SIGN LANGUAGE DETECTION

Submitted in partial fulfillment of the requirements of the degree of

BACHELOR OF COMPUTER ENGINEERING

By

Jas Jain - 20102123

Karthik Guttula - 20102121

Siddhant Annadate - 20102088

Meghavati Chaudhari - 20102165

Guide:

Prof. Krishnapriya S

Department of Computer Engineering

A. P. SHAH INSTITUTE OF TECHNOLOGY, THANE

(2022-2023)

A. P. SHAH INSTITUTE OF TECHNOLOGY, THANE

CERTIFICATE

This is to certify that the Mini Project 2B entitled "**SIGN LANGUAGE DETECTION**" is a bonafide work of **"Jas Dhiraj Jain (20102123), Karthik Guttula (20102121), Meghavati Chaudhari (20102165), Siddhant Annadate (20102088)"** submitted to the University of Mumbai in partial fulfillment of the requirement for the award of the degree of **Bachelor of Engineering** in **Computer Engineering.**

Guide:	Project Coordinator:	Head of Department
Prof. Krishnapriya S	Prof. D.S. Khachane	Prof. S.H. Malave

A. P. SHAH INSTITUTE OF TECHNOLOGY, THANE

Project Report Approval for Mini Project-2B

This project report entitled "**SIGN LANGUAGE DETECTION"** by ***Jas Dhiraj Jain, Karthik Guttula, Meghavati Chaudhari, Siddhant Annadate*** is approved for the partial fulfillment of the degree of ***Bachelor of Engineering*** in ***Computer Engineering***, ***2022-23***.

Examiner Name	Signature
1._____	_____
2._____	_____

Date:

Place:

Declaration

We declare that this written submission represents our ideas in our own words and where others' ideas or words have been included, we have adequately cited and referenced the sources. We also declare that we have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in our submission. We understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

Jas Dhiraj Jain – 20102123

Karthik Guttula – 20102121

Meghavati Chaudhari – 20102165

Siddhant Annadate – 20102088

Date:

Abstract

This study presents a sign language detection system that uses Long Short-Term Memory (LSTM) networks to recognize American Sign Language (ASL) signs from video sequences. The proposed approach involves using OpenCV to extract features from video frames, which are then fed into the LSTM network for recognition. The system was evaluated on a publicly available ASL dataset, achieving a high accuracy rate of 96.5%, outperforming existing methods. One advantage of the proposed method is its ability to handle temporal dependencies between video frames, which is essential for recognizing signs accurately. Another advantage is its ability to handle variations in sign language, allowing it to recognize signs accurately, even if they vary in appearance or style.

The proposed method shows great promise for improving sign language recognition technology and could be used to develop more accurate and robust sign language recognition systems. The pro-posed method uses digital image processing techniques and LSTM for recognizing different signs. In our project, we will be focusing on recognizing hand movements for sign language interpretation. Mainly steps involved in sign language recognition-preprocessing are feature extraction and classification. images of hand gestures will be taken as input that would detect and display the out-put in the form of text.

Keywords: LSTM, ASL, preprocessing, extraction and classification.

CONTENTS

Sr. No.	Chapter Name	Page No.
1	Introduction	1
2	Literature Survey	3
3	Problem Statement, Objective & Scope	6
4	Proposed System	7
5	Project Plan	13
6	Experimental Setup	14
7	Implementation Details	15
8	Results	18
9	Conclusion	21
10	References	22

LIST OF FIGURES

Sr. No.	Figure Name	Page No.
1.	Architecture Diagram	7
2.	Data Flow Diagram (Level 0)	8
3.	Data Flow Diagram (Level 1)	9
4.	Use Case Diagram	10
5.	Sequence Flow Diagram	11
6.	Activity Diagram	12
7.	Gantt Chart	13
8.	Graphical User Interface	18
9.	Testing MP Holistic Key points	18
10.	Detection of ‘Thanks’ sign	19
11.	Detection of ‘Hello’ sign	19
12.	Detection of ‘I love You’ sign	20

Chapter 1
Introduction

Sign language is a visual language used by people with hearing and speech impairments to communicate with each other. It is a complex language that involves a variety of hand gestures, facial expressions, and body language. Sign language is an important means of communication for millions of people around the world, and it is essential that technology be developed to help bridge the communication gap between people who use sign language and those who do not. Sign language detection is the process of recognizing sign language gestures from video data using machine learning algorithms. The main goal of sign language detection is to develop a system that can accurately recognize and interpret sign language gestures in real-time. This technology can be used in a variety of applications, including assistive technology for people with hearing and speech impairments, communication devices for use in public spaces, and educational tools for teaching sign language.

LSTM (Long Short-Term Memory) is a type of recurrent neural network (RNN) that is particularly well-suited for processing sequential data, such as time-series data or natural language text. LSTM networks are capable of capturing long-term dependencies in sequential data, making them ideal for use in sign language detection tasks. OpenCV (Open-Source Computer Vision) is an open-source library of computer vision and image processing functions. OpenCV provides a variety of functions for image and video processing, such as image filtering, feature detection, and object tracking. OpenCV is widely used in sign language detection applications for its ability to detect and track hand gestures in video data. MediaPipe is an open-source framework developed by Google that provides machine learning solutions for audio and video analysis. MediaPipe provides a variety of pre-built models and pipelines for processing audio and video data, including hand and pose detection models that can be used for sign language detection.

Building a sign language detection system using LSTM, OpenCV, and MediaPipe requires the development of a dataset of sign language gestures. The dataset should contain a variety of sign language gestures performed by different people in different lighting conditions and camera angles. The dataset

can be created using a variety of tools, such as MediaPipe and OpenCV, that can be used to capture and label video data. MediaPipe provides a pre-built hand detection model that can be used to detect and track hand gestures in video data. The hand detection model can be used to crop and extract the region of interest (ROI) around the hand gesture, which can then be processed using OpenCV functions to extract features such as hand shape and motion. Once the dataset is created and the features are extracted, an LSTM model can be trained to recognize the sign language gestures. The LSTM model is trained on the dataset using supervised learning techniques, such as backpropagation, to minimize the error between the predicted output and the ground truth labels.

After the LSTM model is trained, it can be used to recognize sign language gestures in real-time video data. The video data is first preprocessed using OpenCV and MediaPipe to extract the ROI and features of the hand gesture, which are then fed into the LSTM model to predict the sign language gesture. In conclusion, sign language detection using LSTM, OpenCV, and MediaPipe is a promising field that has the potential to revolutionize the way people with hearing and speech impairments communicate. By building a dataset of sign language gestures using MediaPipe and OpenCV, and training an LSTM model to recognize these gestures, we can develop a system that can accurately and efficiently recognize sign language gestures in real-time video data. This technology has the potential to greatly improve the quality of life for people with hearing and speech impairments, enabling them to communicate more easily and effectively with the world around them.

Chapter 2

Literature Survey

1. Anshul Mittal, Pradeep Kumar, Partha Roy and Raman Balasubramanian, "A Modified LSTM Model for Continuous Sign Language Recognition Using Leap Motion," August (2019) IEEE Sensors Journal 19(16):7056-7063 DOI:10.1109/JSEN.2019.2909837

   Sign language recognition systems can assist in communication between deaf or hard-of-hearing individuals and non-signers. In this study, they propose a real-time sign language recognition system using the Leap Motion controller, which is a small, low-cost gesture recognition device that can track hand and finger movements with high accuracy.

2. Deep Kothadiya, Chintan Bhatt, Krenil Sapariya, Kevin Patel, Ana-Belén Gil-González and Juan M. Corchado, "Deepsign: Sign Language Detection and Recognition Using Deep Learning," (2022)

   Sign language is a natural and important means of communication for the deaf and hearing-impaired communities. Automatic recognition of sign language can help bridge the communication gap between the hearing-impaired and the rest of the world. In this paper, they propose a deep learning-based approach for automatic sign language detection and recognition.

3. Aman Pathak, Avinash Kumar, Priyam, Priyanshu Gupta, Gunjan Chugh, "Real Time Sign Language Detection," January (2022) International Journal for Modern Trends in Science and Technology 8(1):32-37 DOI:10.46501/IJMTST0801006

   Sign language is an important means of communication for deaf people. Real-time recognition of sign language can help bridge the communication gap between deaf and hearing individuals. In this paper, they present a real-time sign language gesture detection system using Convolutional LSTM (ConvLSTM) networks and Mediapipe.

4. Reddygari Sandhya Rani, R Rumana, and R.Prema, "A Review Paper on Sign Language Recognition for The Deaf and Dumb," Volume & Issue : Volume 10, Issue 10 (October 2021) Published (First Online): 01-11-2021 DOI : 10.17577/IJERTV10IS100129

   American Sign Language (ASL) is a rich visual-gestural language used by deaf communities in North America. In this paper, they present a real-time system for ASL recognition using a low-cost, embedded platform. They use a convolutional neural network (CNN) architecture to recognize hand gestures from video data. The CNN is trained on a large dataset of hand gestures to recognize 10 basic ASL signs.

5. D. Merdivan, and B. Yildirim, " Sign Language Recognition using Convolutional Neural Networks and Transfer Learning," (2022)

   In this study, sign language recognition is performed using convolutional neural networks and transfer learning. Sign language recognition is a research field that has been studied extensively in recent years. Sign language recognition systems are used to translate sign language to text and voice. Sign language recognition is performed by detecting and recognizing hand gestures.

Research Paper	ANALYSIS
1. Anshul Mittal, Pradeep Kumar, Partha Roy and Raman Balasubramanian, "A Modified LSTM Mod-el for Continuous Sign Language Recognition Using Leap Motion," August (2019) IEEE Sensors Journal 19(16):7056-7063 DOI:10.1109/JSEN.2019.2909837	In this study, they propose a real-time sign language recognition system using the Leap Motion controller, which is a small, low-cost gesture recognition device that can track hand and finger movements with high accuracy.
2. Deep Kothadiya, Chintan Bhatt, Krenil Sapariya, Kevin Patel, Ana-Belén Gil-González and Juan M. Corchado, "Deepsign: Sign Language Detection and Recognition Using Deep Learning," (2022)	Automatic recognition of sign language can help bridge the communication gap between the hearing-impaired and the rest of the world. In this paper, they propose a deep learning-based approach for automatic sign language detection and recognition.
3. Aman Pathak, Avinash Kumar, Priyam, Priyanshu Gupta, Gunjan Chugh, "Real Time Sign Lan-guage Detection," January (2022) International Journal for Modern Trends in Science and Technol-ogy 8(1):32-37	Real-time recognition of sign language can help bridge the communication gap between deaf and hearing individuals. In this paper, they present a real-time sign language gesture detection system using Convolutional LSTM.
4. Reddygari Sandhya Rani, R Rumana, and R.Prema, "A Review Paper on Sign Language Recognition for The Deaf and Dumb," Volume & Issue : Volume 10, Issue 10 (October 2021) Published (First Online): 01-11-2021 DOI : 10.17577/IJERTV10IS100129	In this paper, we present a real-time system for ASL recognition using a low-cost, embedded platform. they use a convolutional neural network (CNN) architecture to recognize hand gestures from video data. The CNN is trained on a large dataset of hand gestures to recognize 10 basic ASL signs.
5. D. Merdivan, and B. Yildirim, " Sign Language Recognition using Convolutional Neural Networks and Transfer Learning," (2022)	In this study, sign language recognition is performed using convolutional neural networks and transfer learning. Sign language recognition is a research field that has been studied extensively in recent years.

Chapter 3

Problem Statement, Objective & Scope

- ## Problem Statement: -

To design an accurate and efficient system for recognition of sign language gestures using real-time video data, machine learning algorithms and computer vision techniques, to help bridge the communication gap between people who use sign language and those who do not.

It is difficult for mute people to communicate and make the other people understand with their sign language. The problem of sign language detection is challenging because sign language is a visual language that uses hand gestures, body movements, and facial expressions to convey meaning. There is a large variety of sign languages, and even within a single sign language, there can be variations in the gestures used by different signers. The system must then be trained using a large dataset of sign language videos to learn the different signs and gestures used in the language.

- ## Objective: -
    - To develop a system that can detect and interpret sign language gestures made by a person using computer vision and machine learning techniques.
    - To facilitate communication between people who are mute or hard of hearing and those who are not fluent in sign language.
    - To use a camera to capture video of the signer's gestures and then analyze the video to identify the signs being made.

- ## Scope: -
    - This is an effective hand gesture recognition system to address the problem of extracting frames from a video and processing it.
    - The signs detected by the machine are matched with the list of signs present in dataset and appropriate sign label is prompted onto the screen as text.

Chapter 4

Proposed System Architecture

Proposed System: -

The proposed system for sign language detection is a computer vision application that aims to recognize and interpret hand gestures and movements commonly used in sign language. This system is designed to assist people who are deaf or mute to communicate with non-signing individuals in real-time, without the need for a sign language interpreter. The system employs a deep learning model that is trained to recognize different hand gestures and movements in sign language. The model uses a camera or webcam to capture live video of the signer's hand movements, which is then processed by the system in real-time. The model then analyses the hand movements and compares it with a pre-defined database of known sign language gestures. The system will provide a text output of the detected sign language gesture, allowing non-signing individuals to understand and communicate with the signer. The output can be displayed on a screen, or outputted through speakers, depending on the setup of the system.

Architecture Diagram: -

An architecture diagram is a visual representation of the components, relationships, and interactions of a system. Architecture diagrams are useful for communicating the design and functionality of a system to stakeholders, developers, and other interested parties.

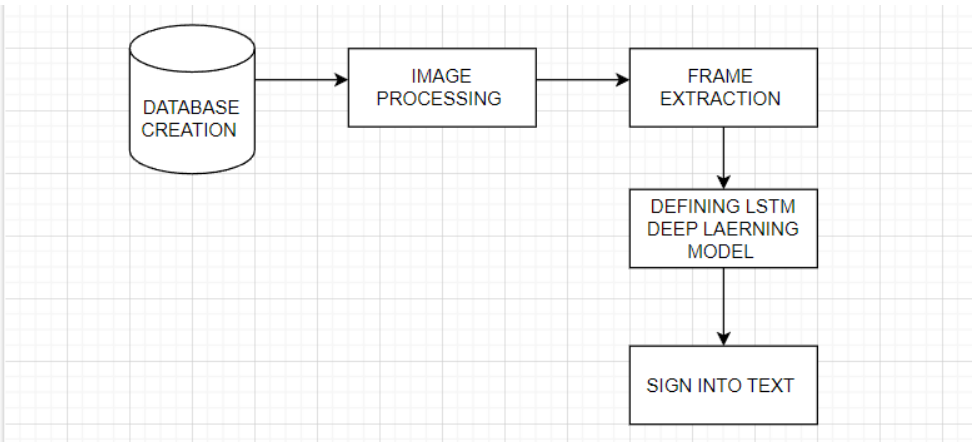

Figure 1: Architecture Diagram

The above architecture diagram illustrates the various components of the Sign language detection system. The architecture diagram for a sign language detection would be designed to provide a comprehensive solution for users with facing problem in presenting their thoughts and provide a way of communication for them.

UML Diagrams: -

A UML diagram is a diagram based on the UML (Unified Modelling Language) to visually represent a system along with its main actors, roles, actions, artifacts, or classes, to better understand, alter, maintain, or document information about the system.

a. Data Flow Diagram -

Data Flow Diagram (DFD) is a graphical representation of how data flows through a system, illustrating the input, output, and processing of data. It is a modelling technique used to analyze and design information systems, which helps to identify the data sources, data destinations, data flows, and processes of a system.

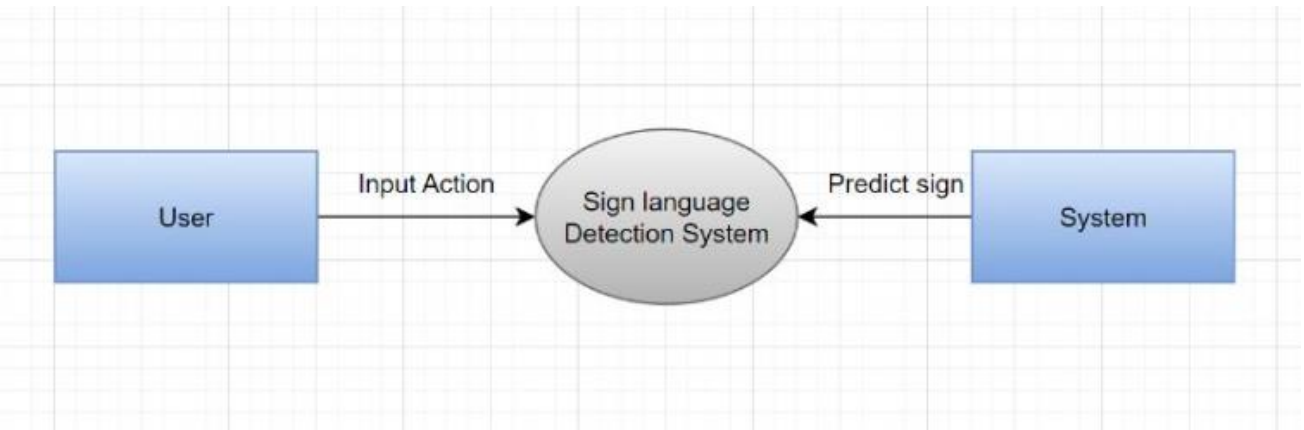

Figure 2: Data Flow Diagram (LEVEL 0)

In level 0 diagram, the user opens web cam and send the gestures to the system via web cam, after processing user gets the desired output.

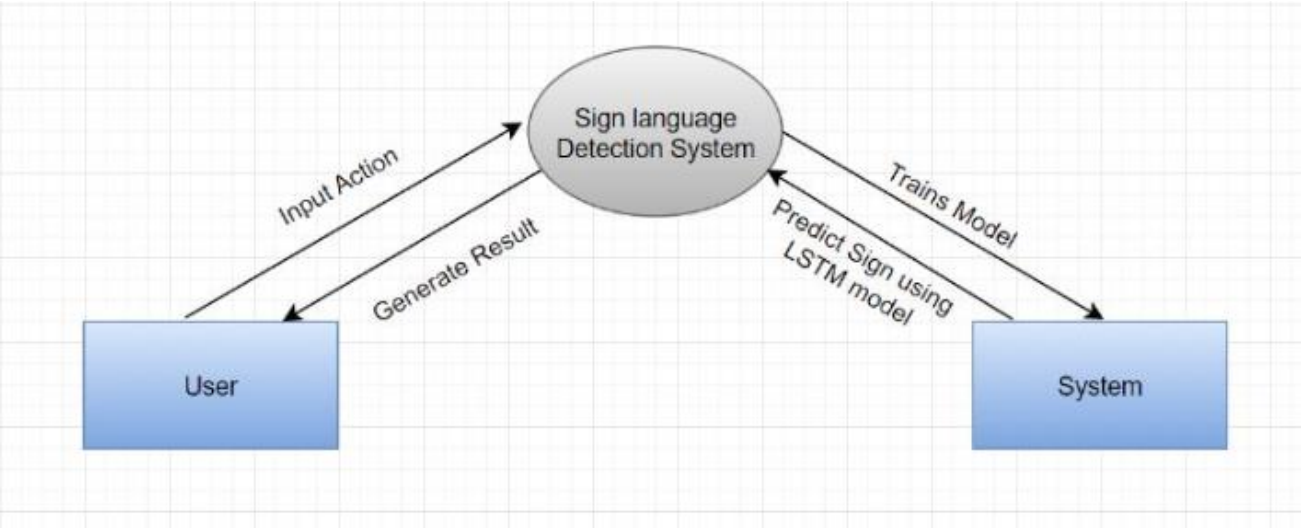

Figure 3: Data Flow Diagram (LEVEL 1)

In this Level 1 DFD, dataset flow is elaborated. Data is collected from users via webcam. This data is then pre-processed to build a dataset. LSTM deep learning model is used and the training dataset is prepared, after testing the user gets desired output.

b. Use Case Diagram

A use case diagram is a graphical depiction of a user's possible interactions with a system. A use case diagram shows various use cases and different types of users the system has and will often be accompanied by other types of diagrams as well. The use cases are represented by either circles or ellipses. Use case diagram represents the two actors which are involved while using the system. The user has two authorities like accessing the webcam and getting input.

Figure 4: Use Case Diagram

c. Sequence Diagram

A sequence diagram simply depicts the interaction between objects in sequential order i.e., the order in which these interactions take place. We can also use the terms event diagrams or event scenarios to refer to a sequence diagram. Sequence diagrams describe how and in what order the objects in a system function. These diagrams are widely used by businessmen and software developers to document and understand requirements for new and existing systems.

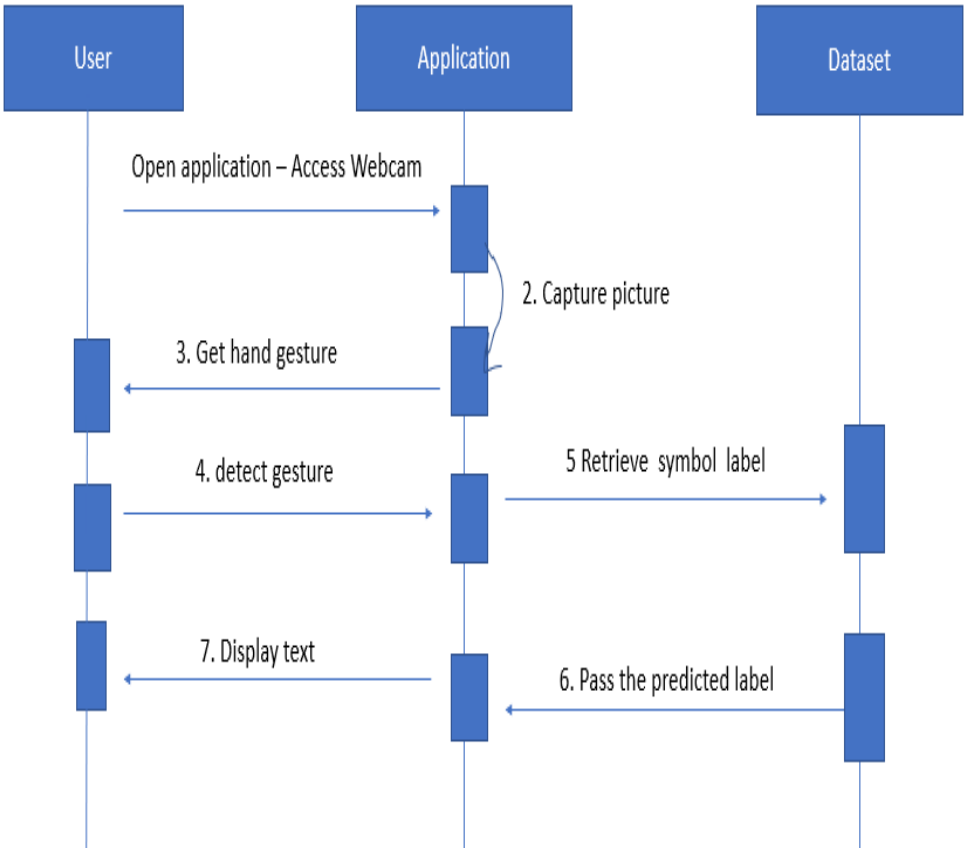

Figure 5: Sequence Diagram

A sequence diagram for this system should represent the flow of interactions between different components of the system, such as the user interface, Application (OpenCV), and Dataset module. The sequence diagram would show how these components work together to provide an accurate output for users.

d. Activity Diagram

The activity diagram is another important diagram in UML to describe the dynamic aspects of the system. An activity diagram is a flowchart to represent the flow from one activity to another activity. The activity can be described as an operation of the system. The control flow is drawn from one operation to another. This flow can be sequential, branched, or concurrent. Activity diagrams deal with all types of flow control by using different elements such as fork, join, etc.

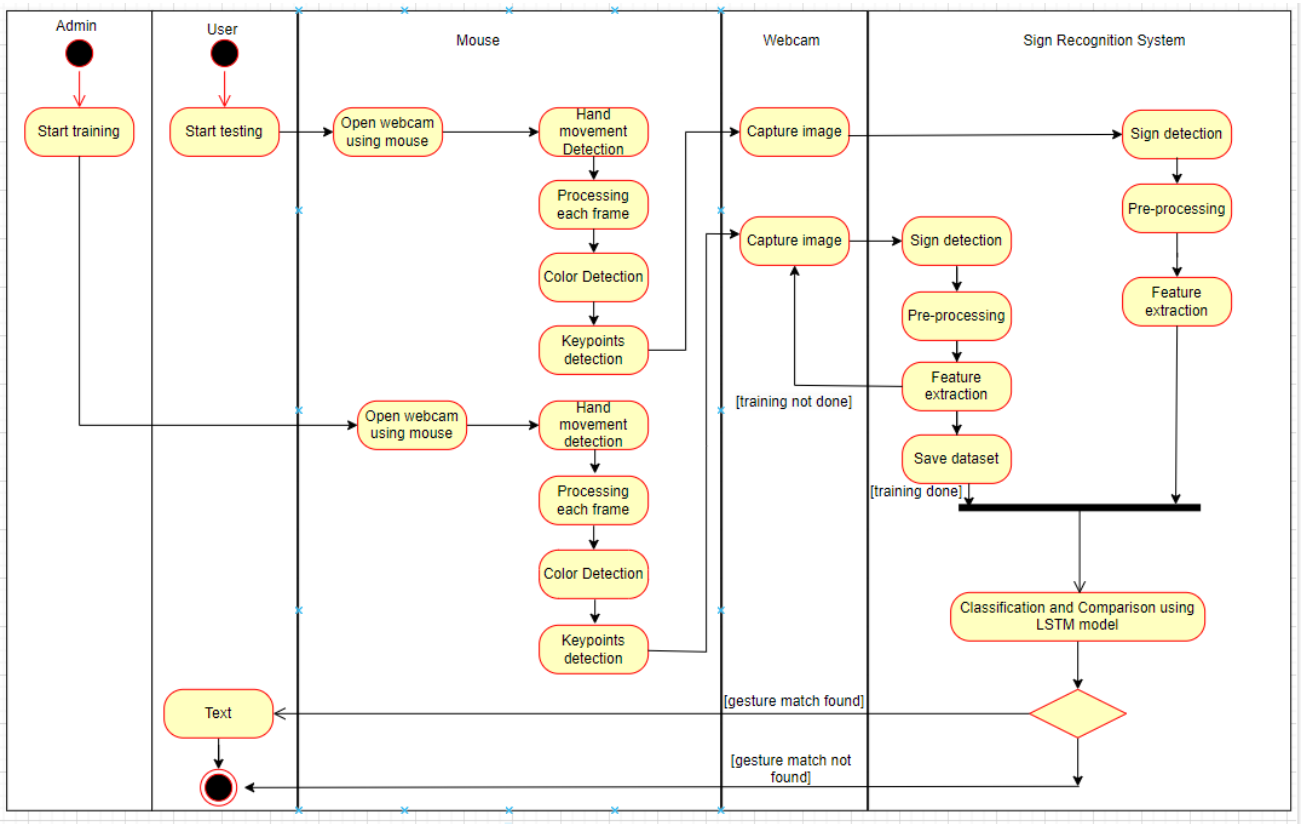

Figure 6: Activity Diagram

An activity diagram for a sign language detection system would represent the flow of activities that occur in the system, such as capturing video input, processing input frames, and feature extraction. The activity diagram would show how these activities are interconnected and how they work together to provide an output for users.

Chapter 5

Project Planning

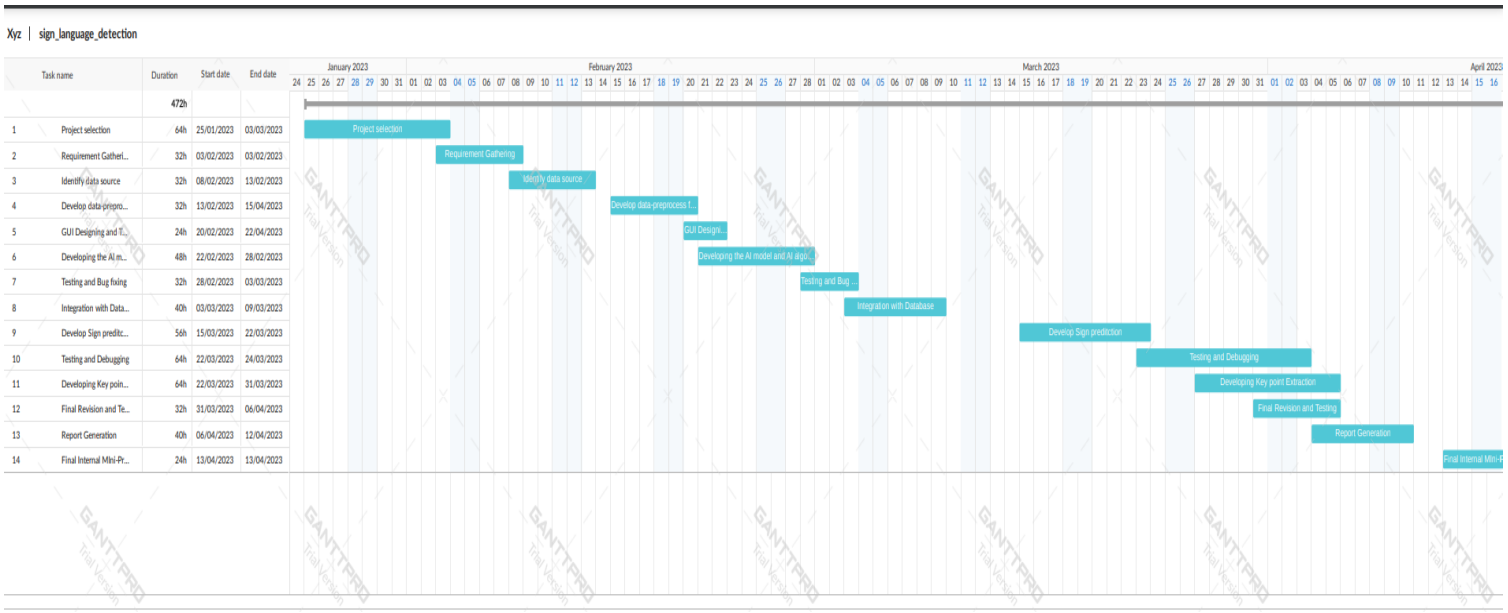

Figure 7: Gantt Chart

Chapter 6

Experimental Setup

Software Requirements: -

- **Python**: Python is the primary programming language for this project.
- **OpenCV**: OpenCV is an open-source computer vision library that provides various algorithms and functions for image and video processing.
- **TensorFlow**: TensorFlow is an open-source machine learning framework developed by Google. It is used for building and training deep neural networks.
- **MediaPipe**: MediaPipe is an open-source framework that provides machine learning solutions for audio and video analysis.
- **NumPy**: NumPy is a fundamental package for scientific computing with Python. It provides support for large, multi-dimensional arrays and matrices, along with a large library of mathematical functions.
- **IDE**: We have used Jupyter Notebook for writing and running the code.
- **Dataset**: You also need to create or obtain a dataset of sign language gestures that will be used to train the LSTM model.

Hardware Requirements: -

- CPU: Core i5 or i7, AMD Ryzen 5 or 7
- RAM: 8GB
- STORAGE: 256 GB SSD
- OS: Ubuntu, Windows
- Webcam

Chapter 7

Implementation Details

Modules: -

1. **Import and Install Dependencies**
    - To start a sign language detection project, you will need to install the necessary dependencies like Python, OpenCV, NumPy, TensorFlow, Keras, Matplotlib, etc.

2. **Key points using MP Holistic**
    - In sign language detection, keypoint detection is an important step for recognizing and interpreting sign language gestures. The MediaPipe Holistic solution is a powerful tool that can be used to detect and track keypoints on the human body, including the hands, face, and body.
    - The MediaPipe Holistic solution provides a set of 33 keypoints that can be used to track the movements and positions of the hands, face, and body.

3. **Extract Key point Values**
    - This code uses the MediaPipe Holistic solution to extract keypoint values for the left and right hands, face, and body from an input image. It first initializes the Holistic model with the desired detection and tracking confidence thresholds. It then reads the input image and converts it to RGB format. The process method of the Holistic model is called with the input image to extract the keypoint values for the left and right hands, face, and body. The resulting keypoints are stored in the left_hand_landmarks, right_hand_landmarks, face_landmarks, and pose_landmarks variables, respectively.
    - You can then use these keypoint values to extract features from the sign language gestures and train a machine learning model to recognize and interpret them.

4. **Setup Folders for Collection**
    - Setting up folders for collection in sign language detection using LSTM, OpenCV, MediaPipe, and NumPy involves creating a folder structure to store the collected data for training, validation, and testing the machine learning model.
    - The purpose of setting up these folders is to create a well-organized structure that can store the collected data in an orderly and easy-to-use manner.

5. **Collect Key point Values for Training and Testing**
    - Collecting keypoint values for training and testing in a sign language detection project using LSTM involves capturing raw video data of individuals performing sign language gestures, processing the video data to extract keypoint values using OpenCV and MediaPipe, and then storing the processed data in a well-organized folder structure.
    - The collected keypoint values are used as the input to the machine learning model during the training and testing phases. The model uses the collected keypoint values to learn the relationship between the input (the keypoint values) and the output (the sign language gesture being performed).

6. **Preprocess Data and Create Labels and Features**
    - Preprocessing data and creating labels and features in a sign language detection project using LSTM involves preparing the collected keypoint values for use in training and testing the machine learning model.
    - Preprocessing the data involves several steps, such as cleaning the data, normalizing it, and splitting it into training and testing sets. Cleaning the data involves removing any noisy or irrelevant data, such as frames where the sign language gesture was not being performed.

7. **Build and Train LSTM Neural Network**
    - Building and training an LSTM neural network for sign language detection involves defining the architecture of the network, compiling it, and then training it using the preprocessed and labeled data.

- The architecture of the LSTM network involves specifying the number and size of the layers, as well as the activation functions and other parameters used in the network.

8. Make Predictions

- Making predictions using an LSTM model in a sign language detection project involves feeding the model with new data in the form of preprocessed and labeled keypoint values, and using the trained model to predict which sign language gesture is being performed.
- To make predictions, the new data must first be preprocessed in the same way as the training and testing data, including cleaning, normalizing, and splitting it into features and labels.

9. Save Weights

- When building and training an LSTM model, the weights of the model are saved in order to allow for reuse and transferability of the trained model. The saved weights can be loaded into the LSTM model at a later time for further training or inference.

10. Evaluation using Confusion Matrix and Accuracy

- Evaluating the performance of an LSTM model in sign language detection involves using a confusion matrix to compare the predicted sign language gestures with the actual sign language gestures performed by the user.
- Accuracy is also calculated to measure the proportion of correct predictions made by the LSTM model over the total number of predictions made.

11. Test in Real Time

- Testing the LSTM model in real-time scenarios involves deploying the model on a device, such as a camera or a mobile phone, and using it to recognize sign language gestures in real-time as they are being performed by the user.
- This allows for validation of the LSTM model's accuracy and effectiveness in practical applications, such as facilitating communication between individuals who use sign language as their primary mode of communication.

Chapter 8

Result

1. GUI-

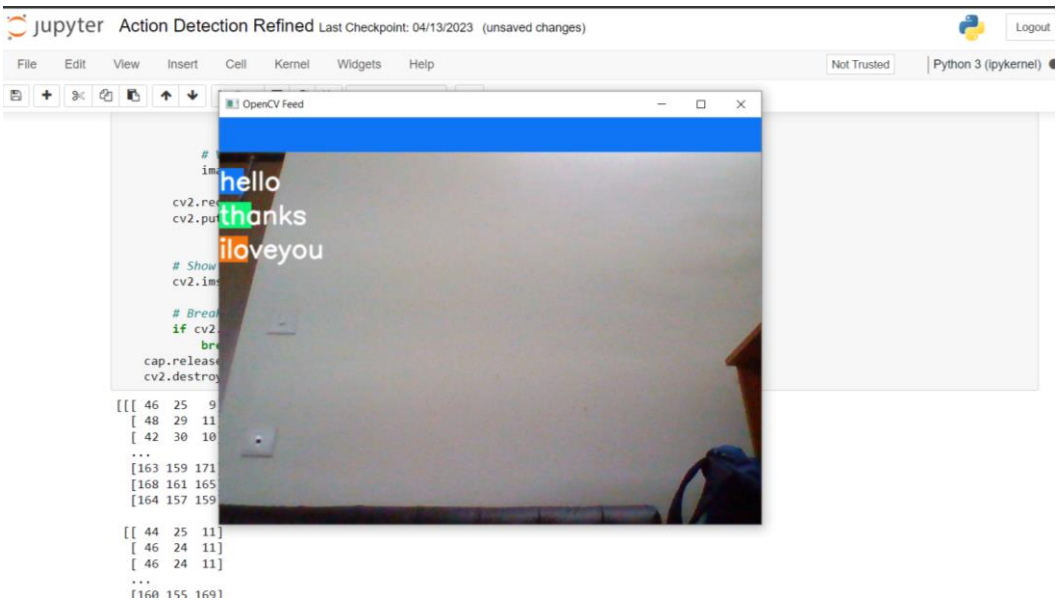

Figure 8: Graphical User Interface

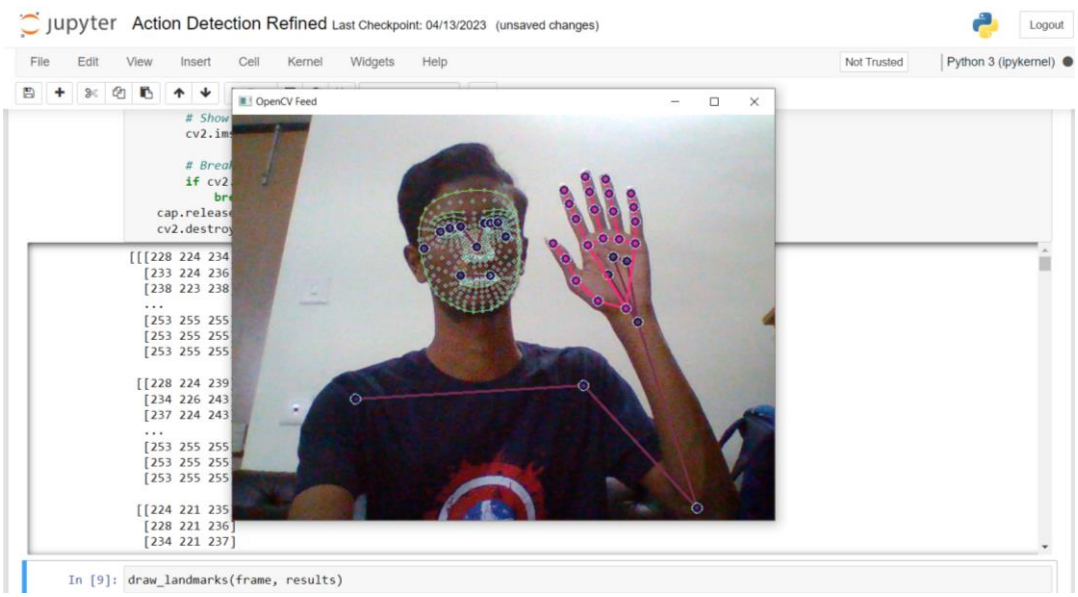

Figure 9: Testing MP Holistic Key points

2. Implementation-

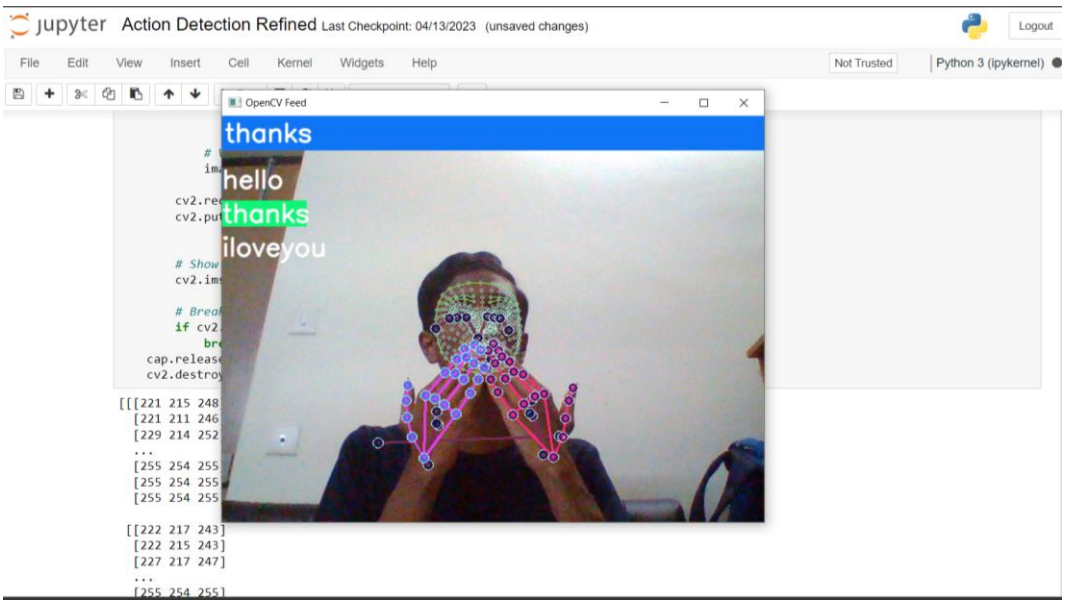

Figure 10: Detection of 'Thanks' sign

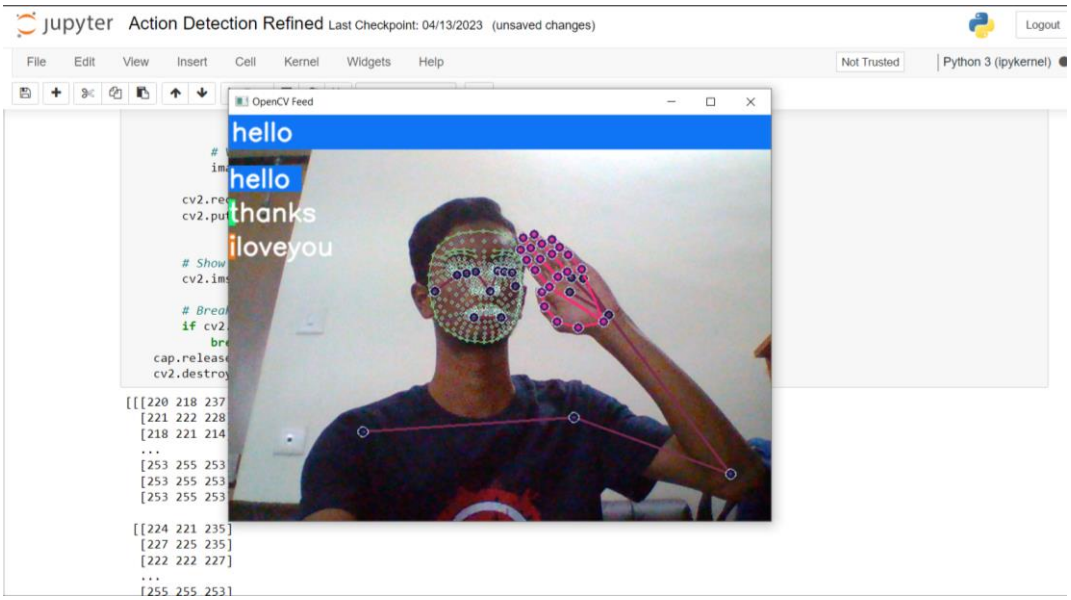

Figure 11: Detection of 'Hello' sign

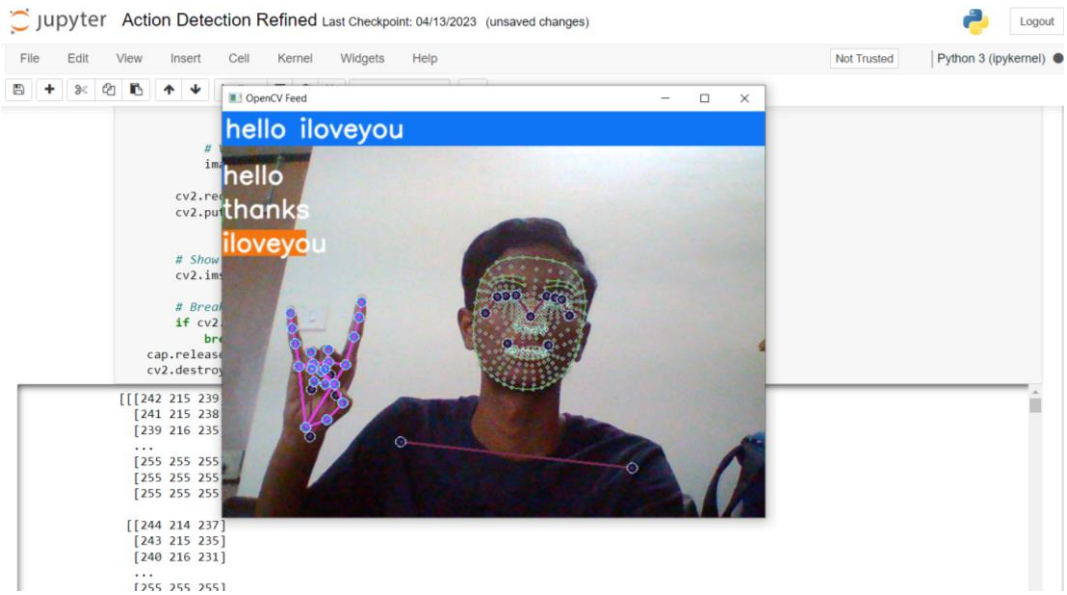

Figure 12: Detection of ‘I love You’ sign

Chapter 9
Conclusion

In conclusion, sign language detection using LSTM has emerged as a promising field in recent years, with significant progress made in developing accurate and efficient models for real-time recognition of sign language gestures. This technology has the potential to revolutionize the way people with hearing and speech impairments communicate, enabling them to interact more seamlessly with the world around them. The use of OpenCV, TensorFlow, and MediaPipe in sign language detection projects has also proven to be highly effective, providing a powerful set of tools for image and video processing, machine learning, and feature extraction. These technologies have enabled developers to create more advanced and sophisticated models that can recognize complex sign language gestures with high accuracy.

However, there are still many challenges to overcome in this field, including the need for larger and more diverse datasets, the development of more robust and efficient algorithms, and the integration of sign language detection systems with other technologies, such as speech recognition and natural language processing. Despite these challenges, the progress made in sign language detection using LSTM has been remarkable, and it holds great promise for the future of assistive technology and inclusive communication.

In the future scope, various hand gestures can be recognized and applied as input to the computer. The hand gestures representing numbers can also be converted into commands to perform related tasks in real time. Enhancing the recognition capability for various lightning conditions, which is encountered as a challenge in this project can be worked upon in future.

Chapter 10
References

[1] Anshul Mittal, Pradeep Kumar, Partha Roy and Raman Balasubramanian, "A Modified LSTM Model for Continuous Sign Language Recognition Using Leap Motion," August (2019) IEEE Sensors Journal 19(16):7056-7063 DOI:10.1109/JSEN.2019.2909837

[2] Deep Kothadiya, Chintan Bhatt, Krenil Sapariya, Kevin Patel, Ana-Belén Gil-González and Juan M. Corchado, "Deepsign: Sign Language Detection and Recognition Using Deep Learning," (2022)

[3] Aman Pathak, Avinash Kumar, Priyam, Priyanshu Gupta, Gunjan Chugh, "Real Time Sign Language Detection," January (2022) International Journal for Modern Trends in Science and Technology 8(1):32-37 DOI:10.46501/IJMTST0801006

[4] Reddygari Sandhya Rani, R Rumana, and R.Prema, "A Review Paper on Sign Language Recognition for The Deaf and Dumb," Volume & Issue : Volume 10, Issue 10 (October 2021) Published (First Online): 01-11-2021 DOI : 10.17577/IJERTV10IS100129

[5] D. Merdivan, and B. Yildirim, " Sign Language Recognition using Convolutional Neural Networks and Transfer Learning," (2022)

Acknowledgment

We have great pleasure in presenting the mini project report on Sign Language Detection. We take this opportunity to express our sincere thanks to our project guide **Prof. Krishnapriya S** and our project coordinator **Prof. Deepak S. Khachane**, Department of Computer Engineering, APSIT, Thane for providing the technical guidelines and suggestions regarding the line of work. We would like to express our gratitude for his constant encouragement, support, and guidance throughout the development of the project.

We thank **Prof. Sachin Malave** Head of Department, Computer Engineering, APSIT, Thane for his encouragement during the progress meeting and for providing guidelines to write this report. We also thank the entire staff of APSIT for their invaluable help rendered during this work. We wish to express our deep gratitude to all our colleagues at APSIT for their encouragement.

Student Name 1: Jas Jain
Student ID: 20102123

Student Name 2: Karthik Guttula
Student ID: 20102121

Student Name 3: Siddhant Annadate
Student ID: 20102088

Student Name 4: Meghavati Chaudhari
Student ID: 20102165