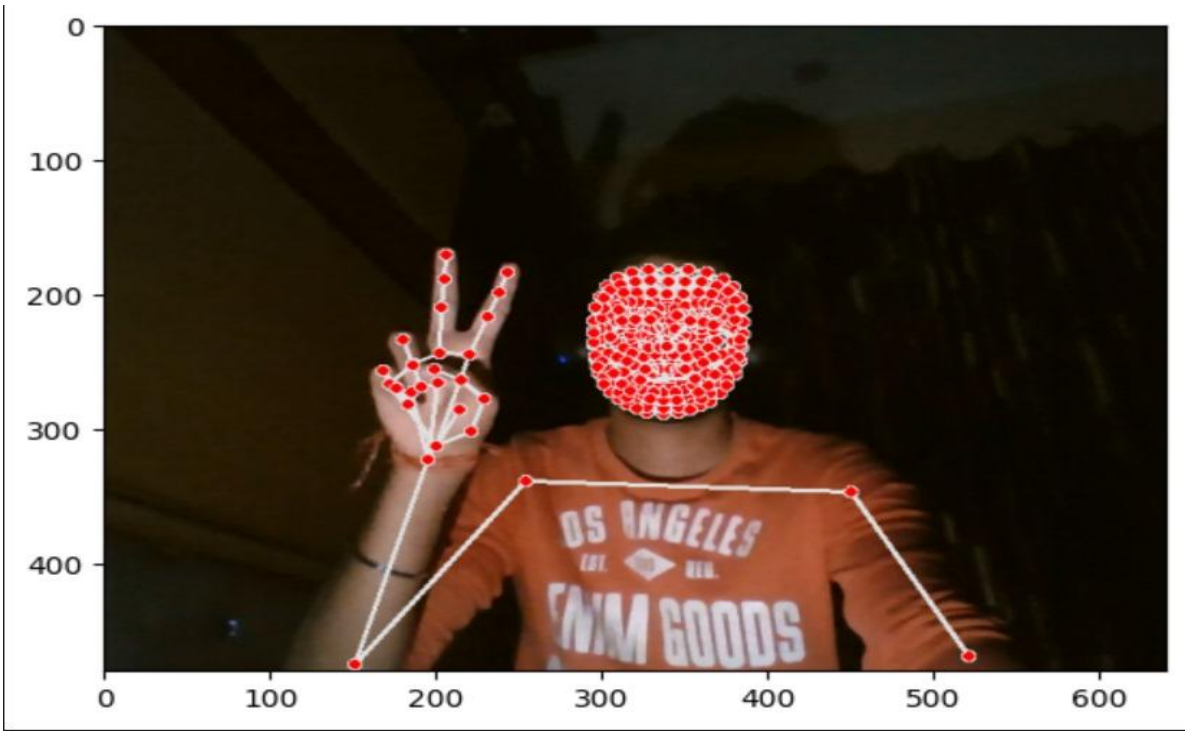

Figure 5: Landmarks Detection

Chapter 4: Implementation Details

This chapter elaborates on the implementation of the Sign Language Detection Model, emphasizing the use of the Mediapipe library for hand, pose, and face detection. Mediapipe's efficient and real-time capability ensures robust feature extraction for both static and dynamic gesture recognition.

4.1 Development Environment

- Programming Language: Python
- Frameworks and Libraries:
  1. Gesture Recognition Models: Scikit-learn (KNN), TensorFlow/Keras (MLP, LSTM)
  2. Feature Extraction: Mediapipe Hands, Pose, and Face
  3. OpenCV (for video and image processing)

4.2 Data Collection and Preprocessing

4.2.1 Data Sources

- Static Gestures: Public datasets with hand postures for American Sign Language (ASL) alphabet and common static symbols.
- Dynamic Gestures: Video datasets capturing sequential movements corresponding to sign language phrases or actions.

4.2.2 Mediapipe Feature Extraction

Mediapipe is used to extract meaningful landmarks for hands, pose, and face:

- Hands:
  1. Detects 21 landmarks on each hand.
  2. Captures the relative positions and orientations of fingers and palm.
  3. Key features: Thumb angles, finger spacing, and palm orientation.
- Pose:
  1. Detects 33 body landmarks.
  2. Used to capture upper-body posture and movement.
  3. Key features: Shoulder and elbow positioning for gestural context.
- Face:
  1. Detects 468 facial landmarks.
  2. Used to infer additional context, such as head tilt or facial expressions.
  3. Key features: Jaw and eyebrow positions for emotion-linked gestures.

4.2.3 Preprocessing Steps

- Data Augmentation:
  1. Variations in lighting, rotation, and scaling are introduced to improve model robustness.
- Normalization:
  1. All landmarks are normalized with respect to the bounding box or frame dimensions to ensure consistency across datasets.
- Sequence Formation for Dynamic Gestures:
  1. Each video is split into frames, and the landmarks (hands, pose, and face) from each frame are compiled into a time-series sequence.
  2. Padding or truncation is applied to standardize sequence lengths.

4.3 Model Implementation

4.3.1 Static Gesture Models

- K-Nearest Neighbors (KNN):
  1. Feature vectors composed of hand landmarks (21 points per hand) are input.
  2. Distance metrics (Euclidean) are used to find the k-nearest neighbors.
  3. Classification based on the majority label of the neighbors.
  4. Tools Used: Scikit-learn.
- Multilayer Perceptron (MLP):
  1. Input: Flattened feature vector from Mediapipe hand, pose, and face landmarks.
- Architecture:
  1. Input Layer: Vector combining hand (42 points), pose (33 points), and face (selected 10 points).
  2. Hidden Layers: Dense layers with ReLU activation for non-linear classification.
  3. Output Layer: Softmax activation for gesture classification.
- Training Parameters:
- Optimizer: Adam.
- Loss Function: Categorical Cross-Entropy.
- Epochs: 50.
- Tools Used: TensorFlow/Keras.

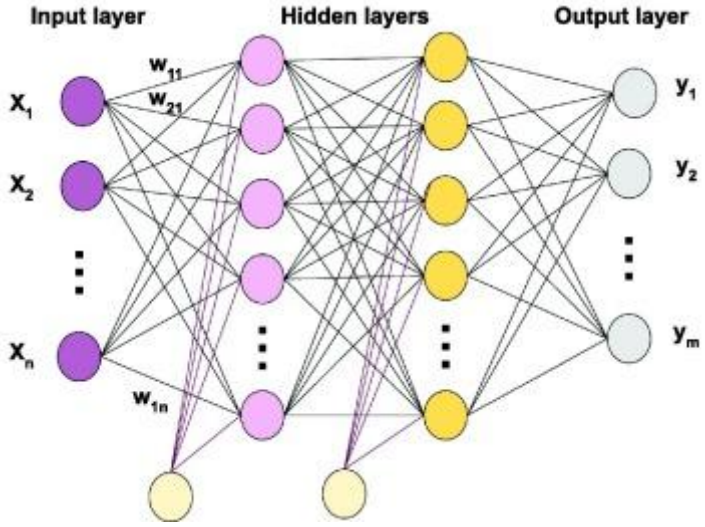

Figure 6: Multilayer Perceptron (MLP)

4.3.2 Dynamic Gesture Model

- Long Short-Term Memory (LSTM):
  1. Input: Sequence of feature vectors combining hand, pose, and face landmarks for each frame.
- Architecture:
  1. Input Layer: Sequence of vectors
  2. LSTM Layers: Two layers with 256 units each to capture temporal dependencies.
  3. Dense Layer: Fully connected layer with softmax activation for multi-class classification.
- Training Parameters:
- Optimizer: Adam.
- Loss Function: Sparse Categorical Cross-Entropy.
- Epochs: 2000
- Tools Used: TensorFlow/Keras.

4.4 Real-Time Gesture Recognition Pipeline

- Input:
  1. Static gestures are captured as single frames.
  2. Dynamic gestures are captured as video sequences.
- Feature Extraction:
  1. Hands: Mediapipe extracts 21 landmarks per hand.
  2. Pose: Upper body and arm movements are captured using pose landmarks.
  3. Face: Key facial landmarks provide additional context.
- Preprocessing:
  1. Landmarks are normalized relative to the bounding box dimensions.
  2. Sequence of features is created for dynamic gestures.
- Classification:
  1. Static gestures are classified using the KNN or MLP model.
  2. Dynamic gestures are classified using the LSTM model.
- Output:
  1. The predicted gesture is displayed as text or converted to speech.

4.5 Challenges Faced

- Real-Time Performance:
  1. Ensuring minimal latency during landmark extraction and inference.
- Data Imbalance:
  1. Certain gestures had limited representation in the dataset.
- Complex Backgrounds:
  1. Background noise occasionally interfered with landmark detection.

4.6 ` Validation and Testing

- Static Gesture Models (KNN and MLP):
  1. Validated with test accuracy of 95%
- Dynamic Gesture Model (LSTM):
  1. Evaluated on unseen video sequences with test accuracy of 84%.

The integration of Mediapipe for hand, pose, and face detection enhances feature extraction accuracy and provides additional contextual information, making the system robust and reliable for real-world applications.

Chapter 5: Results and Discussion

This chapter presents the results obtained from the implementation of the Sign Language Detection Model. The performance of both static and dynamic gesture recognition models is evaluated using various metrics, and the outcomes are analyzed in detail. Real-time testing results and observations are also included to demonstrate the model's effectiveness in practical scenarios.

5.1 Performance Metrics

- The models were evaluated using the following metrics:
  1. Accuracy: Percentage of correctly classified gestures.
  2. Precision: Measure of true positive predictions relative to the total predicted positives.
- Recall (Sensitivity): Measure of true positive predictions relative to the actual positives.
- F1-Score: Harmonic mean of precision and recall.

5.2 Static Gesture Recognition

5.2.1 K-Nearest Neighbors (KNN) Results

- The KNN model showed promising results for static gesture classification, particularly for well-separated gesture classes. The accuracy achieved was 100%.
- Results Summary:
  1. Test Accuracy: 100%
  2. Precision: 1.00
  3. Recall: 1.00
  4. F1-Score: 1.00

5.2.2 Multilayer Perceptron (MLP) Results

- The MLP model outperformed KNN in handling more complex gestures, achieving higher accuracy and better generalization.
- Results Summary:
  1. Test Accuracy: 95%
  2. Precision: 0.94
  3. Recall: 0.89
  4. F1-Score: 0.88.

5.3 Dynamic Gesture Recognition

5.3.1 LSTM Results

- The LSTM model excelled in recognizing sequential dynamic gestures by effectively capturing temporal dependencies. It achieved an accuracy of 84%.
- Results Summary:
  1. Test Accuracy: 84%
  2. Precision: 0.833
  3. Recall: 0.833
  4. F1-Score: 0.778

5.4 Real-Time Testing

5.4.1 Static Gesture Real-Time Results

- The static gesture recognition system performed well in real-time conditions, with minimal latency of [insert value] ms per prediction.
- Observations:
  1. High accuracy for distinct gestures such as "A", "B", and "C".
  2. Minor misclassifications occurred in gestures with similar shapes, such as "M" and "N".

Figure 7: Real-Time Static Gesture Testing Setup.

5.4.2 Dynamic Gesture Real-Time Results

- The dynamic gesture recognition system performed efficiently, processing sequences at [insert value] frames per second.
- **Observations:**
  1. The model successfully captured movement-based gestures, such as "Peace" and "Thank You".
  2. Slight delays in predictions for longer sequences were observed due to sequence formation overhead.

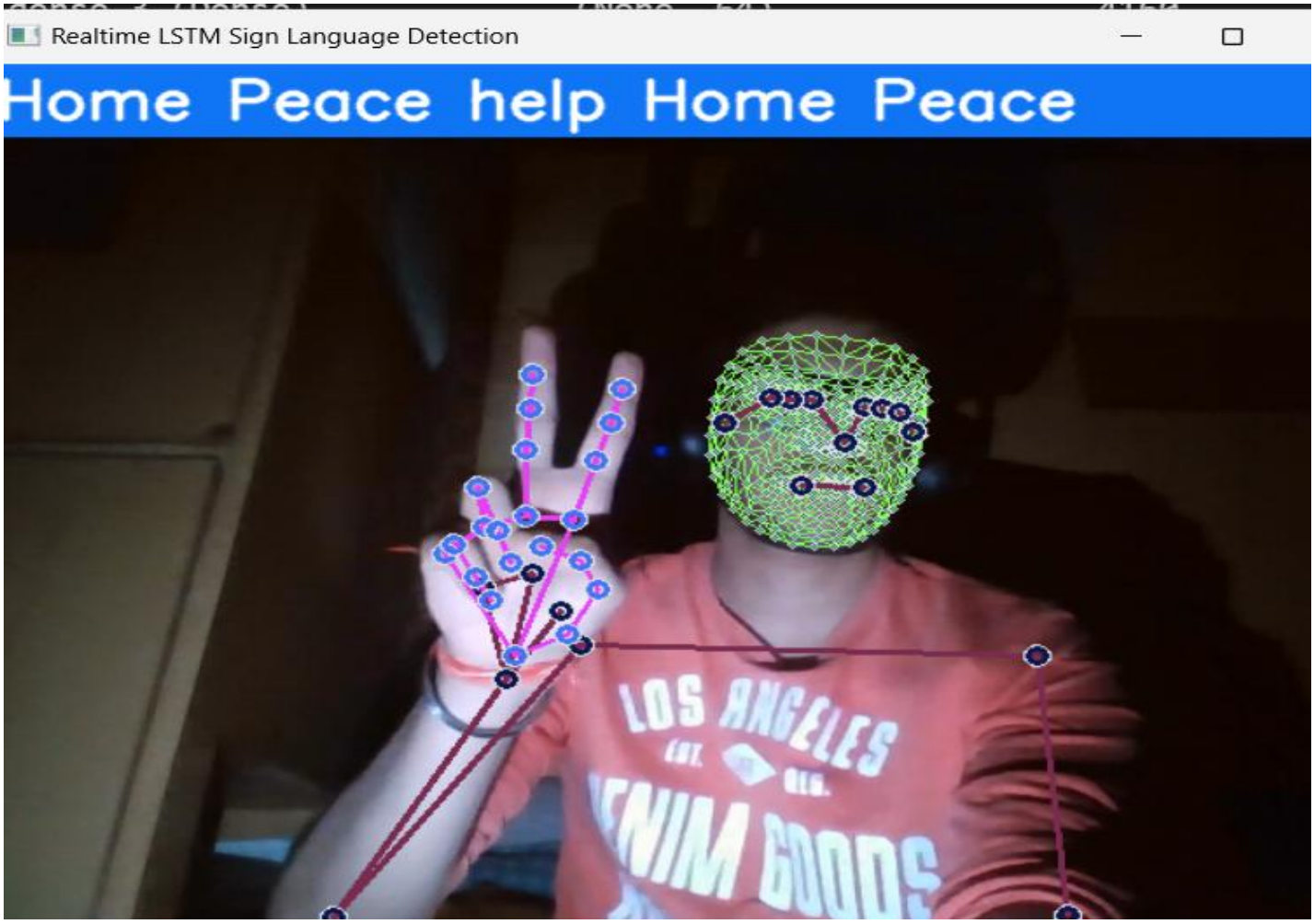

Figure 8: Real-Time Dynamic Gesture Testing Setup.

5.5 Comparative Analysis

- A comparative analysis of the models used for static and dynamic gesture recognition is summarized below.
-

Metric	KNN	MLP	LSTM
Accuracy	100%	95%	84%
Precision	1.00	0.94	0.833
Recall	1.00	0.89	0.833
F1-Score	1.00	0.88	0.778

Table 01: Bar Chart Comparing Metrics Across Models.

5.6 Discussion

- Strengths:
  1. The use of Mediapipe for feature extraction significantly improved accuracy and reduced computational costs.
  2. The modular approach enabled efficient handling of both static and dynamic gestures.
- Limitations:
  1. The model struggled with complex gestures in low-light or cluttered environments.
  2. Latency during dynamic gesture recognition needs further optimization.

- Key Insights:
  1. Combining hand, pose, and face landmarks added contextual understanding, improving the classification of ambiguous gestures.
  2. Sequence length plays a critical role in dynamic gesture recognition accuracy.

CHAPTER 6: CONCLUSION AND FUTURE WORK

6.1 Conclusion

The Sign Talk project successfully developed an intelligent system capable of translating sign language into text and speech in real time. This solution addressed the critical need for bridging communication gaps for individuals with hearing and speech impairments.

The system integrated three robust models:

Alphabet Recognition Model: Achieved an impressive accuracy of 98% using RNN and ANN architectures, effectively identifying individual sign language letters.

Static Gesture Recognition Model: Also attained 98% accuracy, accurately interpreting static gestures like "OK," "Not OK," and "Be Quiet."

Dynamic Gesture Recognition Model: Leveraged an LSTM-based approach to handle sequential data, achieving 84% accuracy in recognizing complex gestures such as "Peace" and "Home."

Key Achievements:

Sign-to-Text Conversion: Accurate recognition of both static and dynamic gestures, enabling seamless conversion of sign language into textual output.

Text-to-Speech Conversion: A unified system capable of synthesizing speech from recognized text, enhancing accessibility for non-signing audiences.

User Empowerment: By providing a user-friendly, real-time solution, Sign Talk empowers individuals with disabilities to communicate more effectively, fostering inclusivity and independence in their daily interactions.

The project demonstrates the potential of machine learning to create impactful assistive technologies, highlighting its role in addressing real-world challenges.

6.2 Future Work

While Sign Talk has achieved remarkable milestones, there remains substantial scope for enhancement and expansion in this domain. Future efforts could focus on incorporating multilingual support, enabling the system to recognize and translate sign languages from different regions such as American Sign Language (ASL) and British Sign Language (BSL), making the solution globally applicable. Additionally, increasing the dataset to include a broader vocabulary of gestures and contextual

sequences would significantly enhance the system's versatility. Improving the accuracy of dynamic gesture recognition through advanced architectures like Transformers could address edge cases and complex gestures. Real-time deployment on edge devices such as mobile phones and augmented reality platforms would ensure portability and accessibility for users in various settings. Furthermore, integrating gesture recognition with facial expressions and body language could provide a more comprehensive communication solution, while IoT integration could empower users to control smart devices through gestures. These advancements would significantly enhance the impact and reach of Sign Talk, driving inclusivity and innovation in assistive technology.

REFERENCES

1. S. K. Card, G. G. Robertson, and J. D. Mackinlay. The infor-mation visualizer, an information workspace. In ACM CHI,pages 181–186, 1991.

2. A. Graves. Supervised sequence labelling with recurrent neural networks. Springer, 2012. 2, 3, 4

3. F. Perronnin, J. Sanchez, and T. Mensink. Improving the fisher kernel for large-scale image classification. In ECCV, 2010. 2, 5

4. H. Wang, D. Oneata, J. Verbeek, and C. Schmid. A robust and efficient video representation for action recognition. International Journal of Computer Vision, 2015. 2, 5, 6

5. P. J. Werbos. Backpropagation through time: what it does and how to do it. Proceedings of the IEEE, 78(10):1550–1560, 1990. 3

6. P. Molchanov, S. Gupta, K. Kim, and K. Pulli. Multi-sensor system for driver's hand-gesture recognition. In IEEE Automatic Face and Gesture Recognition, 2015. 1, 2, 6

7. K. Kim, D. Lee, and I. Essa. Gaussian process regression flow for analysis of motion trajectories. In ICCV, 2011. 2

8. N. Neverova, C. Wolf, G. W. Taylor, and F. Nebout. Mod-drop: adaptive multi-modal gesture recognition. ArXiv preprint arXiv:1501.00102, 2014.

9. X. Yang and Y. Tian. Super normal vector for activity recognition using depth sequences. In CVPR, 2014

10. P. Molchanov, S. Gupta, K. Kim, and J. Kautz. Hand gesture recognition with 3d convolutional neural networks. In CVPRW, 2015

11. E. Ohn-Bar and M. Trivedi. Hand gesture recognition in realtime for automotive interfaces: A multimodal vision-based approach and evaluations. IEEE Trans. on Intelligent Transportation Systems, 15(6):1–10, 2014.

12. X. Shen, G. Hua, L. Williams, and Y. Wu. Dynamic hand gesture recognition: An examplar based approach from motion divergence fields. Image and Vision Computing, 2012.

13. A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar, and L. Fei-Fei. Large-scale video classification with convolutional neural networks. In CVPR, 2014.
14. D. Tran, L. D. Bourdev, R. Fergus, L. Torresani, and M. Paluri. Learning spatiotemporal features with 3d convolutional networks