SIGN LANGUAGE RECOGNITION

MINOR PROJECT REPORT

SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE AWARD OF THE DEGREE OF

Bachelor of Technology

(Computer Science and Engineering)

Submitted By:

Karandeep Singh (2004608)
Manav Loona (2004620)
Manjot Singh Bhatia (2004622)

Submitted To:

Prof. Amanpreet Singh Brar
Associate Professor

Department of Computer Science and Engineering

Guru Nanak Dev Engineering College

Ludhiana, 141006

ABSTRACT

Sign language recognition is a field of study that aims to develop technology to recognize and interpret sign language gestures and motions. The development of sign language recognition technology is an important step towards promoting inclusivity and accessibility for people with hearing disabilities. The goal of this project is to develop a sign language recognition system that can accurately detect and interpret hand gestures and motions used in American Sign Language (ASL). This system will be trained using machine learning algorithms, such as deep neural networks, on a dataset of annotated ASL gestures. The performance of the system will be evaluated using metrics such as accuracy, precision, and recall, and further improvements to the system's accuracy and efficiency will be explored by incorporating additional data augmentation techniques and sensor modalities.

The expected outcome of this project is a working sign language recognition system that can translate ASL gestures into text or speech, and potentially be used to develop assistive devices for deaf individuals or integrated into existing communication tools. Ultimately, the success of this project will contribute to the development of technology that can improve communication for individuals with hearing disabilities, promoting inclusivity and accessibility.

ACKNOWLEDGEMENT

We are highly grateful to the Dr. M.S. Saini, Director, Guru Nanak Dev Engineering College (GNDEC), Ludhiana, for providing this opportunity to carry out the major project work at Sign Language Recognition.

The constant guidance and encouragement received from Dr. Parminder Singh H.O.D. CSE Department, GNDEC Ludhiana has been of great help in carrying out the project work and is acknowledged with reverential thanks.

We would like to express a deep sense of gratitude and thanks profusely to Prof. Amanpreet Singh Brar, without his wise counsel and able guidance, it would have been impossible to complete the project in this manner.

We express gratitude to other faculty members of computer science and engineering department of GNDEC for their intellectual support throughout the course of this work. Finally, we are indebted to all whosoever have contributed in this report work.

Manjot Singh Bhatia
Manav Loona
Karandeep Singh

LIST OF FIGURES

Fig. No.	Figure Description	Page No.
3.1	Design Approach	13
3.2	Block Diagram of the system	14
3.3	Raw image of an alphabet	15
3.4	An alphabet after applying filters	17
3.5	Convolutional Neural Network (CNN)	18
3.6	Sign Language of respective alphabets and numbers	20
3.7	Overall flowchart for sign language recognition system	21
3.8	User Interface Design	22
3.9	Methodology of Work	24
4.1	Gantt chart	28
5.1	Data Collection(Alphabet R)	32
5.2	Data Collection(Alphabet B)	32
5.3	Data Pre-Processing(Alphabet R)	33
5.4	Data Pre-Processing (Alphabet B)	33
5.5	Output of trained data	33
5.6	Conversion of Sign Language to Text	34
5.7	Window after clicking About button	34

Table of Contents

1. INTRODUCTION ……………………………………………………………………………1-5

1.1 Introduction of the project ……………………………………….…..1

1.2 Project Category …….……………….…………..………………...1-2

1.3 Objectives of the project ……….………..…..……………………..2

1.4 Problem Formulation ……………….………….……………..……..2

1.5 Identification/Reorganization of Need …........………………..…...2-3

1.6 Existing System ……...………….………......………..…………3-4

1.7 Proposed System …….…………….………....……………..……..4

1.8 Unique features of the System …….…….......…………...………..4-5

2. RQUIREMENT ANALYSIS AND SYSTEM SPECIFICATION …...………………..…6-12

2.1 Feasibility Study ……………………….…………………………...6-7

2.2 Software Requirement Specification Document …...……………7-10

2.3 Validation ………………………………..……………..…...11

2.4 Expected Hurdles ……………………………..………....……...11

2.5 SDLC Model ….……...……………...……………...………..11-12

3. SYSTEM DESIGN ………………………...…………………………………...………...13-24

3.1 Design Approach ...…………………………………..……………….13

3.2 Detail Approach ...……...…………….…………………………14-20

3.3 System Design using Structured analysis and design tools …….20-21

3.4 User Interface Design ……………….……………………………21-22

3.5 Methodology …….……………..……………………………23-24

4. IMPLEMENTATION, TESTING AND MAINTENANCE ………………...………25-29

4.1 Introduction to Languages, IDE's, Tools and Technologies 25-26

4.2 Coding standards of Python Language used …....…………………27

4.3 Gantt Chart ...………………..………..……..…………………..27-28

4.4 Testing Techniques and Test Plans ..……..……………………..28-29

5. RESULTS AND DISCUSSIONS ….……………………………………...……….30-34

5.1 User interface representation ….………………..……………...........30

5.2 Various Modules of the system ……...……………….………………30-31

5.3 Snapshot of the system ..………………..……………………..……......31-34

6. CONCLUSION AND FUTURE SCOPE ………………..………………………………………35

7. REFERENCES……………………………………………………………………....36

1. Introduction

1.1 Introduction of the project

Sign Language Recognition focuses on developing software or hardware solutions that can recognize and interpret sign language gestures and movements made by individuals with hearing or speech impairments. Sign language is a visual language that uses a combination of hand gestures, facial expressions, and body language to convey information and meaning. It is used by millions of people worldwide as their primary means of communication. However, for those who do not understand sign language, communication with individuals who use it can be challenging.

The goal of sign language recognition is to bridge this communication gap by developing technology that can accurately interpret sign language gestures and translate them into spoken or written language. This can help individuals who use sign language to communicate more effectively with those who do not.

There are several approaches to sign language recognition, including computer vision and machine learning techniques. These methods involve training models to recognize and interpret different sign language gestures and movements based on visual data. With the help of these techniques, it is possible to create systems that can recognize a wide range of sign language gestures and translate them into spoken or written language in real-time.

1.2 Project Category

Our project falls under the category of Application or System Development

This is because the project involves the development of a sign language recognition system, which is a software application or system that can recognize and interpret hand gestures used in American Sign Language (ASL). The system may involve the use of machine learning algorithms and sensor modalities to improve its accuracy and efficiency. The ultimate goal of the project is to develop a working sign language recognition system that can translate ASL gestures into text and potentially be used to develop assistive devices or integrated into existing communication tools.

1.3 Objectives of the project

- To recognize hand gestures which include 26 English alphabets (A-Z) and 10 digits (0-9) using Convolutional Neural Network.
- To convert sign language into words by an algorithm or a model.
- To show on optical viewfinder of camera module what a particular position of hand means with respect to sign language.

1.4 Problem Formulation

- Develop a sign language recognition system that can accurately recognize and interpret American Sign Language (ASL) gestures in real-time.
- The system should be able to translate the ASL gestures into text output that can be easily understood by hearing individuals.
- The system should be efficient, accurate, and user-friendly to support seamless communication between deaf and hearing individuals.

1.5 Identification/Reorganization of Need

In a sign language recognition system, the identification or recognition of need refers to the process of understanding the specific needs and requirements of deaf and hearing individuals in terms of communication. This involves identifying the challenges and limitations of current solutions, as well as the potential benefits and opportunities that can be gained through the development of a sign language recognition system.

The following steps can be taken to identify and recognize the need for a sign language recognition project:

- **Research the communication needs of deaf and hearing individuals:** This involves gathering information about the communication challenges faced by deaf and hearing individuals, the limitations of current solutions, and the potential benefits of a sign language recognition system.
- **Conduct a needs analysis:** This involves analyzing the specific needs and requirements of deaf and hearing individuals in terms of communication, including the types of gestures and signs used in ASL, the level of accuracy and speed required for real-time communication, and the need for a user-friendly and accessible interface.
- **Engage with stakeholders:** This involves consulting with stakeholders such as deaf and hearing individuals, sign language interpreters, and healthcare professionals to gain a better understanding of their needs and perspectives on the communication barriers faced by deaf and hearing individuals.
- **Identify potential solutions:** Based on the needs analysis and stakeholder input, identify potential solutions for addressing the communication barrier between deaf and hearing individuals, such as the development of a sign language recognition system.
- **Evaluate the feasibility of the solution:** Consider the technical, social, and economic feasibility of the proposed solution, including factors such as the availability of resources, the potential impact on stakeholders, and the potential for scalability and sustainability.

1.6 Existing System

In the context of sign language recognition, existing systems refer to systems that have already been developed and are currently available for use. These systems have been tested and evaluated for their accuracy, reliability, and usability, and are typically based on established algorithms and modalities. In these existing systems a user may use different inbuilt libraries of Python like hand detection, cvzone and available trained data which may not be perfect for everyone to use efficiently. It contains limited amount of data and is difficult to manipulate. Existing systems have varying levels of accuracy and robustness, and may have different design goals or user requirements. Some existing systems may focus on recognizing specific signing gestures or languages, while others may provide real-time translation or personalized user interfaces.

1.7 Proposed System

In the context of sign language recognition, a proposed system refers to a system that is currently in the design or development phase and has not yet been implemented. Proposed systems aim to address limitations or gaps in existing systems. Proposed systems, on the other hand, may incorporate new or innovative algorithms, models and may have different design goals or user requirements. Proposed systems may also aim to incorporate new features or technologies that are not currently available in existing systems. For example, a proposed system may use advanced machine learning algorithms to improve recognition accuracy or incorporate novel sensors to capture signing gestures in different ways. In our project we uses a sequential model of CNN and creates our own dataset which increases the flexibility and numbers of layers can be increased or decreased according to requirement which also increases the efficiency and hence results may be improved.

1.8 Unique Features of the System

Sign language recognition projects have some unique features compared to other computer vision or natural language processing projects. Here are some of the unique features of sign language recognition system:

- **Multimodal inputs:** Sign language recognition systems typically take in input data as visual data (images or videos of hand gestures). This requires the system to handle multiple modalities and integrate them to produce a meaningful output.
- **Variability in hand shapes and movements:** Sign language involves a wide range of hand shapes and movements that can vary greatly between different sign languages, dialects, and even individual signers. The system must be able to recognize these variations and generalize across them.
- **Limited training data:** Compared to other computer vision or natural language processing tasks, sign language recognition has limited annotated data available for training and evaluation, especially for under-resourced sign languages. This can make it challenging to develop accurate and robust recognition models.
- **Human-centric evaluation metrics:** The performance of a sign language recognition system is typically evaluated based on its ability to recognize signs accurately and quickly, as well as its ability to convey the intended meaning of the signed message. These evaluation metrics are often human-centric and require human judgments or feedback, which can be time-consuming and subjective.

2. Requirement Analysis and System Specification

2.1 Feasibility Study

- Technical Feasibility: It refers to the ability of the proposed system to be developed and implemented with the available technology and resources. Technical feasibility study is an important aspect of a sign language recognition project that assesses the system's ability to be developed and implemented using available technology and resources.

  It involves evaluating the existing algorithms, models and as well as the potential for integrating new technologies into the system. In terms of existing algorithms, the feasibility study should analyze the effectiveness of various approaches such as machine learning algorithms, computer vision-based algorithms. The study should also evaluate the accuracy, robustness, and real-time performance of the algorithms.
- Economical Feasibility: Economical feasibility studies determine the financial viability of the proposed system. This study includes analyzing the costs associated with developing and implementing the system, as well as the potential benefits and returns on investment.

  This study in context of sign language recognition involves considering factors such as the cost of hardware and software, development and implementation costs, as well as operational and maintenance costs. These costs can include the salaries of software engineers and developers, cost of training and support, and expenses related to purchasing hardware and software. The study should also analyze the potential benefits of the system, such as the potential for increasing accessibility and improving communication for individuals who are deaf or hard of hearing.

- Operational Feasibility: This study evaluates the ability of the proposed system to be used effectively in the intended environment. This study includes analyzing the system's usability, user acceptance, and potential impact on daily operations.

  The operational feasibility study in context of sign language recognition project involves assessing the usability and accessibility of the system for users, including individuals who are deaf or hard of hearing. It also involves evaluating the system's ability to integrate with existing technologies and infrastructure, such as software applications, network systems, and hardware devices.

2.2 Software Requirement Specification Document

2.2.1 Data Requirement

- **Visual data:** Sign language recognition systems require visual data in the form of images of sign language gestures. These can be captured using specialized cameras or sensors, or obtained from publicly available datasets. The quality and resolution of the visual data can have a significant impact on the accuracy and robustness of the recognition system.
- **Annotated data:** Sign language recognition systems require annotated data that labels each gesture with the corresponding meaning or concept. This is typically done manually by human annotators, who may use specialized software or tools to facilitate the annotation process.

2.2.2 Functional Requirement

The functional requirements are the specific features and capabilities that the system must have in order to perform its intended task of recognizing sign language gestures and conveying their meaning. The functional requirement includes the following:

- **Gesture detection:** The system must be able to detect and localize sign language gestures within a video stream or image.
- **Gesture recognition:** The system must be able to recognize the specific sign language gestures and classify them into their corresponding meanings or concepts.
- **Real-time performance:** The system should be able to operate in real-time, with low latency, to enable natural and fluid communication between signers and non-signers.
- **Robustness:** The system should be able to recognize sign language gestures accurately and reliably under a wide range of conditions, such as varying lighting, and signer appearance.
- **Adaptability:** The system should be able to adapt to different sign languages, dialects, and signers, as well as different communication contexts and scenarios.
- **Privacy and security:** The system should ensure the privacy and security of the signers' data and communications, particularly in sensitive contexts such as healthcare or legal settings.

2.2.3 Performance Requirement

- **Performance Requirement**

  Accuracy is one of the most important performance requirements in sign language recognition. The system must be able to accurately recognize signs with a high degree of precision.

  Speed is another important performance requirement, particularly in real-time applications such as interpreting for deaf individuals. The system must be able to recognize signs quickly enough to provide a smooth and seamless user experience.

  Robustness is also crucial, as the system should be able to recognize signs accurately even under challenging conditions such as low lighting or occlusions.

- **Software Requirements**

  Programming Languages: Knowledge of programming languages such as Python will be required to develop the software for Sign Language Recognition.

  Development Environments: Visual Studio Code, or Eclipse may be required for software development.

  Libraries and Frameworks: Machine learning and computer vision libraries and frameworks such as TensorFlow, Keras and OpenCV will be needed to develop models and algorithms for Sign Language Recognition.

  Datasets: Access to Sign Language Recognition datasets such as ASL Alphabet Dataset is required to train the models and algorithms.

- **Hardware Requirements**

  Web Camera: A high-quality web camera is required to capture the gestures and movements made by the individuals using sign language.

  Processing Power: Sign Language Recognition may require high processing power, especially for real-time recognition. Thus, a powerful processor, such as an Intel Core i5 or higher, or a dedicated graphics card may be needed.

  Memory: Sufficient RAM, such as 8 GB or more, will be required to handle the large datasets and models used for Sign Language Recognition.

2.2.4 Dependability Requirement

Dependability requirements refer to the system's ability to perform reliably and consistently over time.

- Availability refers to the system's ability to be accessible and functional when needed. For a sign language recognition system, this means that the system should be available to users when they need it, whether it's for real-time interpretation or for accessing archived sign language images.

- Reliability refers to the system's ability to perform consistently and accurately over time. This includes factors such as the accuracy and speed of sign recognition, as well as the system's ability to handle varying sign language accents.

2.2.5 Maintainability requirement

Maintainability requirements refer to the system's ability to be updated, repaired, and enhanced over time.

- **Modularity:** The system should be modular, which means that it should be composed of separate components that can be updated or replaced without affecting the entire system.
- **Scalability:** The system should be designed to be scalable, which means that it should be able to handle an increasing volume of data over time.

2.2.6 Security Requirement

Sign language recognition system security depend upon various factors such as data privacy , authentication and secure storage etc.

2.2.7 Look and Feel Requirement

The look and feel requirement refers to the user interface design and user experience of the system. This includes factors such as the visual design and usability of the system.

- **Visual design:** The system should have a clean and professional visual design that is appropriate for its intended users. The design should be consistent across all components of the system.

- **Usability:** The system should be easy to use, with clear and concise instructions and feedback provided to the user.

2.3 Validation

Validation refers to the process of verifying that the system is performing as expected and meeting its intended goals and requirements. It helps to ensure that the system is accurate, reliable, and effective for its intended users. In this various types of validation like data validation, algorithm validation and performance validation is implemented on the system and it was found that the implementation is working well.

2.4 Expected Hurdles

Sign language recognition system can face several challenges and hurdles that can impact the accuracy and performance of the system. Some of the expected hurdles are:

- **Data availability:** Sign language recognition systems require large amounts of data to train their algorithms. However, it can be challenging to collect and curate high-quality sign language data, particularly for lesser-known sign languages or dialects.
- **Environmental factors:** Sign language recognition systems can be sensitive to environmental factors such as lighting, background noise, and camera placement. These factors can impact the accuracy and reliability of the system in real-world settings.
- **Hardware limitations:** Sign language recognition systems require specialized hardware, to accurately capture and process sign language data. These hardware requirements can be expensive or challenging to obtain in certain settings.

2.5 SDLC Model to be used

The following are the phases of the SDLC model to be used:

- **Planning:** In this phase, the project goals, scope, requirements, and timelines are defined. This phase also involves identifying the project stakeholders, creating a project roadmap, and assembling the project team.
- **Analysis:** In this phase, the system requirements are analyzed and documented. This includes defining the user stories, use cases, and other functional and non-functional requirements. The analysis phase also involves conducting research on existing sign language recognition systems and technologies.
- **Design:** In this phase, the system architecture and design are developed based on the requirements and analysis phase. The design phase also includes creating wireframes, prototypes, and other design artifacts that will be used to guide the development process.
- **Implementation:** In this phase, the actual development of the system begins. The system is developed in iterations or sprints, with each sprint delivering a working piece of the system that can be tested and evaluated.
- **Testing:** In this phase, the system is tested to ensure that it meets the requirements and performs as expected. Testing can be performed at multiple levels, including unit testing, integration testing, and system testing.
- **Deployment:** In this phase, the system is deployed to a live environment, such as a web server or mobile app store. This phase also involves preparing the system for release and training the users on how to use it.
- **Maintenance:** In this phase, the system is monitored and maintained to ensure that it continues to perform as expected and meets the changing needs of the users. This may involve fixing bugs, adding new features, or upgrading the system to new technologies.

3. System Design

3.1 Design Approach

We have used both Function oriented and Object oriented Approach in our project. As with function oriented we can have easy flow of information and in object oriented we can use concepts like information hiding, benefits of constructor and destructor etc.

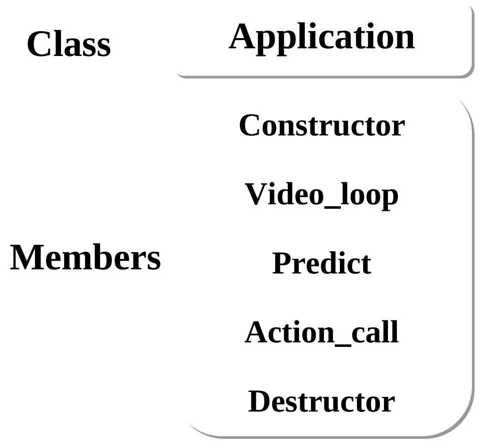


Fig. 3.1 Design Approach

- Function-oriented approach could be used to break down the various stages of the recognition process, such as image capture, feature extraction, and gesture classification, into smaller functions that can be developed and tested independently.
- Object-oriented approach could be used to represent sign language gestures as objects, with attributes such as shape, orientation, and movement, and methods for recognizing and classifying those gestures.

3.2 Detail Design

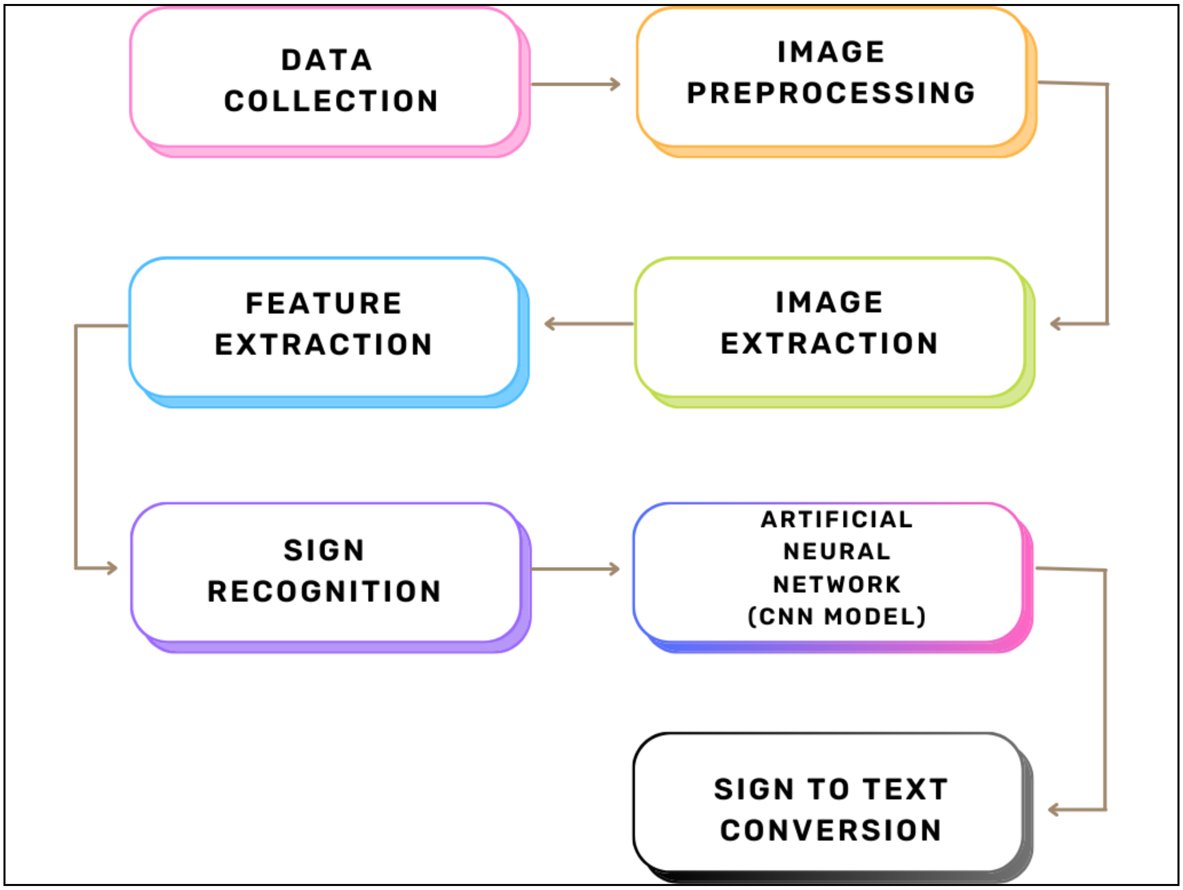


Fig. 3.2 Block Diagram of the system

3.2.1 Recognition of Hand Gestures:

Data Collection:

Data collection is the process of gathering and recording information or data for a particular purpose. In the context of sign language recognition, data collection involves gathering video or motion capture data of people performing various sign language gestures and sentences.

This data is then used to train machine learning models that can recognize and interpret sign language gestures and sentences. The more data that is collected, the more accurate and robust the machine learning models become, resulting in better sign language recognition.

Data collection is crucial in sign language recognition because sign languages are complex and often involve subtle differences in hand shape, movement, and position. By collecting a large and diverse dataset of sign language gestures, the machine learning models can learn to recognize these subtleties and accurately interpret the intended meaning behind the signs.

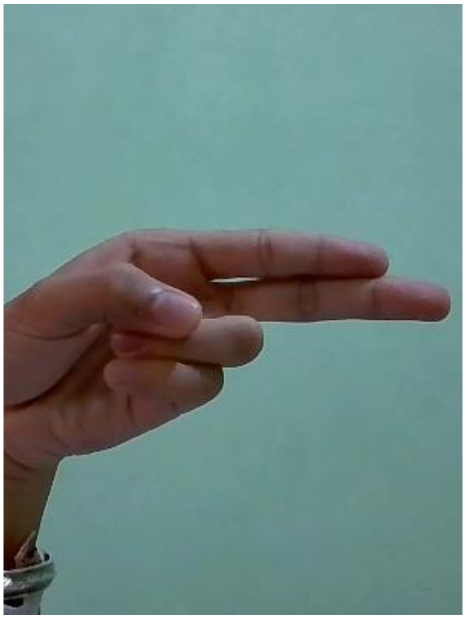


Fig. 3.3 Raw image of an alphabet

Data Pre-Processing:

Data preprocessing is the process of cleaning, transforming, and preparing data for analysis. In the context of sign language recognition, data preprocessing involves preparing the\ motion capture data collected during data collection for use in machine learning models.

Various Filters Used In Image Pre-Processing:

In sign language recognition, image preprocessing techniques involve applying various filters to the images to remove noise, enhance the relevant features, and improve the overall quality of the images. Here are some commonly used filters in image preprocessing for sign language recognition:

- **Grayscale filter** is a common image preprocessing technique that involves converting a color image into a grayscale image. The grayscale filter converts a color image to grayscale by taking the average of the red, green, and blue (RGB) values for each pixel and assigning the same value to each of the three color channels.
- **Gaussian blur** filter is a technique used in image preprocessing to smooth an image and reduce the noise present in the image. It works by convolving the input image with a Gaussian function, which is a bell-shaped curve that assigns weights to each pixel based on its distance from the center pixel. The closer a pixel is to the center, the higher the weight assigned to it.
- **Thresholding** is a common image preprocessing technique used in sign language recognition that involves converting a grayscale image into a binary image by separating the pixels into two categories based on a threshold value. Pixels with brightness values above the threshold value are assigned one value (often white), while pixels with brightness values below the threshold value are assigned another value (often black).
- **Adaptive thresholding** is a technique that adjusts the threshold value locally in different regions of the image. This can be useful in sign language recognition, where lighting conditions and background can vary greatly between different signs and signers.

Overall, grayscale, Gaussian blur, threshold, and adaptive threshold are all useful image processing techniques that can help to improve the quality of sign language data and increase recognition accuracy.

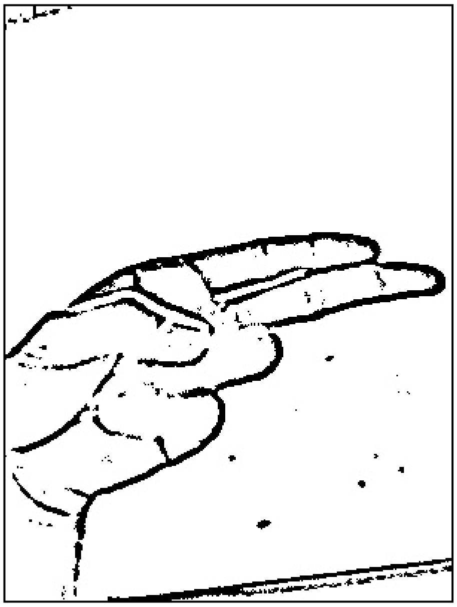


Fig. 3.4 An alphabet after applying filters (Image preprocessed)

3.2.2 Conversion Of Sign Language Into Words :

Convolutional Neural Network (CNN):

A Convolutional Neural Network (CNN) is a type of deep learning algorithm designed to recognize patterns in images, videos, and other multi-dimensional data. CNNs are widely used in computer vision tasks such as object detection, image recognition, and segmentation.

The basic idea behind CNNs is to extract relevant features from the input data through convolutional layers. These layers apply a set of filters (also called kernels) to the input data, which convolve (or slide) over the input data and produce a feature map. Each filter learns to detect a specific feature in the input data, such as edges, corners, or textures.

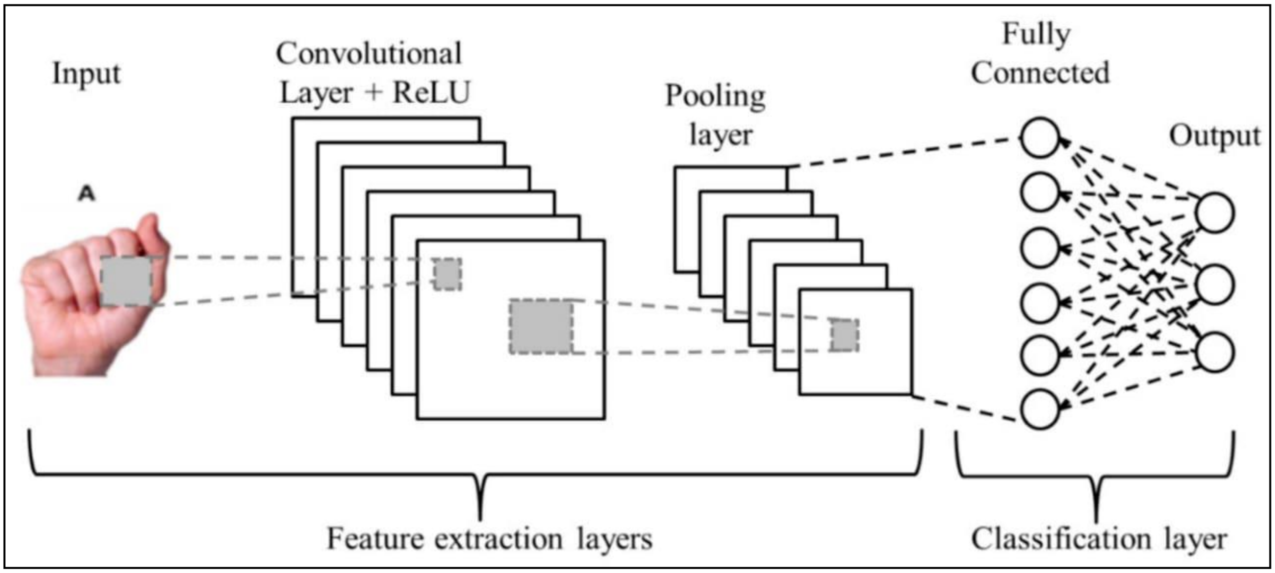


Fig. 3.5 Convolutional Neural Network (CNN)

Sequential Model of Convolutional Neural Network (CNN):

In our project we uses the Sequential model of CNN. A Sequential model of CNN is a type of neural network architecture used for image processing and computer vision tasks. It is a linear stack of layers where the output of one layer is fed as input to the next layer. This model is particularly useful when the data flows from input to output in a sequential order.

The main advantage of using a sequential model in CNN is that it allows for the automatic feature extraction and learning of hierarchical representations from the input data. This is achieved through the use of convolutional layers, which apply a set of filters to the input image and generate feature maps that highlight important patterns and features in the image. The output of the convolutional layers is then fed into pooling layers, which down sample the feature maps and reduce the dimensionality of the data.

Here are some of the main components of a Sequential model of CNN:

Input layer: This layer is responsible for receiving the input image and preparing it for further processing. The input layer usually takes a 3D tensor representing the image with dimensions (height, width, channels).

Convolutional layer: The convolutional layer is the primary building block of CNNs. It applies a set of filters (also known as kernels or weights) to the input image to generate a set of feature maps. Each filter extracts specific features from the image by performing element-wise multiplication with the input image and summing the results.

Activation layer: The activation layer applies a non-linear activation function to the output of the convolutional layer. This allows the CNN to learn more complex and abstract features from the input image. Common activation functions include ReLU (Rectified Linear Unit), sigmoid, and tanh.

Pooling layer: The pooling layer is responsible for downsampling the feature maps generated by the convolutional layer. This helps to reduce the dimensionality of the data and make the model more efficient. The most common pooling operation is max pooling, where the maximum value in each subregion of the feature map is selected.

Flatten layer: The flatten layer is responsible for flattening the output of the previous layer into a 1D tensor. This prepares the data for input into the fully connected layers.

Fully connected layer: The fully connected layer is a traditional neural network layer that connects every neuron in the layer to every neuron in the previous layer. The fully connected layer takes the flattened tensor from the previous layer as input and produces an output tensor representing the probability distribution over the possible classes.

3.2.3 Conversion to Word:

The project will recognize and interpret sign language gestures and movements made by individuals with hearing or speech impairments and then convert that gesture into words.

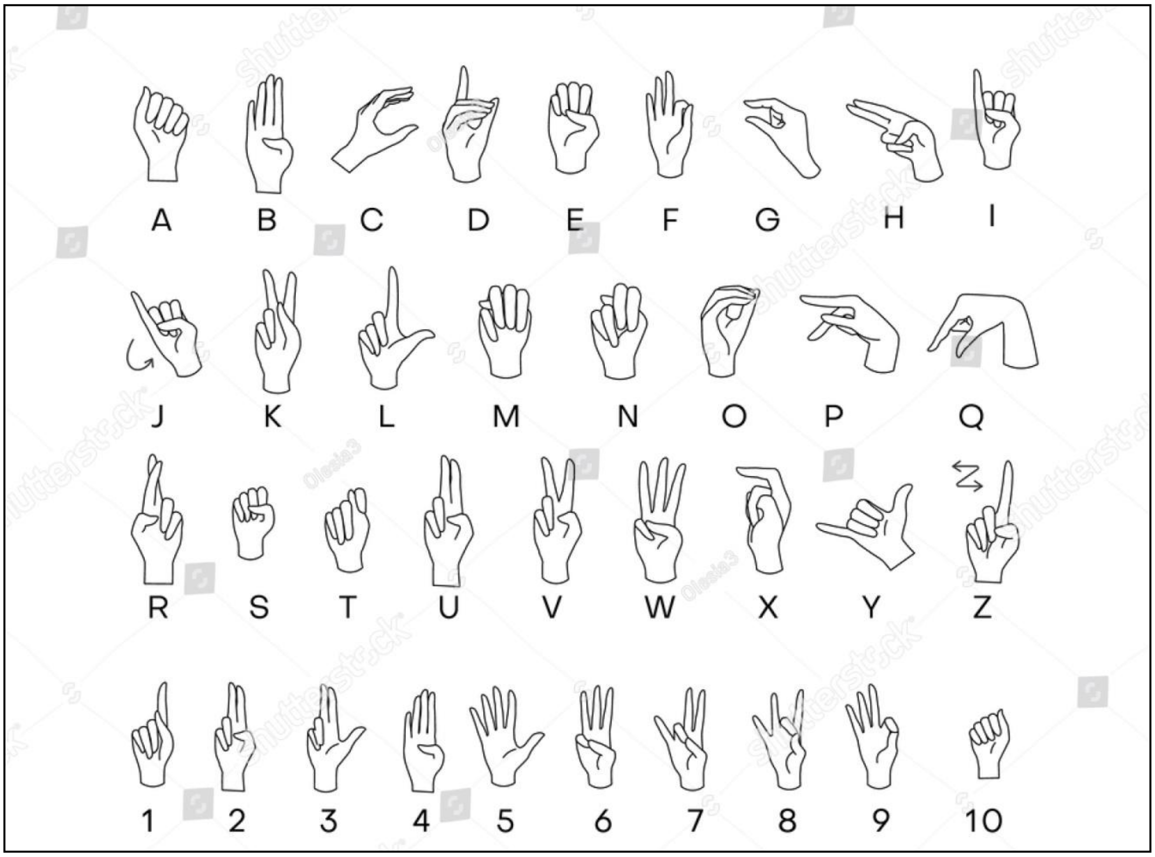


Fig. 3.6 Sign Language of respective alphabets and numbers

3.3 System Design using Structured analysis and design tool such as flowchart

A flowchart is a graphical representation of the steps or activities involved in a project. It helps to visually outline the sequence of steps, decision points, and interactions between various components or stakeholders in the project. Here is the flowchart for our project:

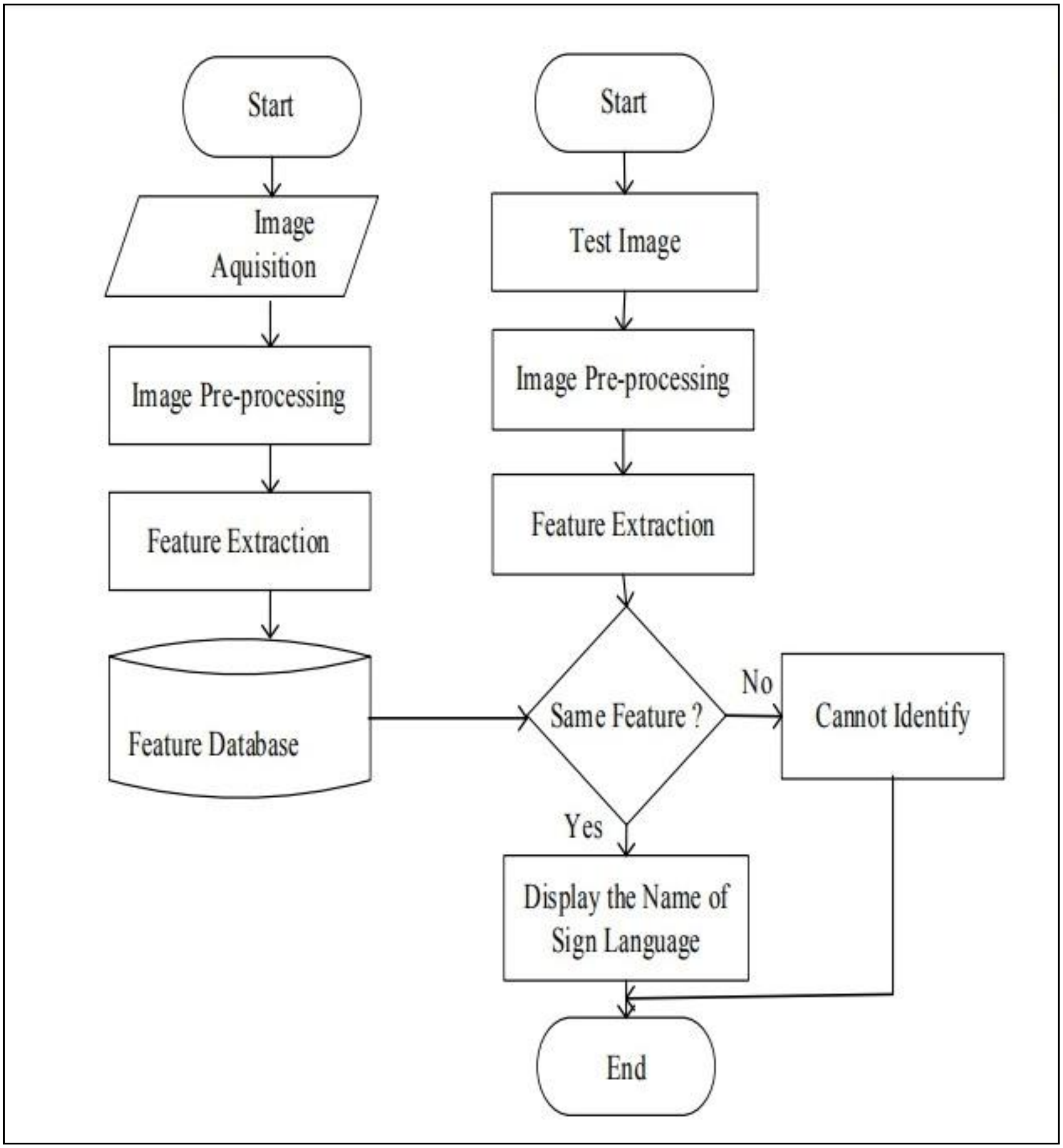


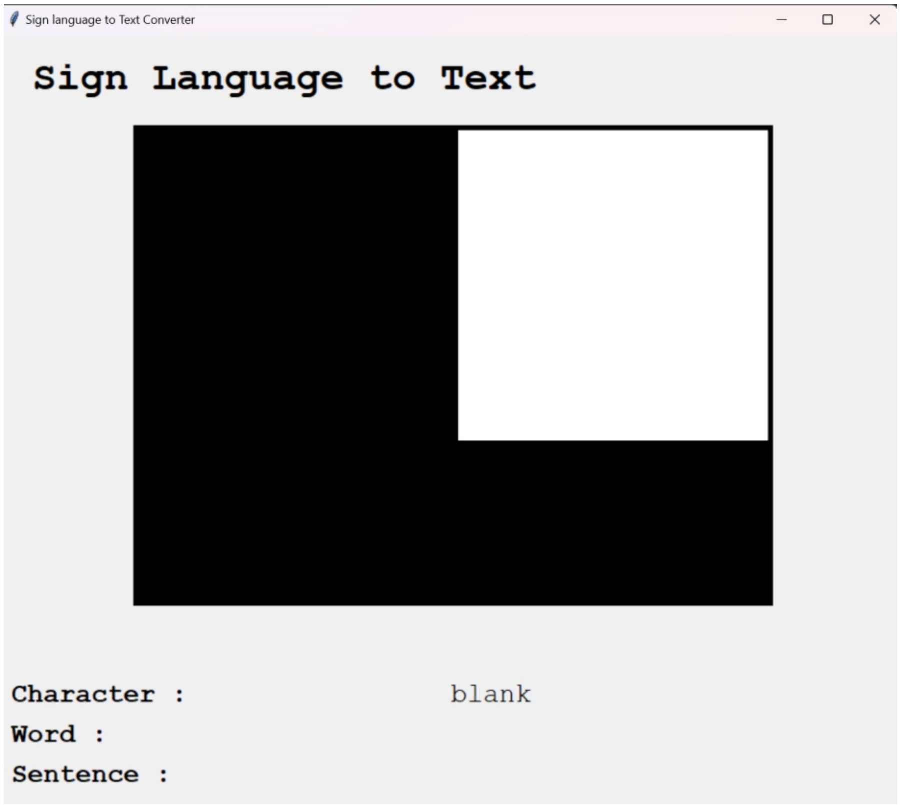
Fig. 3.7 Overall flowchart for sign language recognition system

3.4 User Interface Design

The user interface (UI) design is an important aspect of the system, as it determines how users interact with the system and how effectively they can communicate with it.

Here are some key considerations for UI design:

- **Accessibility:** The UI should be designed to be accessible for users with hearing and/or speech impairments. This could involve providing alternative means of communication, such as text-based chat or visual aids, in addition to the sign language recognition system.
- **User Experience:** The UI should be designed to provide a positive user experience, with clear and intuitive navigation and feedback mechanisms. Users should be able to easily understand how to use the system and receive feedback on their interactions.
- **Sign Language Display:** The UI should include a display area for the sign language gestures, so that users can see their own gestures and those of the system. This could involve using video or animation to display the gestures.
- **Gesture Input:** The UI should provide a means for users to input their sign language gestures. This could involve using a camera to capture the user's gestures, or providing a touchscreen interface for users to draw the gestures.



Sign Language to Text

Character : blank

Word :

Sentence :

Fig. 3.8 User Interface Design

3.5 Methodology

- **Data Acquisition:**

  Data acquisition involves capturing and collecting sign language gestures for use in training and testing the system. This is a critical step in developing an accurate and effective recognition system. There are various methods of data acquisition in sign language recognition, including Motion capture systems and Gesture Gloves etc.

- **Data pre-processing and Feature extraction of collected data:**

  Data preprocessing refers to the process of cleaning and preparing the sign language gesture data for use in the recognition system. The goal of data preprocessing is to remove any noise or unwanted variation in the data and to normalize the data so that it can be used effectively in the recognition algorithms. Some common techniques used in data preprocessing include filtering etc.

- **Feature representation from extracted and pre-processed data:**

  Feature extraction refers to the process of identifying the most important characteristics or features of the sign language gestures and representing them in a form that can be used by the recognition algorithms. Some common techniques used in feature extraction for include Motion-based features and Appearance-based features etc.

  Feature representation refers to the process of converting the sign language gesture data into a format that can be used by the recognition algorithms. This involves extracting the most relevant and informative features of the gesture and representing them in a compact and efficient form.

- **Gesture Classification:**

  Gesture classification refers to the process of identifying the sign language gesture based on the feature representation of the gesture. This involves using machine learning algorithms to classify the gesture into one of several predefined categories. Some common techniques used in gesture classification include Convolutional Neural Networks (CNN) etc.

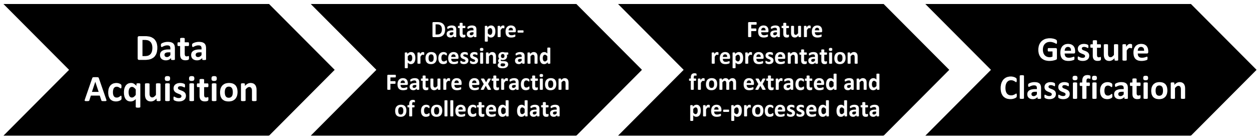

Data Acquisition
Data pre-processing and Feature extraction of collected data
Feature representation from extracted and pre-processed data
Gesture Classification

Fig. 3.9 Methodology of Work

4. Implementation, Testing, and Maintenance

4.1 Introduction to Languages, IDE's, Tools and Technologies used for Implementation

- **Python Language:** Python is a high-level, interpreted programming language known for its simplicity, readability, and versatility. It was created by Guido van Rossum and first released in 1991. Python's design philosophy emphasizes code readability and a clean, straightforward syntax, which makes it an excellent choice for beginners as well as experienced programmers. Python supports multiple programming paradigms, including procedural, object-oriented, and functional programming.

  One of the key features of Python is its extensive standard library, which provides a wide range of pre-built modules and functions that simplify and accelerate development. Additionally, Python has a large and active community that contributes to the creation of numerous third-party libraries and frameworks, expanding its capabilities in various domains, such as web development, data analysis, machine learning, and scientific computing.

- **Visual Studio Code:** Visual Studio Code, commonly referred to as VS Code, is a free and highly popular source code editor developed by Microsoft. It is designed to be lightweight, extensible, and customizable, making it a preferred choice among developers across different programming languages and platforms.

  Here are some key features and aspects of Visual Studio Code:

  > **Cross-platform compatibility:** Visual Studio Code is available for Windows, macOS, and Linux, ensuring that developers can use it on their preferred operating system.
  >
  > **User-friendly interface:** VS Code provides a clean and intuitive interface, which includes a sidebar for file navigation, a central editor area, and various panels for

tasks, extensions, and terminal integration. The interface is highly customizable, allowing users to personalize their coding environment according to their preferences.

- **Tools and Technologies used for Implementation:**

  Web Cameras: Web cameras are often used to capture the 3D movements and depth information of sign language gestures. These cameras provide depth maps and skeletal tracking, enabling precise tracking of hand and body movements.

  Machine Learning: Machine learning algorithms play a crucial role in sign language recognition. Techniques like convolutional neural networks (CNNs) are commonly employed to train models that can classify and recognize sign language gestures.

  Data Annotation Tools: Sign language recognition projects often require large datasets of annotated sign language gestures for training and evaluation. Data annotation tools, such as labeling or VGG Image Annotator (VIA), are used to label and annotate the captured video or image data with corresponding sign language gestures.

  OpenCV: OpenCV (Open Source Computer Vision Library) is a widely used open-source computer vision library. It provides various functions and algorithms for image and video processing, including frame extraction, feature extraction, and gesture tracking. OpenCV is often utilized in sign language recognition projects for pre-processing and analyzing video or image data.

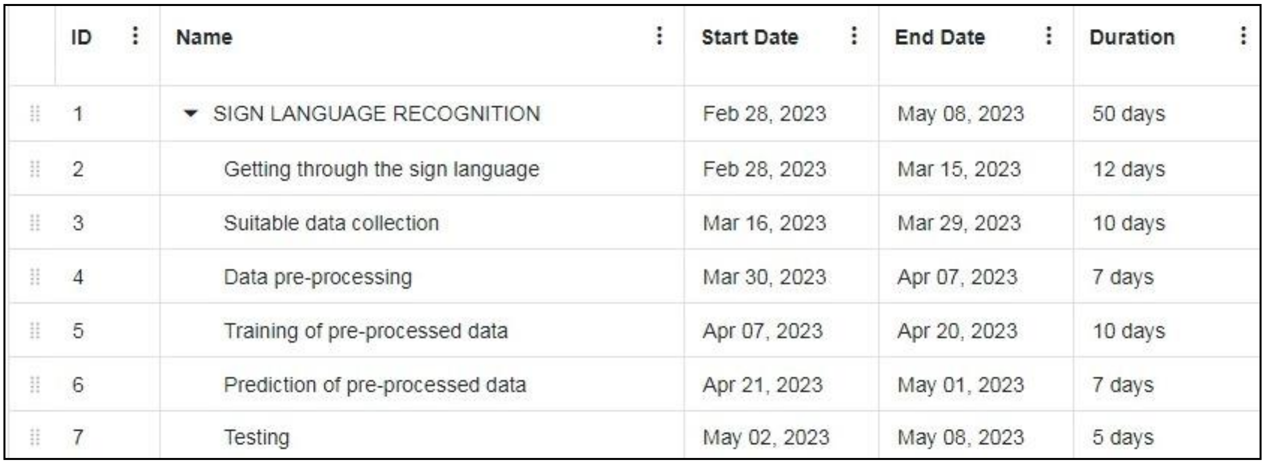
  Python Programming Language: Python is a popular programming language for machine learning and computer vision tasks. Its extensive libraries and frameworks, such as TensorFlow, Keras provide powerful tools for implementing machine learning models and handling data in sign language recognition projects.

4.2 Coding standards of Python Language used

- **Indentation and Line Length:** Use four spaces for indentation, as recommended by PEP 8. Avoid tabs or mixing tabs and spaces.
- **Naming Conventions:** Follow consistent naming conventions to make your code more readable. Use lowercase letters with words separated by underscores for variable and function names (e.g., gesture_recognition). For class names, use the CapWords convention (e.g., GestureRecognition).
- **Imports:** Place imports at the top of the file and organize them in the following order: standard library imports, third-party library imports, and local module imports. Separate each group of imports with a blank line. Avoid using wildcard imports (from module import *) and import only the necessary components.
- **Whitespace:** Use whitespace to enhance code readability. Separate functions and classes with two blank lines. Use a single space around operators and after commas in function arguments.
- **Comments:** Include clear and concise comments to explain the purpose and functionality of your code. Comments should be written in complete sentences and avoid stating the obvious. Commented code should be kept to a minimum, and instead, consider removing or refactoring unused code.

4.3 Gantt Chart

A Gantt chart is a popular project management tool used to represent a project schedule in a visual way. A Gantt chart consists of a horizontal timeline and a set of bars that represent different tasks or activities in the project. Each bar is positioned according to the start and end dates of the task, and its length corresponds to the duration of the task. Dependencies between tasks can also be represented using arrows, connecting one task to another.



ID	Name	Start Date	End Date	Duration
1	SIGN LANGUAGE RECOGNITION	Feb 28, 2023	May 08, 2023	50 days
2	Getting through the sign language	Feb 28, 2023	Mar 15, 2023	12 days
3	Suitable data collection	Mar 16, 2023	Mar 29, 2023	10 days
4	Data pre-processing	Mar 30, 2023	Apr 07, 2023	7 days
5	Training of pre-processed data	Apr 07, 2023	Apr 20, 2023	10 days
6	Prediction of pre-processed data	Apr 21, 2023	May 01, 2023	7 days
7	Testing	May 02, 2023	May 08, 2023	5 days

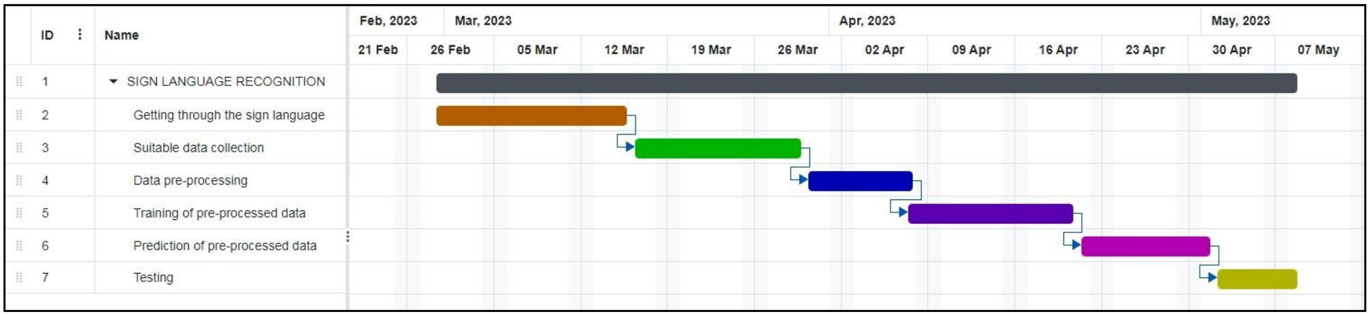


Fig. 4.1 Gantt chart

4.4 Testing Techniques and Test Plans

Here are some common testing techniques and test plans used in sign language projects:

- **Unit Testing:** Unit testing focuses on testing individual components, such as functions, methods, or classes, in isolation. In a sign language recognition system, we] can write unit tests to verify the correctness of individual modules or algorithms involved in gesture recognition. These tests can validate specific functionalities, handle edge cases, and ensure the desired outputs are produced for a given input.

- **Integration Testing:** Integration testing evaluates the interaction and integration between different modules or components of the sign language recognition system. It aims to ensure that the components work harmoniously and produce the expected results when combined. For example, integration testing can involve testing the interaction between the gesture tracking module and the machine learning module to verify that the recognized gestures align with the input data.
- **Functional Testing:** Functional testing focuses on testing the system's functionalities and features as a whole. It verifies whether the sign language recognition system meets the specified functional requirements. This type of testing can involve running the system with different sets of input sign language gestures and verifying that the system correctly recognizes and interprets them.
- **Performance Testing:** Performance testing assesses the system's performance in terms of response time, resource usage, and scalability. In a sign language recognition project, performance testing can involve measuring the time taken to recognize gestures or tracking system resource consumption during real-time gesture recognition. This testing helps ensure that the system performs efficiently and can handle the expected load.

5. Results and Discussions

5.1 User Interface Representation

- **About function:** It is a user defined function which contains details about students and their respective guide.
- **Character:** It tells about current predicting character.
- **Words:** It collects the total predicted characters in sequence.
- **Sentence:** It collects the total words and forms the sentence.
- **Image:** It is the area of screen where real time picture is captured.
- **Region of interest (ROI):** It is the part of image where gaussian filters are applied to convert the sign language into words.

5.2 Brief Description of Various Modules of the system

- **NumPy:** NumPy is a Python library used for numerical computing. It provides a wide range of mathematical functions and tools for working with arrays and matrices.
- **OpenCV:** OpenCV (Open Source Computer Vision Library) is a popular computer vision library that provides a range of algorithms and techniques for image processing, including image filtering, edge detection, feature extraction, and object detection. In sign language recognition projects, OpenCV is often used for hand detection and tracking, as well as feature extraction from hand images.
- **TensorFlow:** TensorFlow is an open-source machine learning library developed by Google. It is commonly used for building and training deep neural networks for a variety of applications, including computer vision and natural language processing. In sign language

recognition projects, TensorFlow is often used for building and training machine learning models, including deep learning models such as convolutional neural networks (CNNs).

- **Keras:** Keras is a high-level neural networks API written in Python that runs on top of TensorFlow. It is designed to simplify the process of building, training, and deploying deep learning models. In sign language recognition projects, Keras is often used for creating and training deep learning models, as well as evaluating their performance.
- **PIL:** The PIL (Python Imaging Library) module provides basic image processing capabilities that can be used to preprocess images before feeding them into machine learning models.
- **OS module:** The OS module provides a way to interact with the operating system, including functions for file I/O and directory manipulation. This can be useful for loading and saving data to and from disk.
- **Sys module:** The sys module provides access to some system-specific parameters and functions. It can be used to control the Python interpreter and obtain information about the Python environment.

5.3 Snapshots of system with brief detail of each

Data Collection

Data collection is a critical part of our project as it allows our machine learning algorithm to learn and recognize different hand gestures with greater accuracy. The more data we collect, the better our algorithm will be at recognizing different hand gestures and converting them into language.

When collecting data for our project, it is important to ensure that we capture a wide variety of hand gestures and in different lighting conditions. This will help to make our algorithm more robust and adaptable to different scenarios.

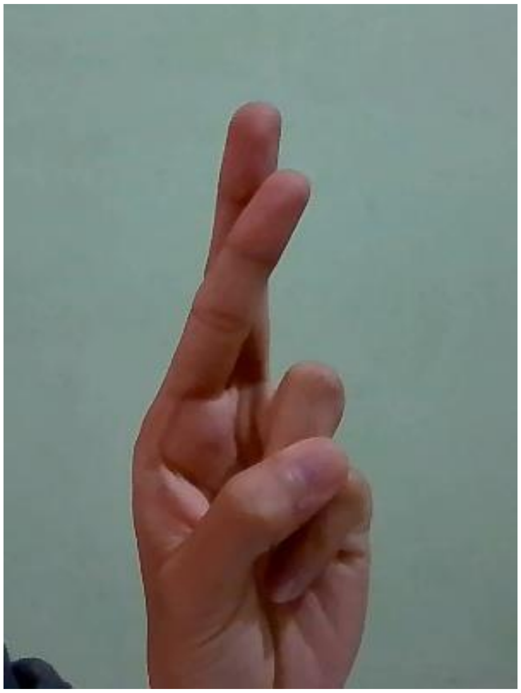


Fig. 5.1 Alphabet R

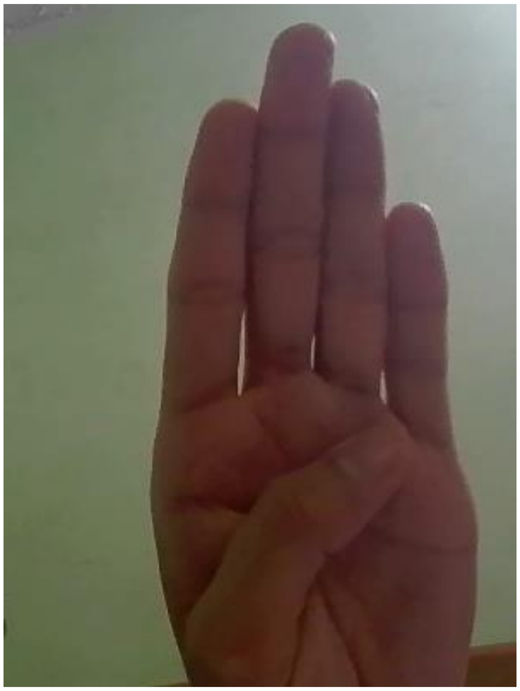


Fig. 5.2 Alphabet B

Data Pre-Processing

Data pre-processing is a crucial step in our hand gesture recognition project as it can help to improve the accuracy and performance of our machine learning model. Here are some ways in which the preprocessing techniques we mentioned - Gaussian blur and adaptive thresholding - can be helpful:

- **Image Details:** By applying Gaussian blur, we can reduce the details in the image, making it easier for our model to identify the important features of the hand gesture.
- **Feature extraction:** Adaptive thresholding can help to extract the key features of the hand gesture, such as the outline or contour. This can help our model to identify the gesture more accurately.

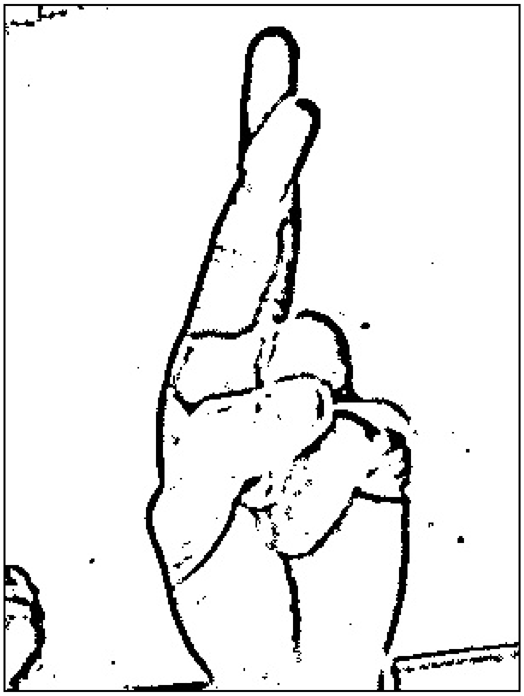


Fig. 5.3 Alphabet R

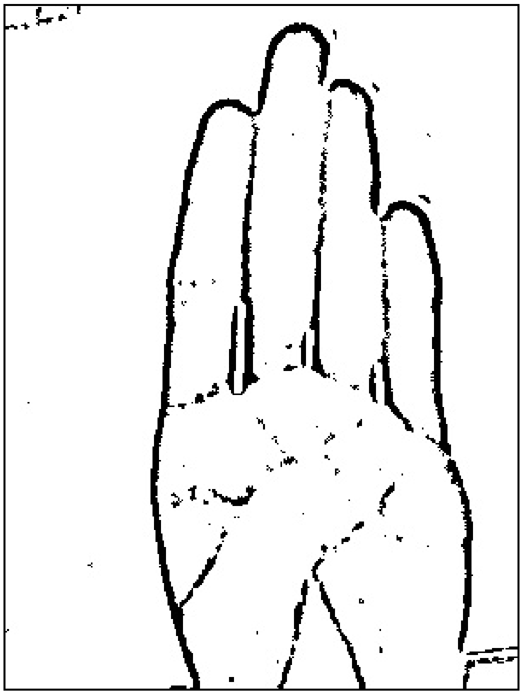
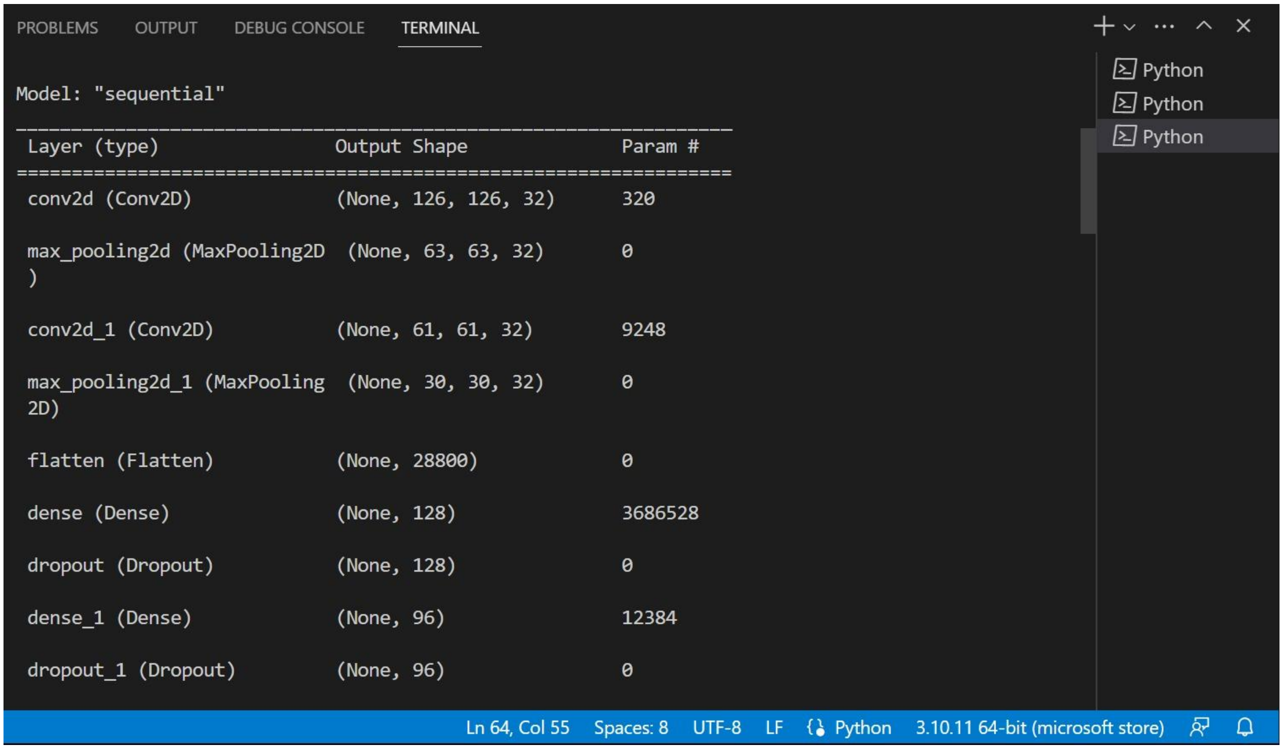
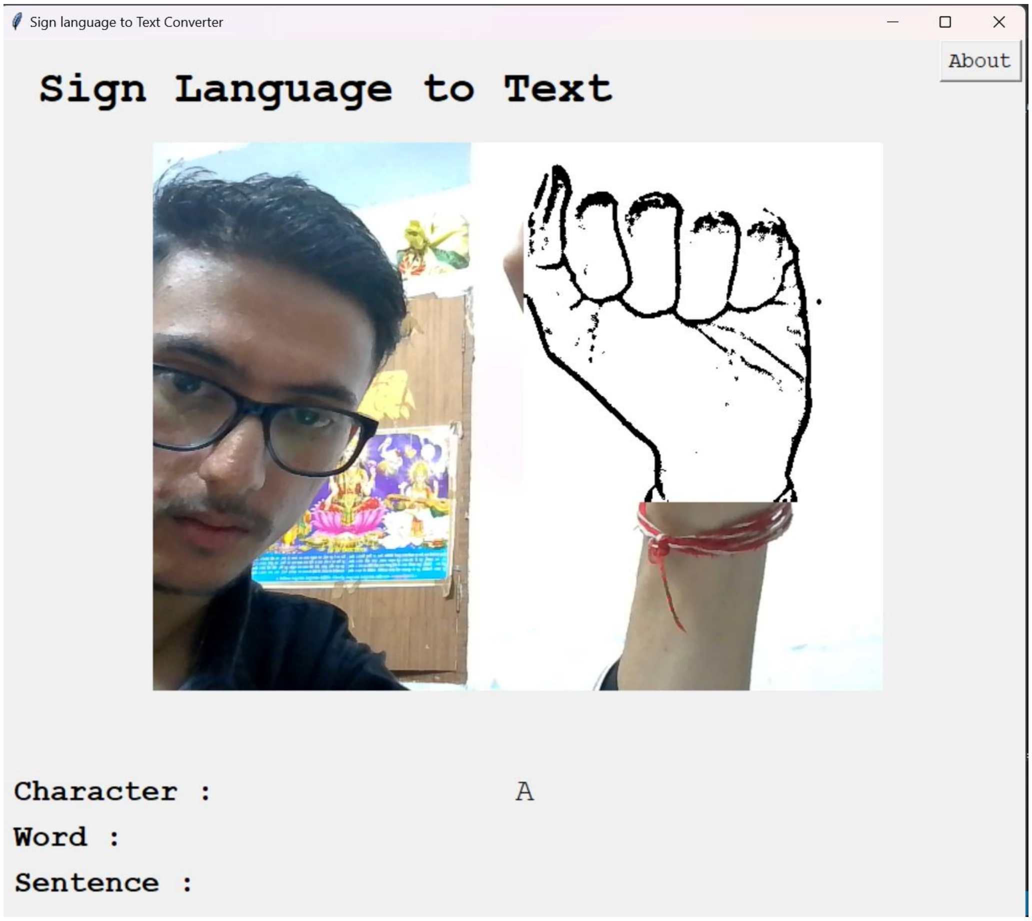


Fig. 5.4 Alphabet B



```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 126, 126, 32)      320

 max_pooling2d (MaxPooling2D  (None, 63, 63, 32)       0
 )

 conv2d_1 (Conv2D)           (None, 61, 61, 32)        9248

 max_pooling2d_1 (MaxPooling  (None, 30, 30, 32)       0
 2D)

 flatten (Flatten)           (None, 28800)             0

 dense (Dense)               (None, 128)               3686528

 dropout (Dropout)           (None, 128)               0

 dense_1 (Dense)             (None, 96)                12384

 dropout_1 (Dropout)         (None, 96)                0
```

Fig. 5.5 Output of trained data



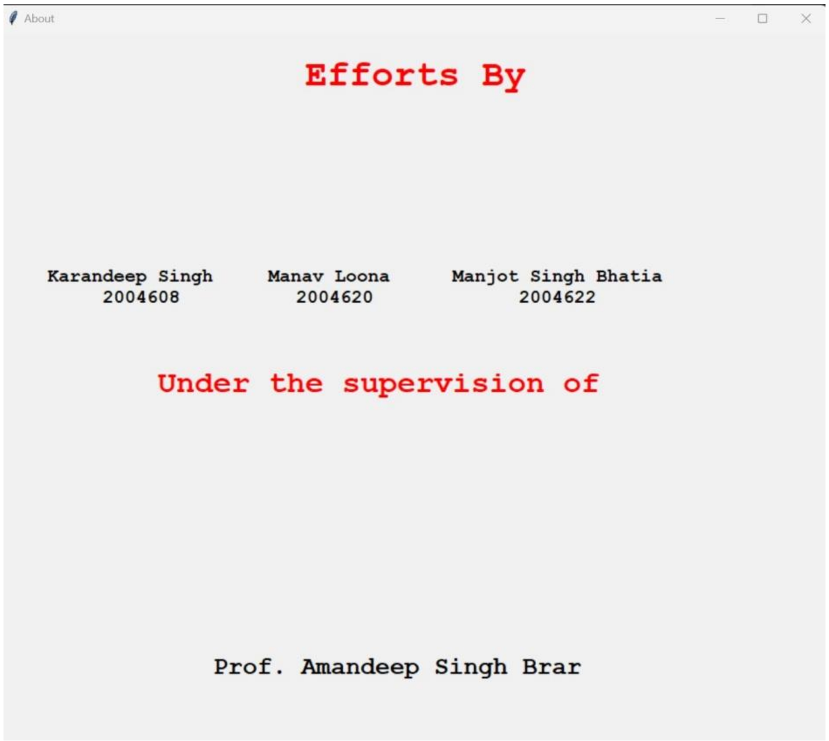
Sign Language to Text

Character : A

Word :

Sentence :

Fig 5.6 Conversion of Sign Language to Text



Efforts By

Karandeep Singh
2004608

Manav Loona
2004620

Manjot Singh Bhatia
2004622

Under the supervision of

Prof. Amandeep Singh Brar

Fig 5.7 Window after clicking About button

6. Conclusion and Future Scope

We have made a implementation of sign language recognition system using machine learning algorithm or model. This project has the potential to greatly benefit the deaf and hard of hearing community. By using advanced technology and machine learning algorithms, we were able to develop a system that accurately recognizes sign language gestures in real time. This technology has the potential to break down communication barriers and allow for more inclusivity and accessibility in various settings, such as education, healthcare, and daily communication.

Accuracy was around 75 - 80 % . This was calculated by taking letters manually and counting how many times system has detected it properly.

In future, this project is broad and varied, with many areas for potential research and development. One of the key areas for future work is to expand the vocabulary of the system to include more sign languages and signs. This could involve collecting more sign language data and creating more sophisticated machine learning algorithms to recognize the signs. Additionally, improving the accuracy of the system by exploring new techniques for feature extraction, data preprocessing, and machine learning algorithms is another important area of research. The possibility of real-time sign language recognition and integrating gesture recognition into the system is also an area of interest.

7. References

[1]Buehler, T., Everingham, M., & Zisserman, A. (2009). Recognition and synthesis of hand-squeezes in sign language videos. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (pp. 859-866).

[2] Yao, T., Li, Y., Huang, Y., Li, Z., & Jiang, T. (2018). Sign language recognition using 3D convolutional neural networks. IEEE Access, 6, 14429-14438.

[3] Garcia-Hernandez, N., Camarena-Ibarrola, A., & Carrasco-Ochoa, J. A. (2017). Sign language recognition based on fuzzy inference systems and image processing. IEEE Transactions on Human-Machine Systems, 47(2), 174-183.

[4] Zhou, L., & Wong, K. Y. K. (2020). Sign language recognition using deep learning: A survey. IEEE Transactions on Neural Systems and Rehabilitation Engineering, 28(1), 4-16.

[5] Lee, D., & Kim, K. (2019). Sign language recognition based on deep learning with adaptive hand gesture segmentation. IEEE Access, 7, 12781-12790.