SIGN LANGUAGE TO TEXT LANGUAGE

Application Development-I Report Submitted
In partial fulfilment of the requirement for the award of the degree of

Bachelor of Technology
In
Computer Science and Engineering -Artificial Intelligence and Machine Learning

by

P.HUSSAINSETTY - 22N31A66D9

T.VIJESHWARREDDY - 22N31A66H6

Y.RAJASEKHAR REDDY - 22N31A66J6

Under the Guidance of
Ms.B.Jyothi
Assistant Professor

Computational Intelligence Department
MRCET

DEPARTMENT OF COMPUTATIONAL INTELLIGENCE

MALLA REDDY COLLEGE OF ENGINEERING AND TECHNOLOGY
(Affiliated to JNTU, Hyderabad)

ACCREDITED by AICTE-NBA
Maisammaguda, Dhulapally post, Secunderabad-500014.

2024-2025

DECLARATION

I hereby declare that the project entitled **"SIGN LANGUAGE TO TEXT LANGUAGE"** submitted to **Malla Reddy College of Engineering and Technology,** affiliated t**o** Jawaharlal Nehru Technological University Hyderabad (JNTUH) for the award of the degree of **Bachelor of Technology** in **Computer Science and Engineering- Artificial Intelligence and Machine Learning** is a result of original research work done by me.

It is further declared that the project report or any part there of has not been previously submitted to any University or Institute for the award of degree or diploma.

P.Hussain setty (22N31A66D9)
T.Vijeshwar reddy (22N31A66H2)
Y.Rajasekhar reddy (22N31A66J6)

MALLA REDDY COLLEGE
OF ENGINEERING & TECHNOLOGY
(AUTONOMOUS INSTITUTION - UGC, GOVT. OF INDIA)
Affiliated to JNTUH; Approved by AICTE, NBA-Tier 1 & NAAC with A-GRADE | ISO 9001:2015

CERTIFICATE

This is to certify that this is the bonafide record of the project titled **"Handwritten Numerical Character Recognition Using Deep Learning"** submitted by **P.Hussain Setty(22N31A66D9), T.Vijeshwar reddy(22N31A66H6), Y.Rajasekhar reddy (22N31A66J6)** of B.Tech in the partial fulfillment of the requirements for the degree of **Bachelor of Technology** in **Computer Science and Engineering- Artificial Intelligence and Machine Learning**, Dept. of CI during the year 2024-2025. The results embodied in this project report have not been submitted to any other university or institute for the award of any degree or diploma.

Ms.B.Jyothi
Assistant Professor
INTERNAL GUIDE

Dr. D. Sujatha M.Tech, PhD
Professor
HEAD OF THE DEPARTMENT

EXTERNAL EXAMINER

ACKNOWLEDGEMENT

We feel honored and privileged to place our warm salutation to our college Malla Reddy College of Engineering and technology (UGC-Autonomous), our Director ***Dr. VSK Reddy*** who gave us the opportunity to have experience in engineering and profound technical knowledge.

We are indebted to our Principal ***Dr. S. Srinivasa Rao*** for providing us with facilities to do our project and his constant encouragement and moral support which motivated us to move forward with the project.

We would like to express our gratitude to our Head of the Department ***Dr. D. Sujatha,*** *Professor* for encouraging us in every aspect of our system development and helping us realize our full potential.

We would like to thank our Application Development-I coordinator **Dr. P. Hari Krishna**, Associate Professor as well as our internal guide **Ms.B.Jyothi,** Assistant Professor, for their structured guidance and never-ending encouragement. We are extremely grateful for valuable suggestions and unflinching co-operation throughout Application Development-I work.

We would also like to thank all supporting staff of department of CI and all other departments who have been helpful directly or indirectly in making our Application Development-I a success.

We would like to thank our parents and friends who have helped us with their valuable suggestions and support has been very helpful in various phases of the completion of the Application Development-I.

By

P.Hussain Setty (22N31A66F9)
T.Vijeshwar Reddy (22N31A66G2)
Y.Rajasekhar Reddy (22N31A66J8)

ABSTRACT

Sign language is a vital means of communication for the Deaf and hard of hearing communities. However, the language barrier between sign language users and those who do not understand it often poses significant communication challenges. This project presents a novel system for sign language-to-text conversion. The proposed system utilizes advanced computer vision techniques to accurately recognize and translate sign language gestures into written text. A convolutional neural network (CNN) is employed for hand gesture detection and classification, while a recurrent neural network (RNN) processes the sequence of gestures to generate coherent text.

The system is trained on a comprehensive dataset encompassing various sign language gestures, ensuring high accuracy and robustness. Experimental results demonstrate the efficacy of the approach, with the system achieving a high recognition rate and efficient performance. This technology holds promise for enhancing accessibility and inclusivity, enabling more effective communication for sign language users in diverse settings.

TABLE OF CONTENTS

CHAPTER 1

INTRODUCTION

1.1Purpose:

Real-time sign language translation offers significant benefits in improving communication and accessibility. It helps individuals who are deaf or hard-of-hearing communicate with those who may not know sign language by instantly converting signs into readable text. This facilitates smoother, more effective conversations, making it easier for people to interact in various settings. Additionally, such technology enhances accessibility by making services, education, and daily life more inclusive for those who use sign language. It allows them to engage more seamlessly in different environments, whether at work, school, or in public spaces. Moreover, real-time translation bridges language barriers between different sign languages, such as American Sign Language (ASL) and British Sign Language (BSL), by converting signs into a common written form. This helps individuals from diverse linguistic backgrounds communicate with one another, fostering greater understanding and inclusivity.

1.2 Motivation of project:

Deaf or hard-of-hearing individuals often encounter significant communication challenges when interacting with people who do not know sign language, leading to barriers in education, healthcare, customer service, and social interactions. While sign language is a fully developed language with its own grammar and syntax, it is not universally understood. This creates a gap that can be bridged by a conversion system that transcribes gestures and signs into text, enabling clearer communication. Recent advancements in technology, particularly in machine learning, computer vision, and AI-based models, have made this possible. These technologies can now detect and interpret the key elements of sign language, such as hand shapes, movements, facial expressions, and body language, allowing for accurate real-time sign-to-text translation. The growing demand for inclusive technologies has further emphasized the importance of accessibility, driving the development of tools that promote inclusivity. This shift is part of a broader effort to ensure that people with disabilities can fully participate in education, government services, and public life, fostering a more inclusive society where individuals can communicate freely, regardless of their language or disability.

1.3 Scope of project:

The scope of the Sign Language to Text project is focused on developing a system that converts sign language gestures into written text in real time, aiming to improve communication for deaf and hard-of-hearing individuals. The project leverages advanced technologies like computer vision and machine learning to accurately recognize hand shapes, facial expressions, and body movements, which are crucial elements of sign language. It seeks to support multiple sign languages, such as American Sign Language (ASL) and British Sign Language (BSL), to ensure inclusivity for diverse users worldwide.

The system will be designed for various applications, including educational settings,workplaces, and public services, allowing sign language users to communicate more effectively with people who don't know sign language. The project also aims to be integrated into smart devices, enabling users to access real-time translation anytime, anywhere. While addressing challenges such as ensuring high recognition accuracy and real-time processing, the project will also focus on creating a user-friendly interface and ensuring data privacy. In the long term, the project may expand to include machine learning improvements, integration with other assistive technologies, and broader support for regional variations in sign language, ultimately fostering a more inclusive society** where communication barriers are minimized.

1.4 Project Features:

The system features are as follows:

1. **Real-Time Translation**: The system will convert sign language gestures into text instantly, allowing seamless communication between sign language users and non-users.

2. **Multi-Language Support**: It will support multiple sign languages (e.g., ASL, BSL, ISL) to cater to a wide range of users from different regions and backgrounds.

3. **High Accuracy and Recognition**: The technology will use advanced computer vision and machine learning to accurately recognize and translate gestures, ensuring reliable text output.

4. **User-Friendly Interface**: The system will be easy to use, with an intuitive design that makes it accessible to both deaf users and those unfamiliar with sign language.

CHAPTER 2

SYSTEM REQUIREMENTS

2.1 Hardware Requirements:

- RAM 4 GB
- Processor Intel(R)core (TM)i3 or more.
- Keyboard
- Mouse
- HDD 512 GB

2.2 Software requirements:

Frontend:

JavaScript
WebRTC

Backend:

- TensorFlow
- Python
- Open CV
- Flask

Operating System:

- Windows 11

2.3 Existing System:

DeepLearningComputerVision: A model that uses Deep Neural Network architectures to recognize hand gestures and generate corresponding English text.

2.3.1 Drawbacks of existing system:

- **Facial expressions and lip movements:** Many systems have difficulties in recording hand and finger movements and an inability to capture facial expressions, lip movements, and eye movements.
- **Accessibility:** Many systems rely on specialized hardware or apps, which may not be accessible to all users.

2.4 Proposed System:

Input Method:

- Video stream (camera or wearable devices like gloves).

Preprocessing:

- Hand detection and tracking (e.g., using MediaPipe, OpenPose).
- Segmentation of hands and body.
- Hand landmark detection (key points, joints).

Gesture Recognition:

- Static signs: CNNs for handshape recognition.
- Dynamic signs: RNNs, LSTMs, or Transformers for movement recognition.
- Contextual understanding for sentence structure.

Post-Processing:

- Word and sentence formation.
- Disambiguation using NLP techniques.

Text Output:

- Display text in real-time or save as a text file.
- Feedback to the user for accuracy.

Machine Learning:

- Train on large annotated sign language datasets (e.g., RWTH-PHOENIX, ASL datasets).
- Data augmentation to improve model robustness.

CHAPTER 3

SYSTEM DESIGN

3.1 System Architecture

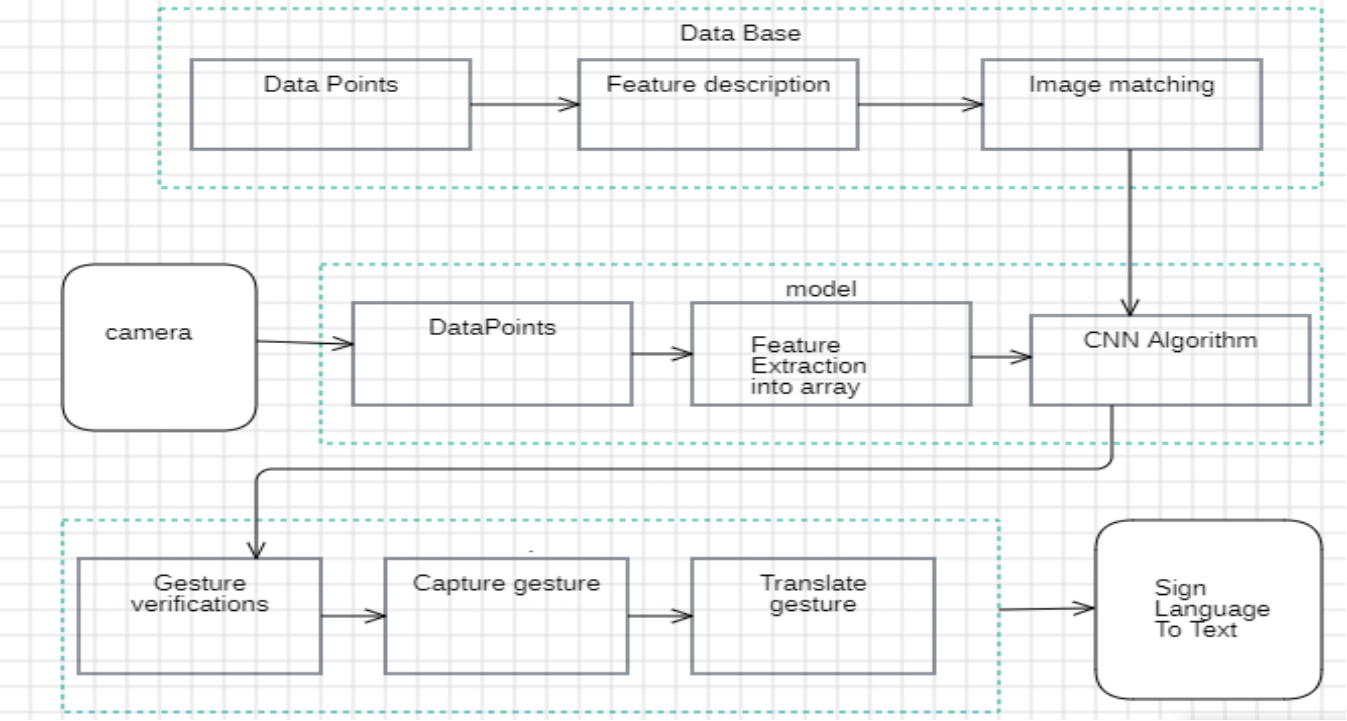


Fig 3.1 System Architecture

3.2 UML Diagrams

3.2.1 Use case diagram

Use Case during requirement elicitation and analysis to represent the functionality of the system. Use case describes a function by the system that yields a visible result for an actor. The identification of actors and use cases result in the definitions of the boundary of the system i.e., differentiating the tasks accomplished by the system and the tasks accomplished by its environment.

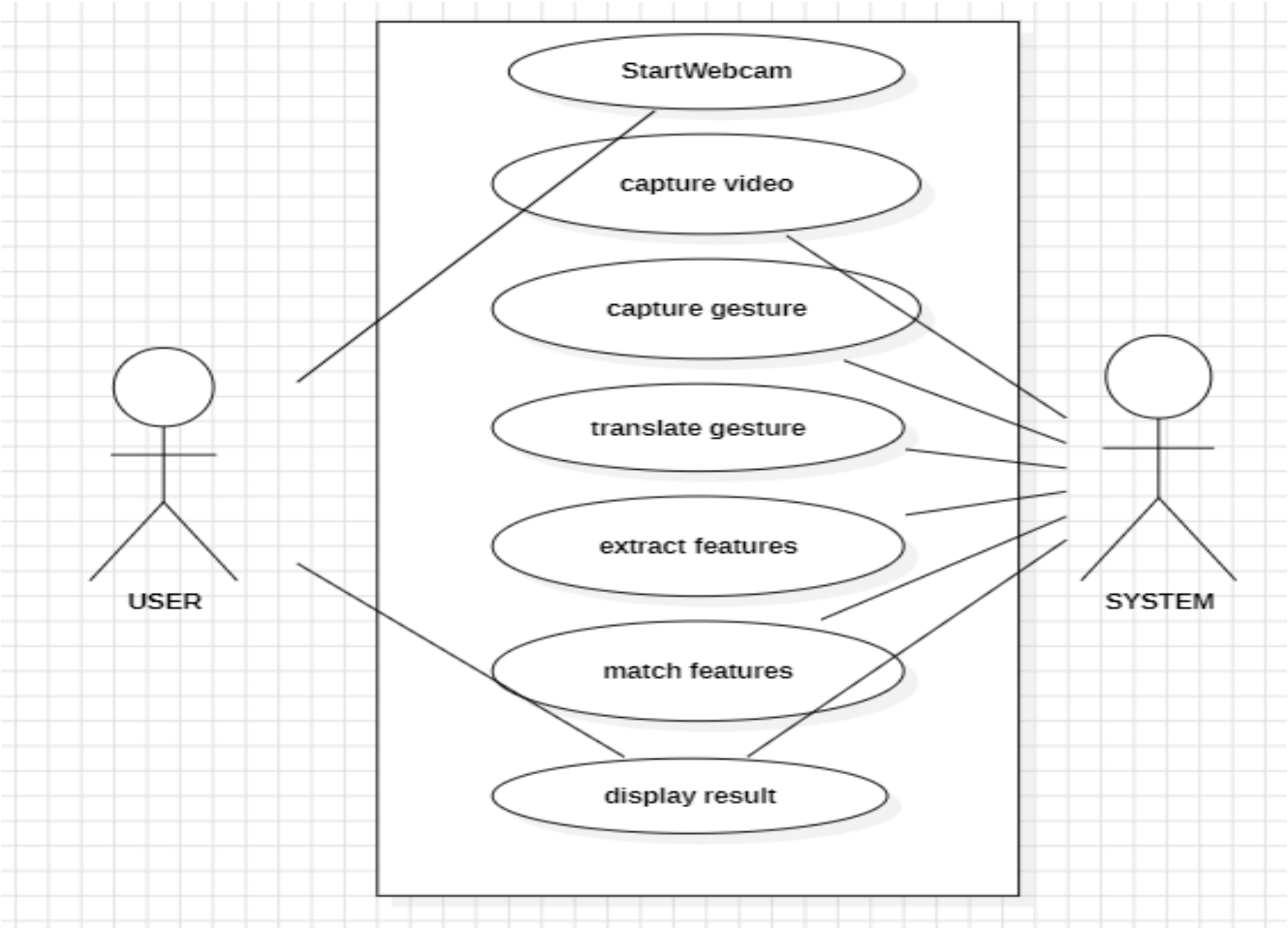


Fig 3.2 Use Case Diagram

3.2.2 Class Diagram

Class diagrams model class structure and contents using design elements such as classes, packages and objects. Class diagram describe the different perspective when designing a system-conceptual, specification and implementation. Classes are composed of three things: name, attributes, and

operations. Class diagram also display relationships such as containment, inheritance, association etc. The association relationship is most common relationship in a class diagram. The association shows the relationship between instances of classes.

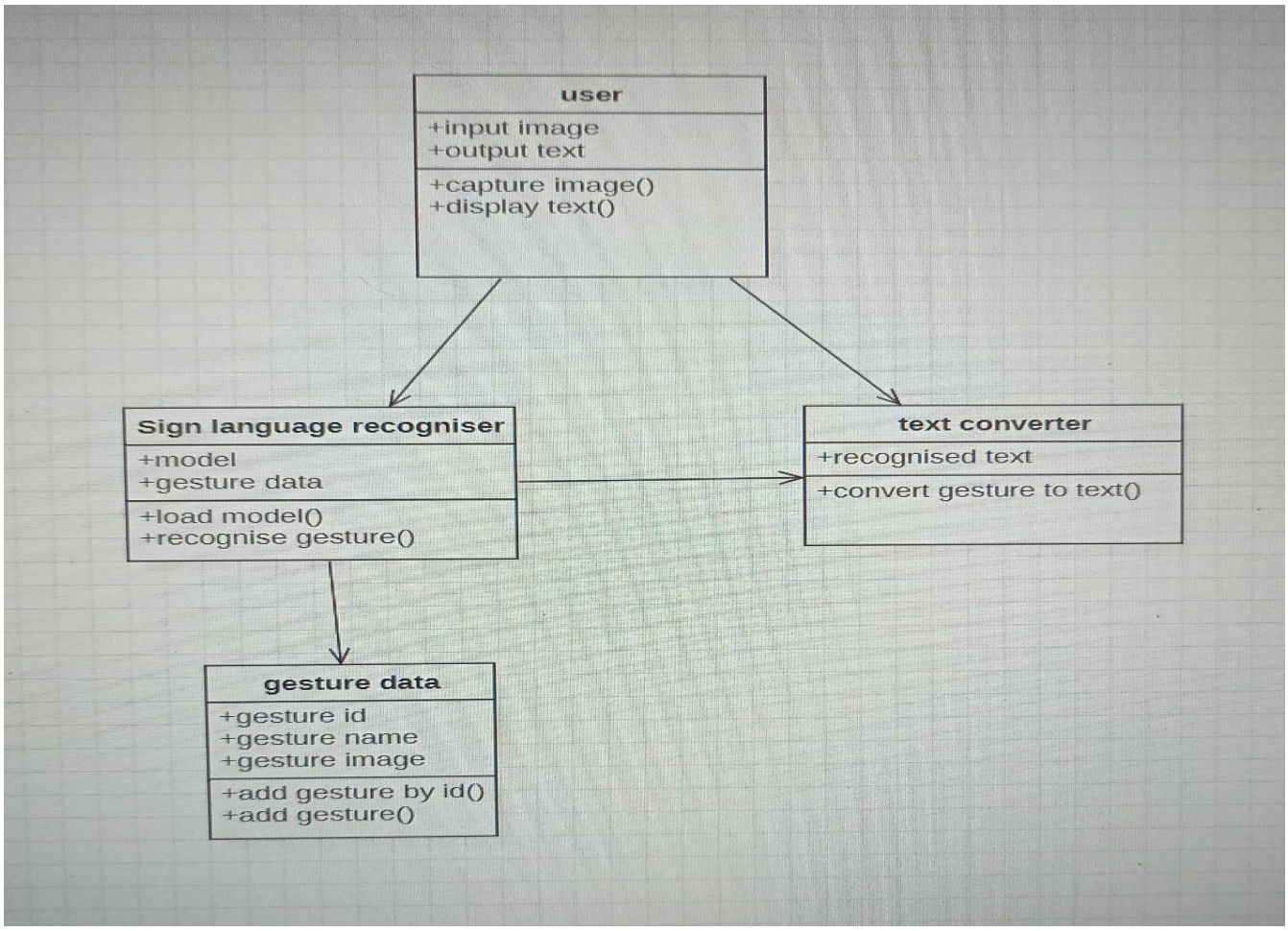


Fig 3.3 Class Diagram

3.2.3 Sequence Diagram

Sequence diagram displays the time sequence of the objects participating in the interaction. This consists of the vertical dimension (time) and horizontal dimension (different objects).

Objects: An object can be thought of as an entity that exists at a specified time and has a definite value, as well as a holder of identity. A sequence diagram depicts item interactions in chronological order. It illustrates the scenario's objects and classes, as well as the sequence of messages sent between them in order to carry out the scenario's functionality. In the Logical View of the system under development, sequence diagrams are often related with use case realizations. Event diagrams and event scenarios are other names for sequence diagrams. A sequence diagram depicts multiple processes or things that exist simultaneously as parallel vertical lines (lifelines), and the messages passed between them as horizontal arrows, in the order in which they occur. This enables for the graphical specification of simple runtime scenarios.

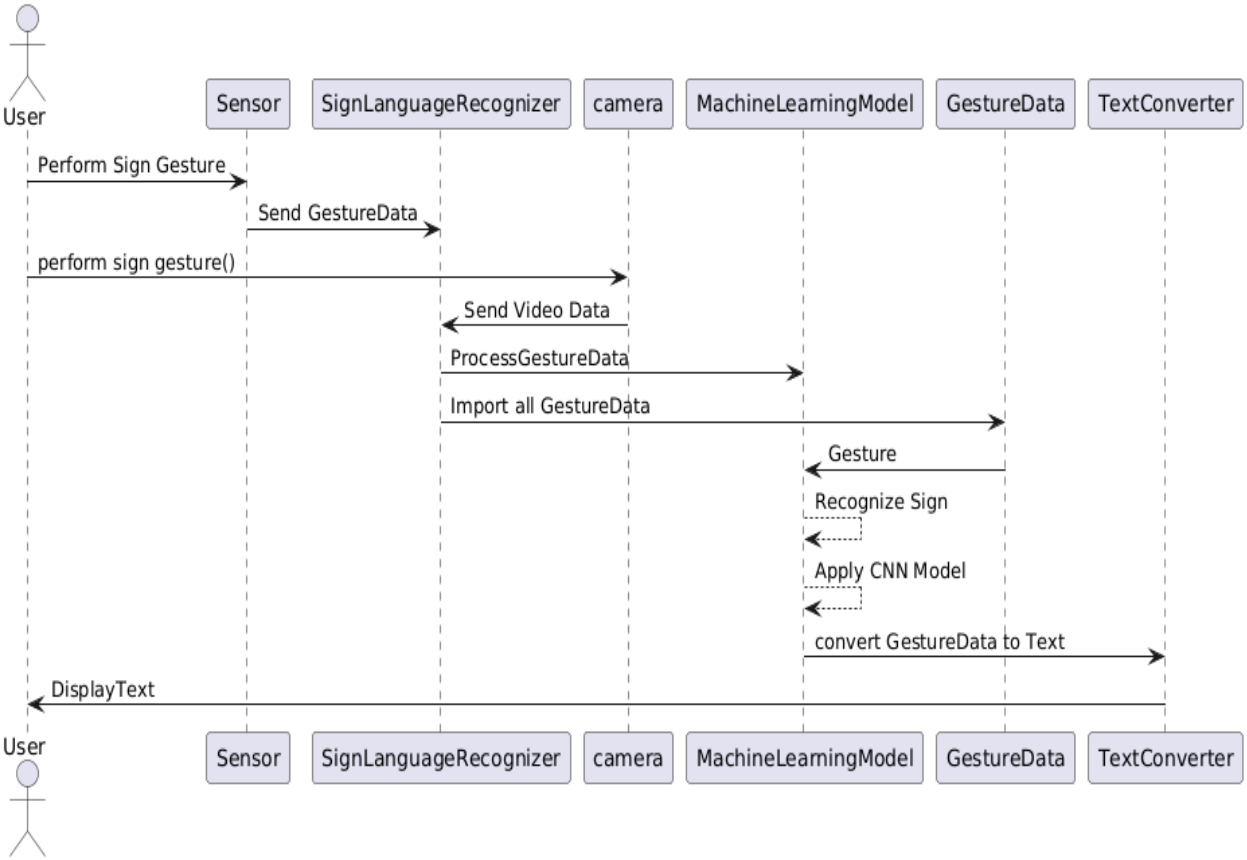


Fig 3.4 Sequence Diagram

3.2.4 Activity Diagram

The process flows in the system are captured in the activity diagram. Similar to a state diagram, an activity diagram also consists of activities, actions, transitions, initial and final states, and guard conditions

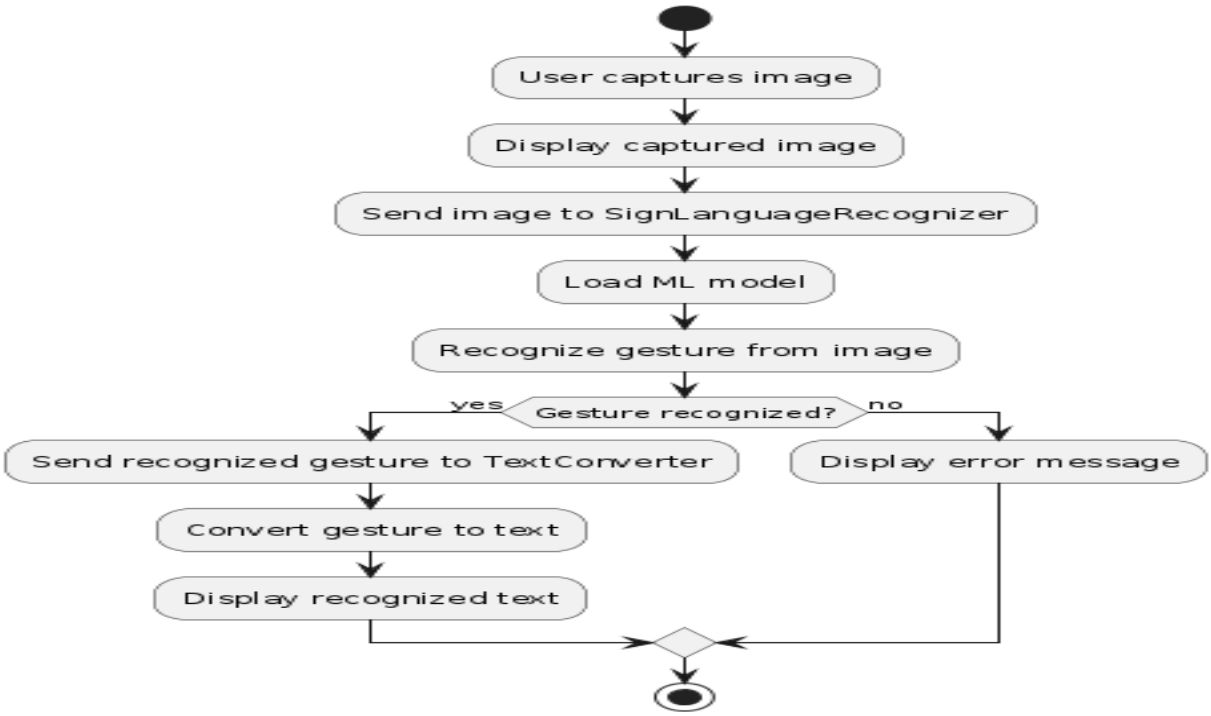


Fig 3.5 Activity Diagram

CHAPTER 4

IMPLEMENTATION

4.1: Code

4.1.1: DataCollection

```
import cv2
from cvzone.HandTrackingModule import HandDetector
import numpy as np
import math
import time

cap = cv2.VideoCapture(0)
detector = HandDetector(maxHands=1)
offset = 20
imgSize = 300
counter = 0

folder = "Data/Okay"

while True:
    success, img = cap.read()
    hands, img = detector.findHands(img)
    if hands:
        hand = hands[0]
        x, y, w, h = hand['bbox']

        imgWhite = np.ones((imgSize, imgSize, 3), np.uint8)*255

        imgCrop = img[y-offset:y + h + offset, x-offset:x + w + offset]
        imgCropShape = imgCrop.shape

        aspectRatio = h / w

        if aspectRatio > 1:
            k = imgSize / h
            wCal = math.ceil(k * w)
            imgResize = cv2.resize(imgCrop, (wCal, imgSize))
            imgResizeShape = imgResize.shape
            wGap = math.ceil((imgSize-wCal)/2)
```

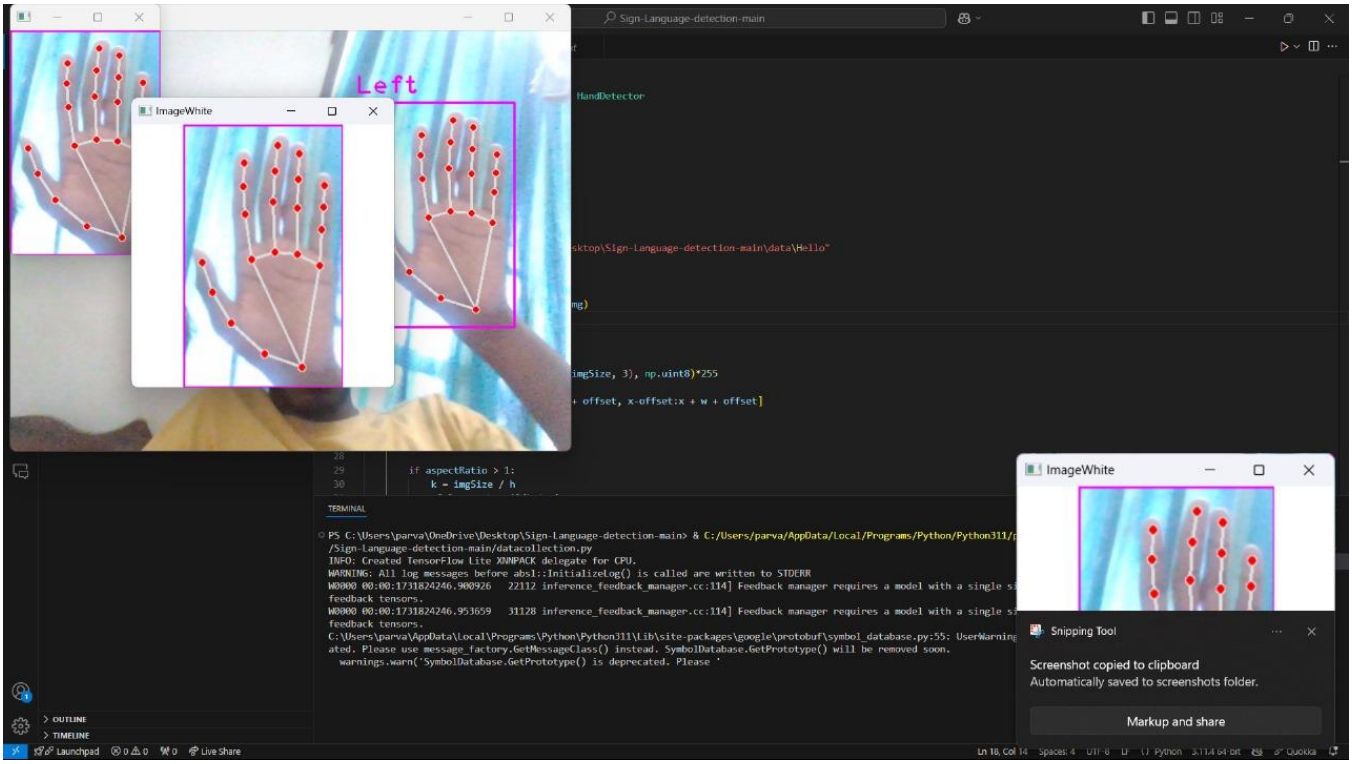
```
            imgWhite[:, wGap: wCal + wGap] = imgResize

        else:
            k = imgSize / w
            hCal = math.ceil(k * h)
            imgResize = cv2.resize(imgCrop, (imgSize, hCal))
            imgResizeShape = imgResize.shape
            hGap = math.ceil((imgSize - hCal) / 2)
            imgWhite[hGap: hCal + hGap, :] = imgResize

        cv2.imshow('ImageCrop', imgCrop)
        cv2.imshow('ImageWhite', imgWhite)

    cv2.imshow('Image', img)
    key = cv2.waitKey(1)
    if key == ord("s"):
        counter += 1
        cv2.imwrite(f'{folder}/Image_{time.time()}.jpg', imgWhite)
        print(counter)
```

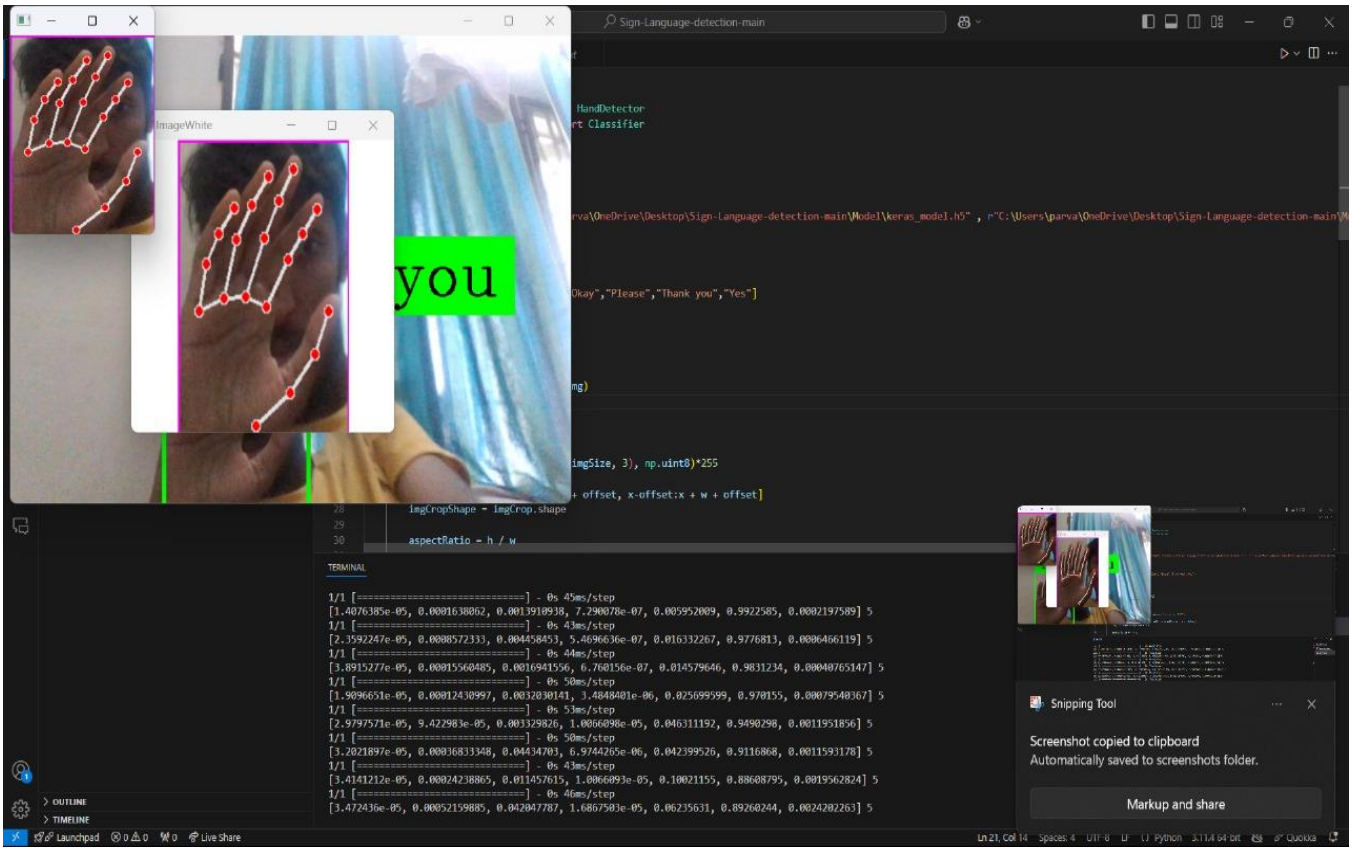
4.1.2 Test File

```
import cv2
from cvzone.HandTrackingModule import HandDetector
from cvzone.ClassificationModule import Classifier
import numpy as np
import math

cap = cv2.VideoCapture(0)
detector = HandDetector(maxHands=1)
classifier = Classifier("Model/keras_model.h5" , "Model/labels.txt")
offset = 20
imgSize = 300
counter = 0

labels = ["Hello","I love you","No","Okay","Please","Thank you","Yes"]


while True:
    success, img = cap.read()
    imgOutput = img.copy()
    hands, img = detector.findHands(img)
    if hands:
        hand = hands[0]
        x, y, w, h = hand['bbox']
```

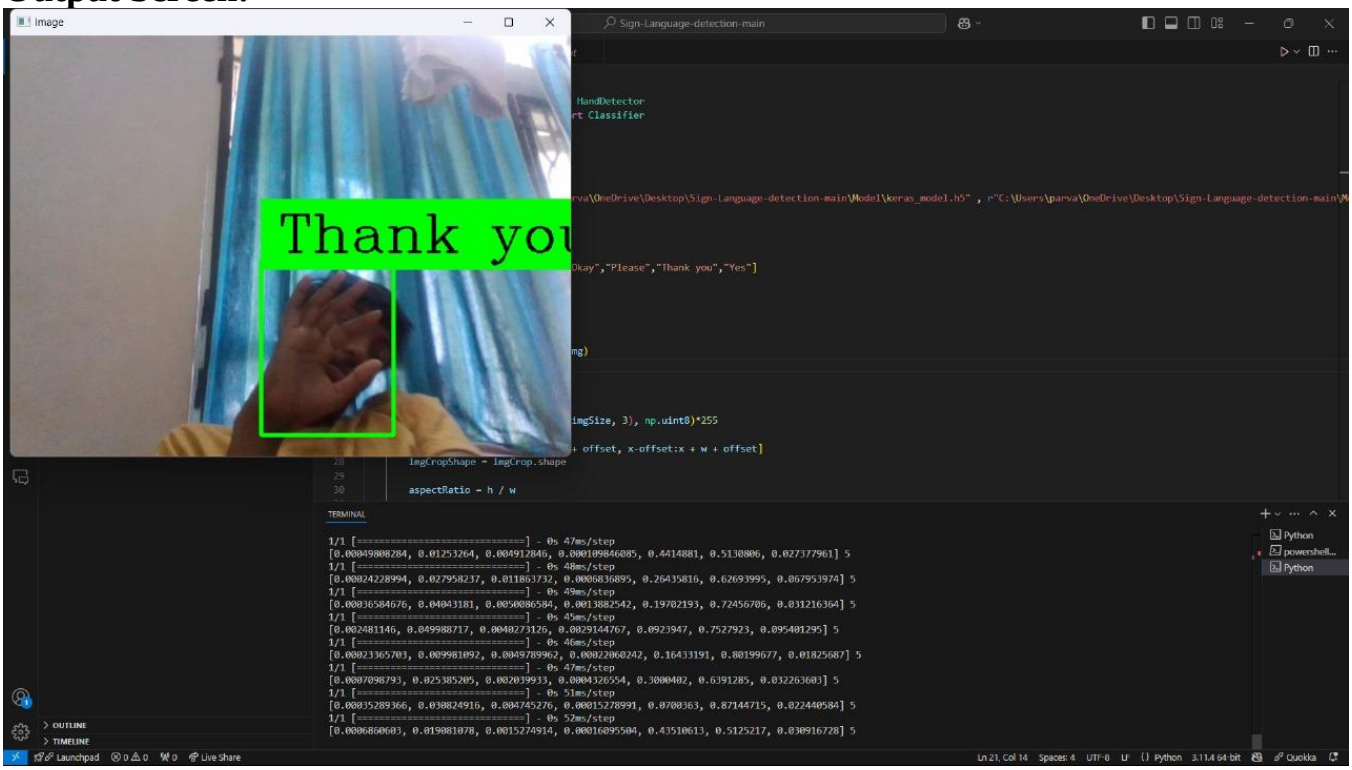
```
    imgWhite = np.ones((imgSize, imgSize, 3), np.uint8)*255

    imgCrop = img[y-offset:y + h + offset, x-offset:x + w + offset]
    imgCropShape = imgCrop.shape

    aspectRatio = h / w

    if aspectRatio > 1:
        k = imgSize / h
        wCal = math.ceil(k * w)
        imgResize = cv2.resize(imgCrop, (wCal, imgSize))
        imgResizeShape = imgResize.shape
        wGap = math.ceil((imgSize-wCal)/2)
        imgWhite[:, wGap: wCal + wGap] = imgResize
        prediction , index = classifier.getPrediction(imgWhite, draw= False)
        print(prediction, index)

    else:
        k = imgSize / w
        hCal = math.ceil(k * h)
        imgResize = cv2.resize(imgCrop, (imgSize, hCal))
        imgResizeShape = imgResize.shape
        hGap = math.ceil((imgSize - hCal) / 2)
        imgWhite[hGap: hCal + hGap, :] = imgResize
        prediction , index = classifier.getPrediction(imgWhite, draw= False)


    cv2.rectangle(imgOutput,(x-offset,y-offset-70),(x -offset+400, y - offset+60-
50),(0,255,0),cv2.FILLED)

    cv2.putText(imgOutput,labels[index],(x,y-30),cv2.FONT_HERSHEY_COMPLEX,2,(0,0,0),2)
    cv2.rectangle(imgOutput,(x-offset,y-offset),(x + w + offset, y+h + offset),(0,255,0),4)

    cv2.imshow('ImageCrop', imgCrop)
    cv2.imshow('ImageWhite', imgWhite)

  cv2.imshow('Image', imgOutput)
  cv2.waitKey(1)
```

Data collection:




ImageWhite
you
Sign-Language-detection-main
HandDetector
rt Classifier
Okay","Please","Thank you","Yes"]
imgSize, 3), np.uint8)*255
+ offset, x-offset:x + w + offset]
imgCropShape = imgCrop.shape
aspectRatio = h / w
TERMINAL
1/1 [==============================] - 0s 45ms/step
[1.4076385e-05, 0.0001638062, 0.0013910938, 7.290078e-07, 0.005952009, 0.9922585, 0.0002197589] 5
1/1 [==============================] - 0s 43ms/step
[2.3592247e-05, 0.0008572333, 0.004458453, 5.4696636e-07, 0.016332267, 0.9776813, 0.0006466119] 5
1/1 [==============================] - 0s 44ms/step
[3.8915277e-05, 0.00015560485, 0.0016941556, 6.760156e-07, 0.014579646, 0.9831234, 0.00040765147] 5
1/1 [==============================] - 0s 50ms/step
[1.9096651e-05, 0.00012430997, 0.0032030141, 3.4848401e-06, 0.025699599, 0.970155, 0.00079540367] 5
1/1 [==============================] - 0s 53ms/step
[2.9797571e-05, 9.422983e-05, 0.003329826, 1.0066098e-05, 0.046311192, 0.9490298, 0.0011951856] 5
1/1 [==============================] - 0s 50ms/step
[3.2021897e-05, 0.00036833348, 0.04434703, 6.9744265e-06, 0.042399526, 0.9116868, 0.0011593178] 5
1/1 [==============================] - 0s 43ms/step
[3.4141212e-05, 0.00024238865, 0.011457615, 1.0066093e-05, 0.10021155, 0.88608795, 0.0019562824] 5
1/1 [==============================] - 0s 46ms/step
[3.472436e-05, 0.00052159885, 0.042047787, 1.6867503e-05, 0.06235631, 0.89260244, 0.002420226 3] 5
Snipping Tool
Screenshot copied to clipboard
Automatically saved to screenshots folder.
Markup and share
OUTLINE
TIMELINE
Launchpad
Live Share

Output Screen:



The model successfully recognizes the handwritten Text and display ”Thank You”

4.3: Testing

Software testing is a critical element of software quality assurance and represents the ultimate review of specification, design and code generation.

CHAPTER 5

CONCLUSION & FUTURE SCOPE

5.1: Conclusion

The development of technology to convert **sign language into text** offers significant potential for bridging communication gaps between the Deaf and Hard of Hearing (DHH) community and the wider world. Using tools like computer vision, machine learning, and natural language processing, it's now possible to translate sign language into written words, improving access to communication for those who use sign language.

Recent advancements have focused on areas like:

- **Gesture Recognition**: Using cameras and sensors to detect and interpret hand and body movements that make up sign language.
- **Context Awareness**: Developing systems that can understand not just gestures, but also facial expressions and body language, which are integral parts of sign language.
- **Real-time Translation**: Creating technologies that can instantly convert sign language into text, facilitating more natural conversations.

However, there are still challenges to address:

- **Accuracy**: Sign language varies widely across different cultures, regions, and even individuals. Making sure the technology can recognize these differences accurately is key.
- **Context**: Sign language often involves subtle cultural and contextual cues that are difficult for machines to understand.
- **Real-world Application**: For these systems to be truly effective, they must work in different environments and with a wide range of users, from well-lit spaces to outdoor settings with varying lighting.

5.2: Future Scope

The future of **sign language translation technology** is full of exciting possibilities. Here are a few key areas where we can expect to see progress:

1. **Better Recognition Systems**:
    a. As AI and machine learning continue to evolve, these systems will become more accurate, reducing mistakes and improving their ability to handle different sign languages and contexts.
2. **Augmented Reality (AR) and Virtual Reality (VR)**:
    a. AR and VR could revolutionize the way we learn and interact using sign language. Virtual avatars could provide real-time, immersive translation, while AR could help people understand and practice sign language in dynamic environments.
3. **Universal Sign Language Systems**:
    a. Efforts to create universal or adaptable sign language systems could allow people from different countries and cultures to communicate more easily with one another, breaking down language barriers.
4. **Wearable Technology**:
    a. Wearables like smart gloves or haptic devices could capture more detailed hand gestures and offer more accurate translation, especially in real-world scenarios where visual recognition might be challenging.
5. **Integration with Other AI Technologies**:
    a. As AI technology advances, sign language translation could be integrated into **virtual assistants** or **chatbots**, allowing sign language users to interact with machines more naturally. AI-powered translation tools could also be embedded in smartphones and other devices for real-time communication.
6. **Cross-Platform Accessibility**:
    a. The goal is for sign language translation tools to work seamlessly across multiple devices and platforms, from public services to online communication tools, providing greater accessibility to users wherever they are.
7. **Ethical and Privacy Considerations**:

With the rise of data-driven technologies, it’s crucial that we also address privacy concerns. Since sign language involves personal gestures, ensuring that data is securely handled and that users' identities are protected will be essential.

CHAPTER 6

BIBILOGRAPHY

- [1] Signet: A Deep Learning based Indian Sign Language Recognition System by Sruthi C. J and Lijiya A Member, IEEE. 2019.
- [2] Deep learning-based sign language recognition system for static signs by Ankita Wadhawan1 • Parteek Kumar. 2020.
- [3] Bhagat, N. K., Vishnusai, Y., & Rathna, G. N. (2019). Indian Sign Language by Gesture Recognition using Image Processing and Deep Learning. 2019 Digital, Image Computing: Techniques and Applications (DICTA).
- [4] Likhar, P., Bhagat, N. K., & G N, R. (2020). Deep Learning Methods forIndian Sign Language Recognition. 2020 IEEE 10th International Conferenceon Consumer Electronics (ICCE-Berlin).
- [5] Dudhal, A., Mathkar, H., Jain, A., Kadam, O., & Shirole, M. (2019). Hybrid SIFT Feature Extraction Approach for Indian Sign Language Recognition System Based on CNN. Lecture Notes in Computational Vision and Biomechanics, 727–738.
- [6] Sign Language Recognition Using Convolutional Neural Networks International Journal IJRITCC International Journal IJRITCC N. Priyadharsini RAJESWARI N MCA. 2017.