

# **CHAPTER 1: INTRODUCTION**

## CHAPTER 1

### INTRODUCTION

#### 1.1 Introduction:

The aim of this project is to improve the communication with the people who has hearing difficulties and using any sign language to express themselves. At the first sight, as an idea, how difficult could make a sign languages converter. It is said that Sign language is the mother language of deaf people. This includes the combination of hand movements, arms or body and facial expressions. There are 135 types of sign languages all over the world. Some of them are American Sign Language, Indian Sign Language, British Sign Language, many more. We are using Indian Sign Language in this project. This system allows the deaf community to enjoy all sorts of things that normal people do from daily interaction to accessing the information. This application takes speech as input, converts it into text and then displays the Indian Sign Language images.

- **Speech Input:** The system captures speech from the user as an input.
- **Speech Recognition:** The input speech is processed using the Google Speech API to convert it into text.
- **Text Preprocessing:** The converted text undergoes preprocessing using Natural Language Processing (NLP) techniques.
- **Dictionary-Based Translation:** The processed text is translated into Indian Sign Language using a predefined dictionary containing images and GIFs representing various words and phrases.

#### 1.2 Problem Statement:

Sign language is communication language used by the deaf peoples using face, hands or eyes while using vocal tract. Sign language recognizer tool is used for recognizing sign language of deaf and dumb people. Gesture recognition is an important topic due to the fact that segmenting a foreground object from a cluttered background is a challenging problem.

There is a difference when human looks at an image and a computer looking at an image. For Humans it is easier to find out what is in an image but not for a computer. It is because of this, computer vision problems remain a challenge.

### **1.3 Existing System:**

There have been many projects done on the sign languages that convert sign language as input to text or audio as output. But audio to sign language conversion systems have been rarely developed. It is useful to both normal and deaf people.

- Lack of efficient translation from speech to sign language.
- Limited accuracy in recognizing sign language gestures.
- Inability to handle complex sentences and variations in sign language expressions.

### **1.4 Proposed System:**

In this project we introduce new technology that is audio to sign language translator using python.

- A. In this it takes audio as input, search that recording using Google api, display the text on screen and finally it gives sign code of given input using ISL (Indian Sign Language) generator. All the words in the sentence are then checked against the words in the dictionary containing images and GIFs representing the words. If the words are not found, its corresponding synonym is replaced. Set of gestures are predefined in the system.
- B. We are going to consider the Indian sign language image as input and convert that into text/speech.

### **Key Features:**

#### **1.4.1 Audio to sign language:**

- a) Audio to Text Conversion:
  - Audio input is taken using python PyAudio module.
  - Conversion of audio to text using microphone

- Dependency parser is used for analysing grammar of the sentence and obtaining relationship between words.

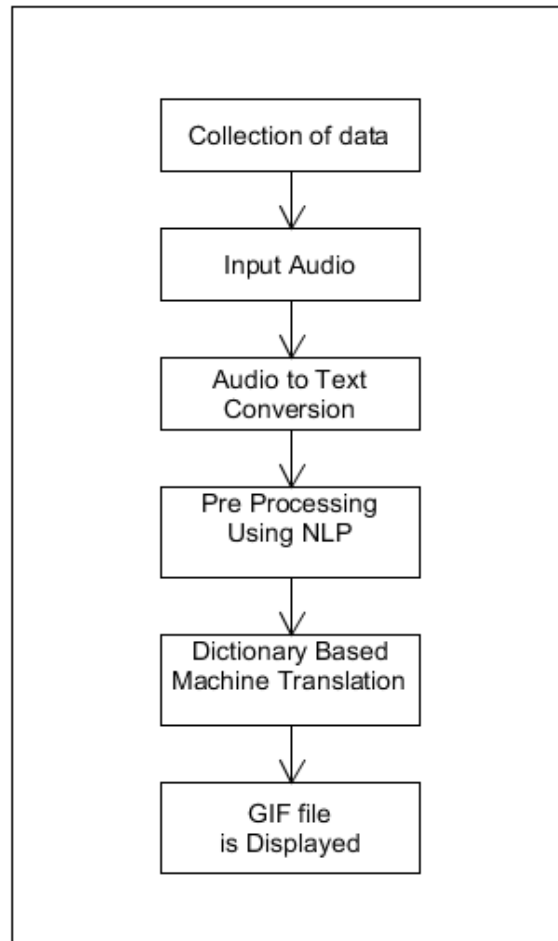
b) Text to Sign Language:

- Speech recognition using Google Speech API.
- Text Pre-processing using NLP.
- Dictionary based Machine Translation.
- ISL Generator: ISL of input sentence using ISL grammar rules.
- Generation of Sign language with GIF file.

#### **1.4.2 Image to Text/Speech:**

- Collection of Indian Sign Language datasets
- Applying Pre-Processing Techniques (resize, noise remove, reconversion) for the collected datasets.
- Splitting the data into train and test
- Applying CNN Algorithms for training datasets
- Build Model
- Create Web App for the build model

## 1.5 Architecture:



*Figure 1.1 Block diagram*

## 1.6 Algorithm Used:

- **NLP**

Natural language processing (NLP) is the ability of a computer program to understand human language as it is spoken and written -- referred to as natural language. It is a component of artificial intelligence.

NLP enables computers to understand natural language as humans do. Whether the language is spoken or written, natural language processing uses artificial intelligence to take real-world input, process it, and make sense of it in a way a computer can understand. Just as

humans have different sensors -- such as ears to hear and eyes to see -- computers have programs to read and microphones to collect audio. And just as humans have a brain to process that input, computers have a program to process their respective inputs. At some point in processing, the input is converted to code that the computer can understand.

### **1.7 Objectives:**

- The main objective of this project is to build an automated system.
- The proposed system has learning material for deaf children which has impact on them in learning language basics.
- This application convert image to text which helps children to learn
- The skill will go a long way in their life in the context of school and afterwards. This is the motivation behind taking up this research.

## **CHAPTER 2:**

# **LITERATURE SURVEY**

## CHAPTER 2

### LITERATURE SURVEY

#### 2.1 Introduction:

The study of sign language translation systems has evolved significantly with advancements in artificial intelligence, image processing, and deep learning techniques. Various research papers have been analyzed to understand existing solutions, challenges, and potential improvements. This chapter categorizes these studies into different sections, providing a structured review of the relevant literature

#### 2.2 Sign Language Recognition Techniques:

1. **Title:** Real Time Sign Language Recognition Using Deep Learning

- **Author:** Sanket Bankar, Tushar Kadam, Vedant Korhale, Mrs. A. A. Kulkarni.
- **Year of Publication:** Apr 2022
- Sign language is an extremely important communication tool for many deaf and mute people. So, we proposed a model to recognize sign gestures using YOLOv5 (You only look once version 5). This model can detect sign gestures in complex environment also. For this model we got the accuracy of 88.4% with precision of 76.6% and recall of 81.2%. The proposed model has evaluated on a labeled dataset Roboflow. Additionally, we added some images for training and testing to get better accuracy. We compared our model with CNN (convolutional neural network) where we got accuracy of 52.98%. We checked this model for real time detection also and got the accurate results [1].

2. **Title:** American Sign Language Recognition Using Computer Vision

- **Author:** Nihar Joshi, Ryan Gudal, Tianyu Jiang, Samantha Clark.
- **Year of Publication:** 2021



- American Sign Language (ASL) is the primary form of communication for 1 million individuals in the Deaf community, hard-of-hearing community, as well as family or friends of people in the Deaf community. However, for the remaining U.S. population, very few people know ASL. As a result, there is a significant communication barrier between many ASL signers and the rest of Americans. Communication with signers usually requires Deaf Interpreters, who are not always available and are in high demand – the number of interpreters needed is expected to increase by 19% over 10 years [2].

## 2.3 Speech-to-Sign Language Conversion:

### 3. **Title:** Translating Speech to Indian Sign Language Using Natural Language Processing

- **Author:** Purushottam Sharma, Devesh Tulsian, Chaman Verma, Pratibha Sharma, Nancy Nancy.
- **Year of Publication:** 2022
- Language plays a vital role in the communication of ideas, thoughts, and information to others. Hearing-impaired people also understand our thoughts using a language known as sign language. Every country has a different sign language which is based on their native language. In our research paper, our major focus is on Indian Sign Language, which is mostly used by hearing- and speaking-impaired communities in India. While communicating our thoughts and views with others, one of the most essential factors is listening. What if the other party is not able to hear or grasp what you are talking about? This situation is faced by nearly every hearing-impaired person in our society. This led to the idea of introducing an audio to Indian Sign Language translation system which can erase this gap in communication between hearing-impaired people and society. The system accepts audio and text as input and matches it with the videos present in the database created by the authors. If matched, it shows corresponding sign movements based on the grammar rules of Indian Sign Language as output; if not, it then goes through

the processes of tokenization and lemmatization. The heart of the system is natural language processing which equips the system with tokenization, parsing, lemmatization, and part-of-speech tagging [3].

4. **Title:** Real-Time Speech to Sign Language Translation Using Machine and Deep Learning

- **Author:** Deepak Rai, Niharika Rana, Naman Kotak, Manya Sharma.
- **Year of Publication:** 2024
- This approach critically analyzed the current technology for speech-to-sign language translation. To extract key features from the input speech signal, audio processing techniques such as Mel-frequency cepstral coefficients (MFCCs), Fast Fourier transform (FFT), and Discrete Cosine Transform (DCT) are used first. Combining the advantages of both architectures—convolutional neural networks (CNNs) and recurrent neural networks (RNNs)—creates a potent feature extraction method that can effectively extract features with both temporal and spatial patterns. Following their retrieval, the attributes are input into a Text-to-Sign (TTS) module, which uses a Convolutional Long Short-Term Memory (ConvLSTM) network to generate the proper sign language sequence. The created sequence of sign language is animated using motion graphics (MG). This technology uses motion capture data that has already been recorded to create realistic and expressive sign language motions. Motion graphics offer an appropriate choice for real-time translation applications that will be using Long Short-Term Memory Long time (LSTM) by balancing usability with the size of the necessary motion capture database [4].

## 2.4 Gesture Recognition for Sign Language:

5. **Title:** Vision-based hand gesture recognition using deep learning for the interpretation of sign language

- **Author:** Sakshi Sharma, Sukhwinder Singh

- **Year of Publication:** 2021
- Hand gestures have been the key component of communication since the beginning of an era. The hand gestures are the foundation of sign language, which is a visual form of communication. In this paper, a deep learning based convolutional neural network (CNN) model is specifically designed for the recognition of gesture-based sign language. This model has a compact representation that achieves better classification accuracy with a fewer number of model parameters over the other existing architectures of CNN. In order to evaluate the efficacy of this model, VGG-11 and VGG-16 have also been trained and tested in this work. To evaluate the performance, 2 datasets have been considered. First, in this work, a large collection of Indian sign language (ISL) gestures consisting of 2150 images is collected using RGB camera, and second, a publicly available American sign language (ASL) dataset is used. The highest accuracy of 99.96% and 100% is obtained by the proposed model for ISL and ASL datasets respectively [5].

6. **Title:** Real-Time Hand Gesture Recognition Using Fine-Tuned Convolutional Neural Network

- **Author:** Jaya Prakash Sahoo, Allam Jaya Prakash, Paweł Pławiak, Saunak Samantray.
- **Year of Publication:** 2022
- Hand gesture recognition is one of the most effective modes of interaction between humans and computers due to being highly flexible and user-friendly. A real-time hand gesture recognition system should aim to develop a user-independent interface with high recognition performance. Nowadays, convolutional neural networks (CNNs) show high recognition rates in image classification problems. Due to the unavailability of large labeled image samples in static hand gesture images, it is a challenging task to train deep CNN networks such as AlexNet, VGG-16 and ResNet from scratch. Therefore, inspired by CNN performance, an end-to-end fine-tuning method of a pre-trained CNN model with score-level fusion technique is proposed

here to recognize hand gestures in a dataset with a low number of gesture images. The effectiveness of the proposed technique is evaluated using leave-one-subject-out cross-validation (LOO CV) and regular CV tests on two benchmark datasets. A real-time American sign language (ASL) recognition system is developed and tested using the proposed technique[6].

## 2.5 Speech-to-Sign Language Conversion:

### 7. **Title:** A Two-Way Integrated Communication System for the Deaf and Mute

- **Author:** Godson Thomas, Gokul Rejithkumar, P. Sreevidya, Beenu Rijju.
- **Year of Publication:** 2022
- non-disabled person finds it onerous to learn sign language to communicate with the deaf-mutes. This paper discusses a two-way communication system that bridges this communication gap. The proposed system is a portable standalone embedded system that acts as a real-time sign language translator between the deaf-mutes and non-mutes. A fingerspelling data set of Indian sign language that includes some commonly used words were created for training and validating the model. The system uses computer vision techniques and customized speech translator APIs, enabling two-way communication between the deaf-mutes and non-mutes [7].

### 8. **Title:** A Study on Communication Platforms and Services Needed and Available for Mute and Deaf Community

- **Author:** Snehal Patil, Yash Shah, Payal Narkhede, Abhinav Thakare, Rahul Pitale
- **Year of Publication:** 2021
- According to the World Health Organization, Mute and deaf community are among the 5% of our world population that is approximately 430 million people. This number can increase to up to 2.5 billion till 2050 i.e. one among 10 people would suffer from some kind of hearing disorder. Often these differently abled people are neglected by society. These people face various

problems like lack of basic education, platforms to express themselves to people who don't understand sign language or able to earn jobs to fulfil their basic needs, etc. According WHO unaddressed issues faced by mute and deaf people were that, there are almost 32 million kids who suffer from hearing loss and these are from currently developing countries where there is limited budget to provide education, or any other facilities that could help these people to compete with world. Thus, in adults the unemployment rate is quite high. Further leading to social isolation, loneliness, depression. Thus, this survey majorly focuses on comparing various currently available communication platforms and services for these differently abled people and further what are the necessary services they need. This survey also compares the techniques to build platforms to bridge the gap between these people and people lacking the knowledge of sign language.

## **CHAPTER 3:**

# **SYSTEM DESIGN**

## CHAPTER 3

### SYSTEM DESIGN

#### 3.1 Introduction:

The Software Design Document serves as a blueprint for the development of an Android application, providing a comprehensive framework detailing how the application should be built. This document includes narrative and graphical representations of the software design, ensuring clarity and a structured development approach. It comprises various modeling techniques such as use case models, sequence diagrams, data flow diagrams, and other supporting requirement specifications that aid in effective software development and system understanding.

##### 3.1.1 Scope:

This software Design Document is for a base level system, which will work as a proof of concept for the use of building a system that provides a base level of functionality to show feasibility for large-scale production use. The software Design Document, the focus placed on generation of the documents and modification of the documents. The system will be used in conjunction with other pre-existing systems and will consist largely of a document interaction layer that abstracts document interactions and handling of the document objects. This Document provides the Design specifications of smart translation.

##### 3.1.2 System Overview:

The system is designed to provide smart translation services, utilizing advanced Natural Language Processing (NLP) techniques and machine learning algorithms. It ensures accurate, efficient, and context-aware translations. The system comprises various components, including:

- **User Interface (UI):** Provides an intuitive and interactive interface for users to input, view, and manage translations.

- **Translation Engine:** Utilizes AI-based models to process and generate translations with high accuracy.
- **Database Management:** Stores and retrieves documents, translation history, and user preferences.
- **APIs and Integration:** Allows integration with external applications and third-party translation services.
- **Security and Authentication:** Implements user authentication, access control, and encryption to protect sensitive data.

### **3.1.3 Design Constraints:**

- The system should support multi-language translations with high accuracy.
- It should provide real-time translation capabilities.
- The architecture should be scalable to support future enhancements.
- The UI should be responsive and accessible across different devices.
- Integration with third-party services should be seamless and secure.

## **3.2 Data Flow Diagram:**

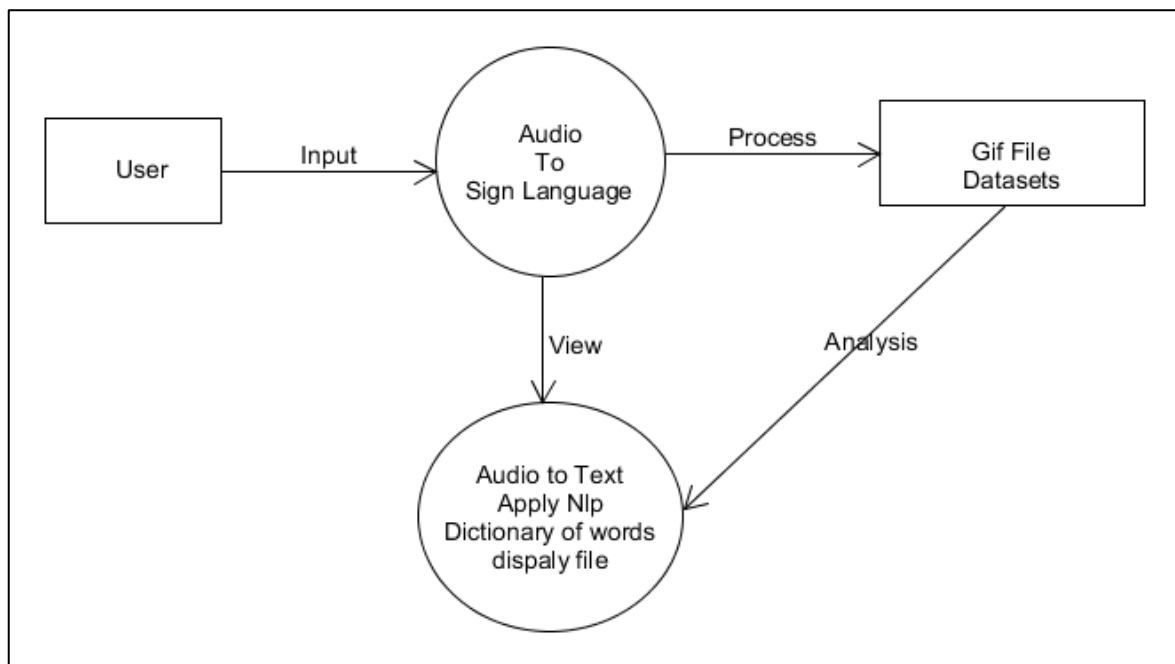
A Data Flow Diagram (DFD) is a graphical representation that depicts the flow of data within a system, illustrating how information moves from input to output through various processes. It uses standardized symbols such as rectangles (representing external entities), circles (depicting processes), arrows (indicating data flow), and open-ended rectangles (representing data stores). These symbols help in visualizing the movement, transformation, and storage of data in a structured manner.

DFDs are categorized into different levels based on their complexity. A Context Diagram (Level 0 DFD) provides a high-level overview of the system, showing external entities and their interactions with the system. Level 1 DFDs break down the major processes into sub-

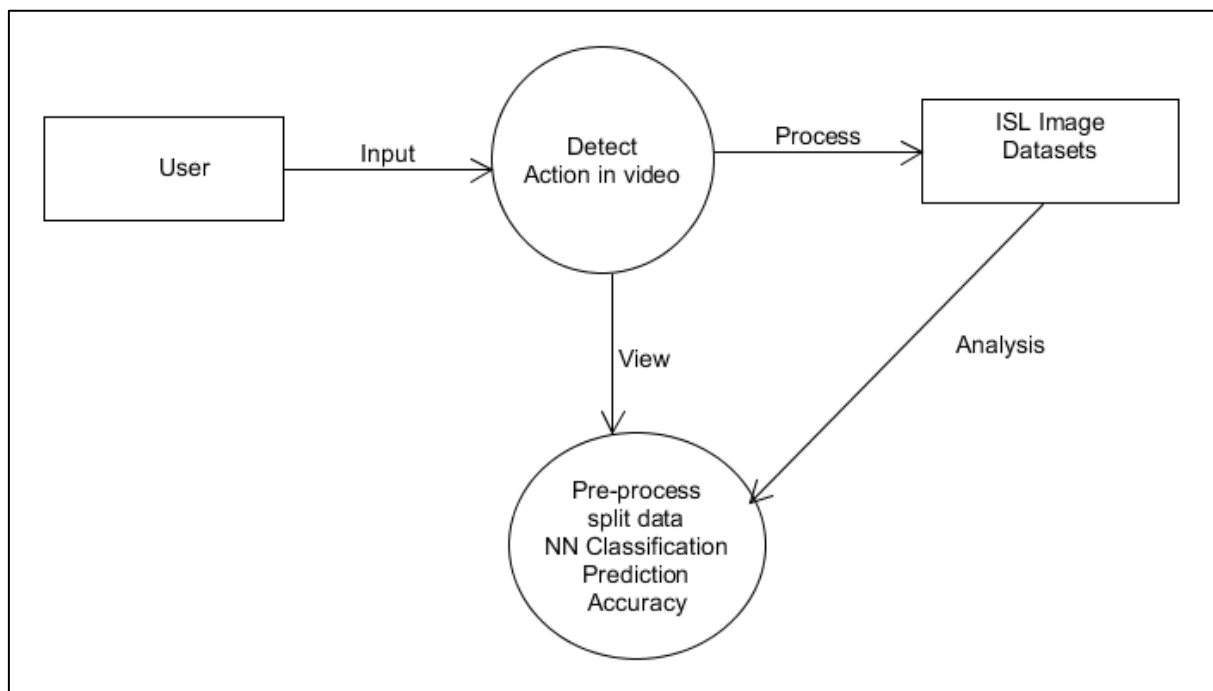


processes, giving a detailed view of data movement. Further decomposition leads to Level 2 and beyond, offering in-depth insight into the system's internal workings.

DFDs are widely used in system analysis, software development, and business process modeling, helping stakeholders understand workflows, identify inefficiencies, and optimize processes. They play a crucial role in requirement gathering, system design, and documentation, making them a fundamental tool for developers, analysts, and project managers. By providing a clear, visual representation, DFDs facilitate effective communication among technical teams and business stakeholders, ensuring better alignment in system development and process improvement



*Figure 3.1 Audio to Sign Language*



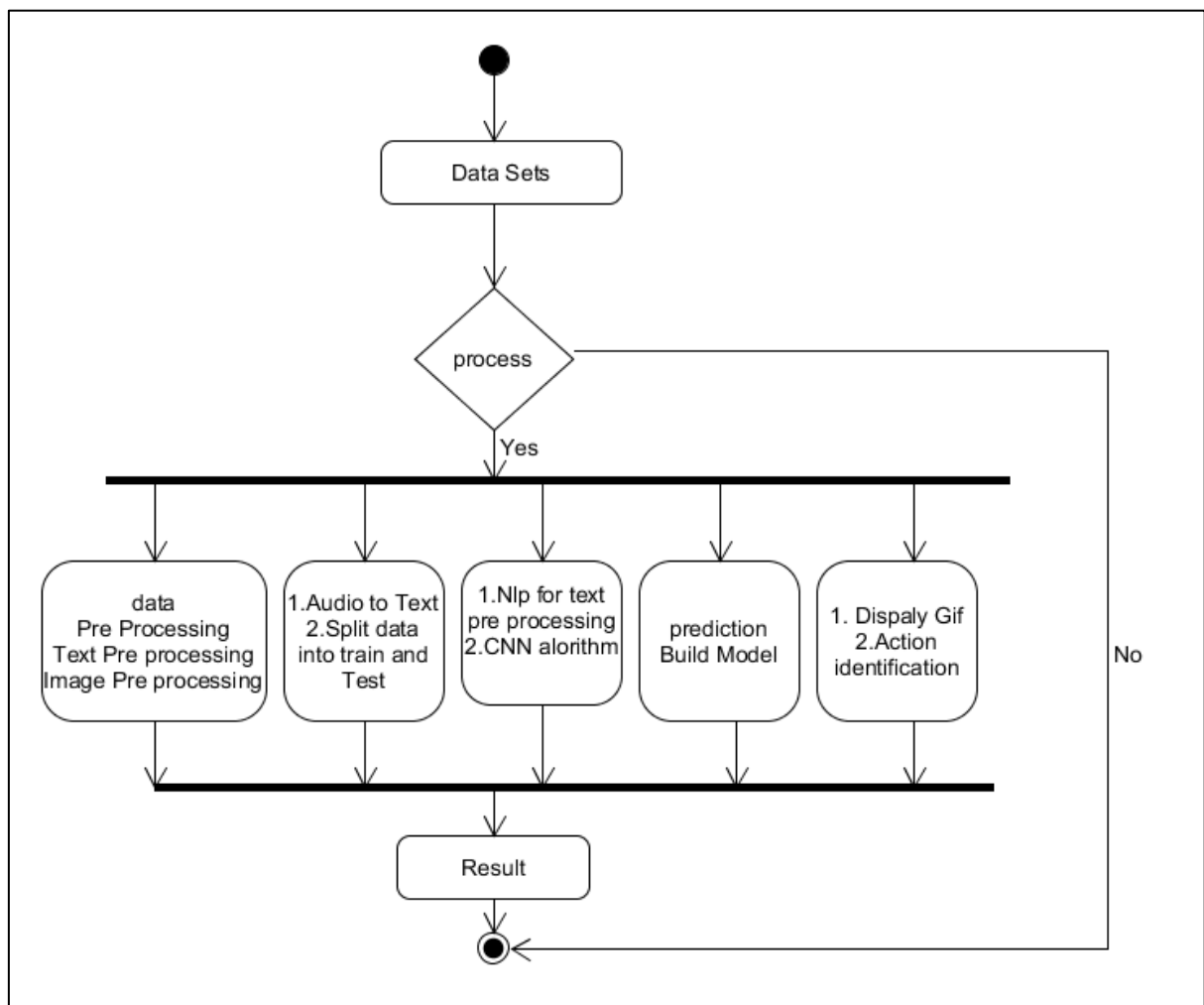
*Figure 3.2 Sign Language to Text*

### 3.3 Activity Diagram:

Activity Diagrams describe how activities are coordinated to provide a service which can be at different levels of abstraction. Typically, an event needs to be achieved by some operations, particularly where the operation is intended to achieve a number of different things that require coordination, or how the events in a single use case relate to one another, in particular, use cases where activities may overlap and require coordination. It is also suitable for modeling how a collection of use cases coordinate to represent business workflows.

- **Use in Software Development:** Activity diagrams help in modeling business processes, system workflows, and user interactions. They provide clarity in the execution flow of use cases, algorithms, and high-level processes.
- **Parallel Processing Representation:** They support concurrent processing, which helps visualize activities that can execute simultaneously using fork and join nodes.

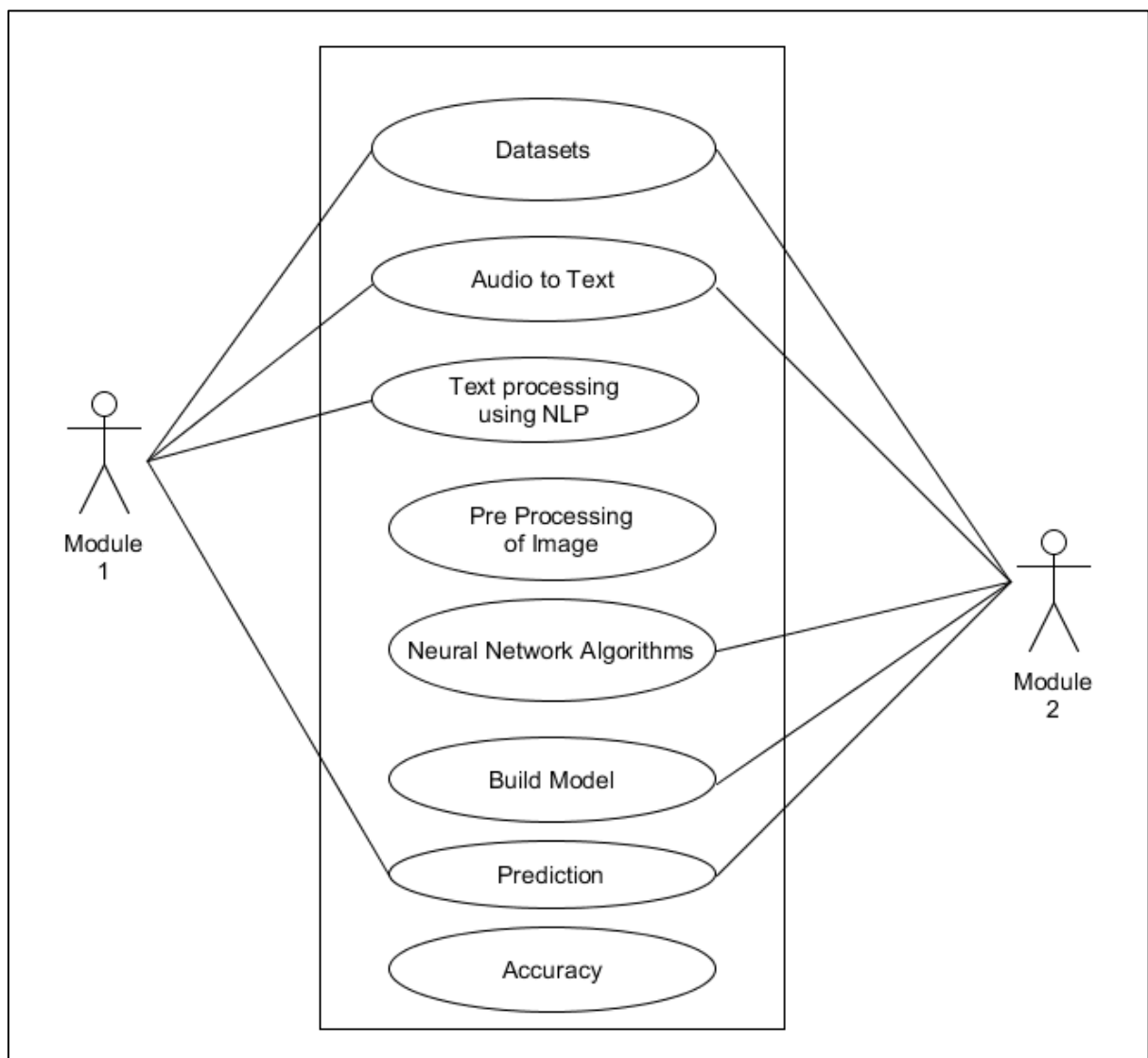
- Error Handling and Exception Flows: They can model alternative paths in case of failures, exceptions, or different decision outcomes.
- Documentation and Communication Tool: They act as a bridge between technical and non-technical stakeholders, improving understanding and documentation of system behaviors.
- Integration with UML Models: Activity diagrams complement use case diagrams, sequence diagrams, and state diagrams, providing a detailed behavioral perspective of a system.



*Figure 3.3 Activity Diagram*

### 3.4 Use Case Diagram:

A **UML** use case diagram is the primary form of system/software requirements for a new software program underdeveloped. Use cases specify the expected behavior (what), and not the exact method of making it happen (how). Use cases once specified can be denoted both textual and visual representation (i.e. use case diagram). A key concept of use case modeling is that it helps us design a system from the end user's perspective. It is an effective technique for communicating system behavior in the user's terms by specifying all externally visible system behavior.



*Figure 3.4 Use Case Diagram*

### 3.5 Sequence Diagram:

- **Login and Authentication Process:** This sequence diagram outlines the steps involved in user authentication. It includes interactions between the user, authentication service, and database. The process starts when a user enters credentials, which are verified against stored records. Upon successful verification, the system grants access and retrieves user-specific settings.
- **Translation Request Handling:** This diagram details how a user submits a translation request, which is processed by the translation engine. The engine retrieves linguistic data, applies NLP techniques, and returns a translated output to the user interface. Error handling mechanisms are included to manage failed translations or unsupported languages.
- **Document Upload and Processing:** This sequence represents the workflow for document translation. Users upload files through the interface, triggering the backend to extract text, process it through the translation module, and return the translated document. The system ensures file format compatibility, progress tracking, and result storage for future reference.
- **Speech Recognition Workflow:** The diagram illustrates how speech-to-text conversion is handled. The user provides audio input, which is processed by the speech recognition API. The system converts the audio into text, applies necessary linguistic corrections, and returns the translated text. The workflow includes real-time processing steps and feedback loops for better accuracy.
- **Machine Learning Workflow:** This sequence diagram represents the processing flow for AI-driven translation improvements. The system collects datasets, processes text using NLP, pre-processes images when applicable, and applies neural network algorithms to build a prediction model. The model is continuously trained for improved accuracy.

- **Prediction and Accuracy Calculation:** This sequence outlines how the system refines its translations over time. Predictions from the neural network model are compared with expected results to improve accuracy, ensuring a more reliable translation output for users.

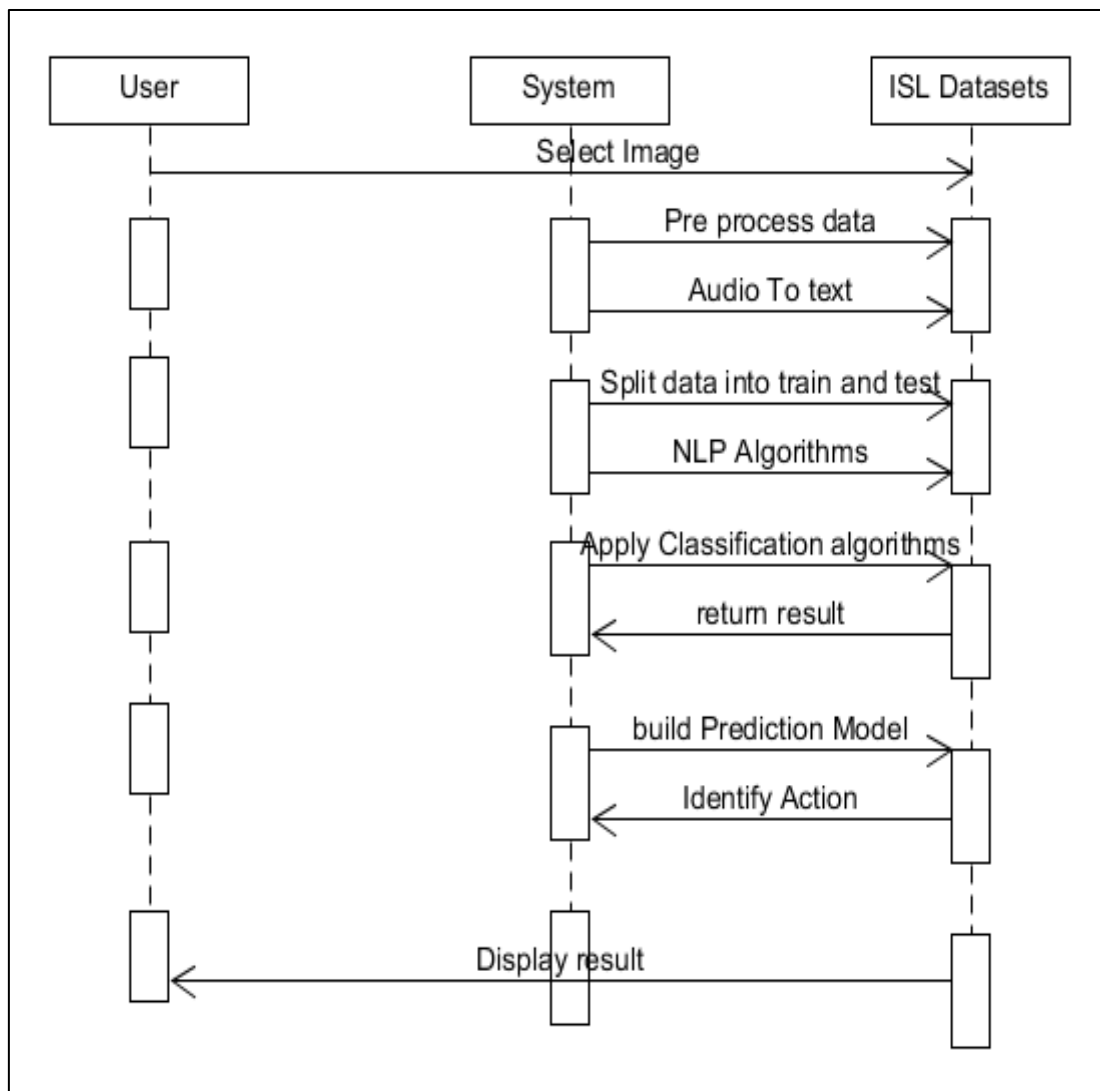


Figure 3.5 Sequence Diagram

## **CHAPTER 4:**

# **SYSTEM REQUIREMENT SPECIFICATIONS**

## CHAPTER 4

### SYSTEM REQUIREMENT SPECIFICATIONS

#### 4.1 Introduction:

The introduction of the Software Requirements Specification (SRS) provides an overview of the entire SRS with purpose, scope, definitions, acronyms, abbreviations, references and overview of the SRS. The aim of this document is to gather, analyse, and give an in-depth insight of the complete “Smart Translation for physically challenged people” by defining the problem statement in detail.

#### 4.2 Purpose

The Purpose of the Software Requirements Specification is to give the specialized, Functional and non-useful highlights, needed to build up a web application App. The whole application intended to give client adaptability to finding the briefest as well as efficient way. To put it plainly, the motivation behind this SRS record is to give an itemized outline of our product item, its boundaries and objectives. This archive depicts the task's intended interest group and its UI, equipment and programming prerequisites. It characterizes how our customer, group and crowd see the item and its usefulness.

##### 4.2.1 Scope

The scope of this system is to presents a review on data mining techniques used for the prediction of translation of sign language. It is evident from the system that data mining technique, like classification, is highly efficient in prediction of Indian automobile.



### 4.3 Software Architecture:

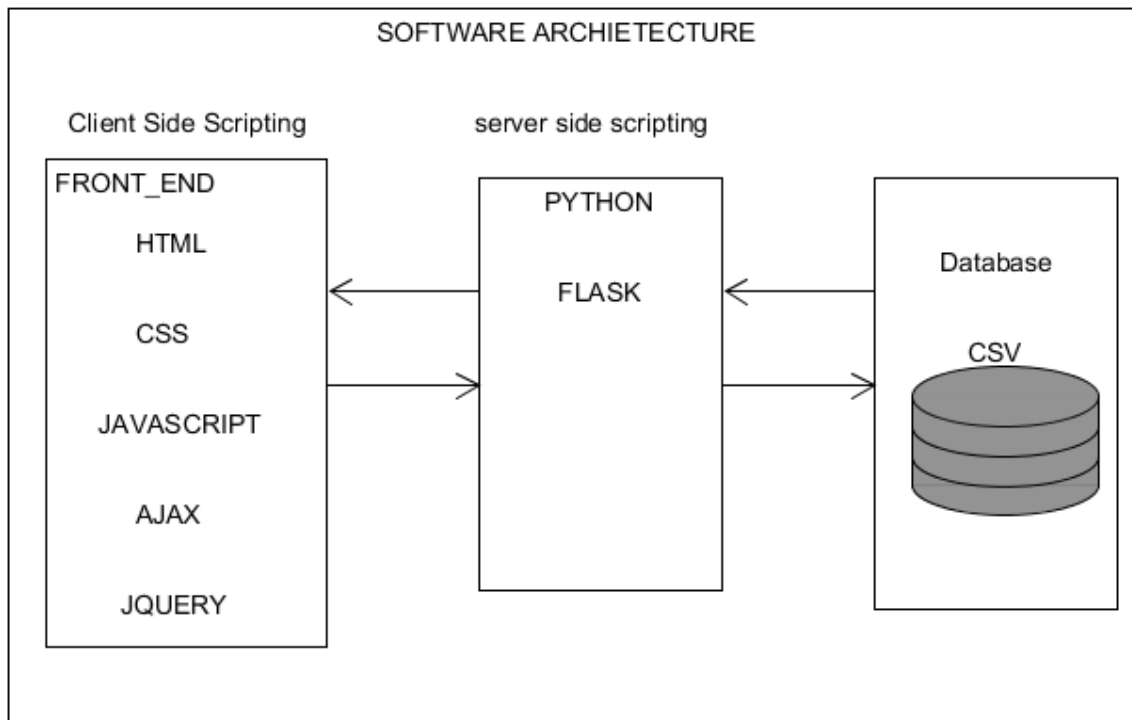


Figure 4.1 Software architecture

#### 4.3.1 Acronyms and Abbreviation:

##### ➤ Python:

Python is a deciphered, significant level, broadly useful programming language. Made by Guido van Rossum and first delivered in 1991, Python's plan reasoning accentuates code meaningfulness with its prominent utilization of critical whitespace. Its language develops and object-arranged methodology plan to assist software engineers with composing clear, consistent code for little and huge scope ventures.

Python is progressively composed and trash gathered. It underpins numerous programming standards, including procedural, object-arranged, and practical programming. Python is frequently portrayed as a "batteries included" language because of its thorough standard library. Python is a multi-worldview programming language. Article arranged programming and organized writing computer programs are completely upheld, and a significant number

of its highlights uphold useful programming and angle situated programming (counting by metaprogramming and metaobjects (enchantment methods)).] Many different standards are upheld by means of expansions, including plan by agreement and rationale programming.

➤ **Flask:**

Flask is a miniature web system written in Python. It is delegated a microframework in light of the fact that it doesn't need specific apparatuses or libraries.[3] It has no information base deliberation layer, structure approval, or whatever other segments where prior outsider libraries give normal capacities. In any case, Flask upholds augmentations that can include application includes as though they were executed in Flask itself. Augmentations exist for object-social mappers, structure approval, transfer dealing with, different open confirmation advancements and a few basic system related devices. Augmentations are refreshed unmistakably more as often as possible than the center Flask program.

➤ **Anaconda:**

Anaconda is a free and open-source circulation of the programming dialects Python and R . The dissemination accompanies the Python translator and different bundles identified with AI and information science.

Essentially, the thought behind Anaconda is to make it simple for individuals inspired by those fields to introduce all (or a large portion) of the bundles required with a solitary establishment.

An open-source bundle and condition the executive's framework called Conda, which makes it simple to introduce/update bundles and make/load situations.

- AI libraries like TensorFlow, scikit-learn and Theano.
- Information science libraries like pandas, NumPy and Dask.
- Perception libraries like Bokeh, Datashader, matplotlib and Holoviews.
- Jupyter Notebook, a shareable note pad that joins live code, representations and text.

➤ **Numpy:**

NumPy is the principal bundle for logical registering with Python. It contains in addition to other things:

- An amazing N-dimensional cluster object.
- sophisticated (broadcasting) capacities.
- tools for incorporating C/C++ and Fortran code.
- useful straight polynomial math, Fourier change, and arbitrary number abilities.

➤ **Pandas:**

Pandas is an open source, BSD-authorized library giving elite, simple to-utilize information structures and information investigation apparatuses for the Python programming language.

Pandas is a Num FOCUS supported undertaking. This will help guarantee the achievement of improvement of pandas as an a-list open-source venture, and makes it conceivable to give to the task.

#### 4.4 Functional Requirements:

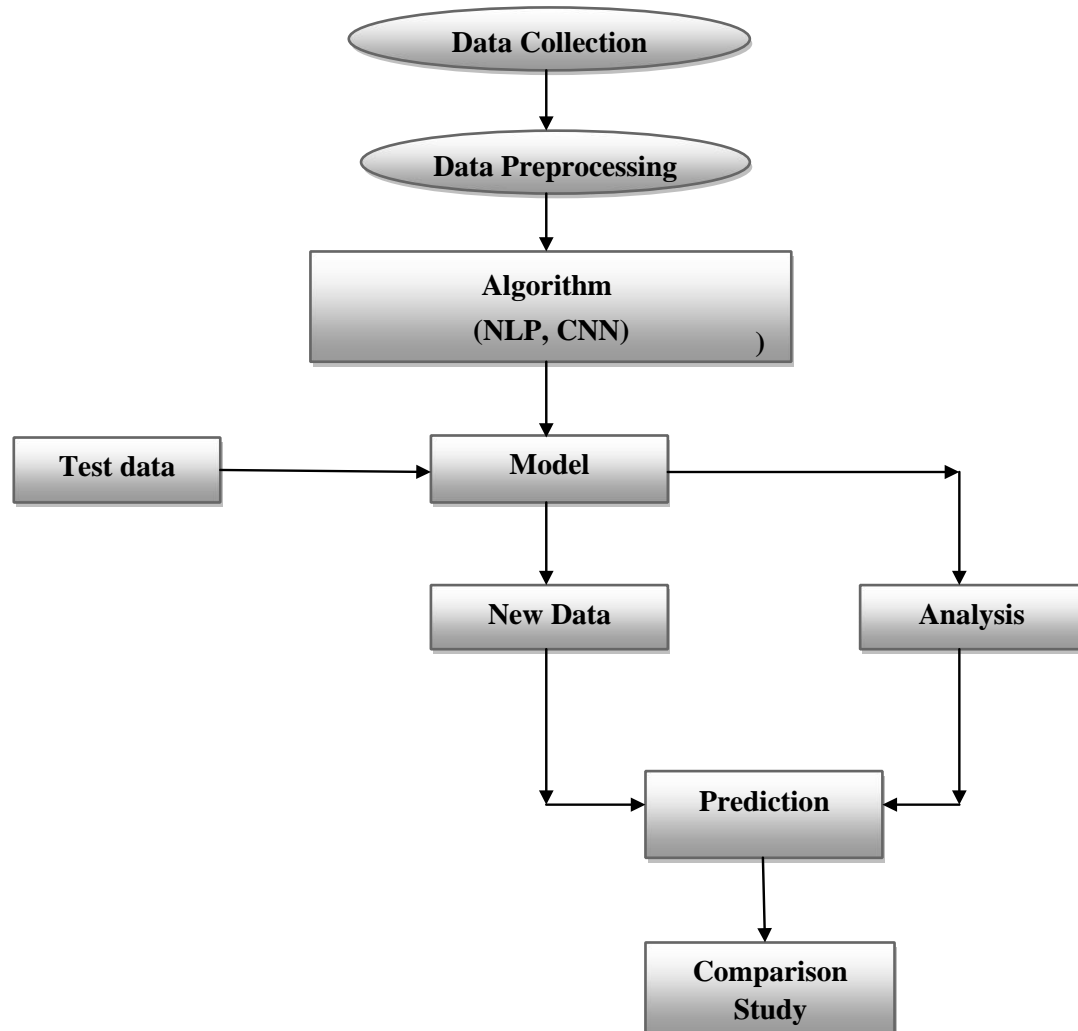


Figure 4.2 Functional Flow Diagram

##### 4.4.1 Product Functions

1. Uploading Data Uploading data sets to the database.
2. Perform is done by each algorithm based on the constraints. prediction By entering the audio can view the details of the vehicle details along with the result.
4. Comparison Study Prediction results and its stages of each algorithm is represented through graph.

#### 4.4.2 General Constraints:

The results generated have to be entered in to the system and any error or any value entered out of the boundary will not be understood by the system. In any case if the database crashes, the whole information collected and the results generated will be of no use.

### 4.5 Specific Requirements:

This section provides a detailed description of all inputs into and outputs from the system. It also gives a description of the hardware, software and communication interfaces and provides basic prototypes of the user interface.

#### 4.5.1 Hardware Requirements:

- **Processor:** Intel Core i5 or equivalent ARM-based processor.
- **RAM:** Minimum 4GB for smooth performance.
- **Storage:** At least 50GB of free space.
- **Camera:** HD webcam for sign language recognition.
- **Microphone:** High-quality microphone for speech input.

#### 4.5.2 Software Requirements:

- Operating System: Windows, Linux, or macOS.
- Programming Languages: Python, JavaScript.
- Frameworks & Libraries:
  - Django for backend development.
  - TensorFlow and OpenCV for AI processing.
  - MediaPipe for gesture detection.
- Database: Firebase for authentication and real-time database.
- Tools: Android Studio, Visual Studio Code, and Jupyter Notebook.

## 4.6 Non-Functional requirements

A non-functional requirement is a requirement that specifies criteria that can be used to judge the operation of a system, rather than specific behaviors. They are contrasted with functional requirements that define specific behavior or functions. The plan for implementing functional requirements is detailed in the system design. The plan for implementing non-functional requirements is detailed in the system architecture, because they are usually Architecturally Significant Requirement.

### 1. Accessibility:

It refers to the design of products, devices, services, or environments for people who experience disabilities. The concept of accessible design and practice of accessible development ensure both "direct access" (i.e. unassisted) and "indirect access" meaning compatibility with a person's assistive technology.

### 2. Simplicity:

The project is driven by a simple user interface.

### 3. Availability:

Availability of a system may also be increased by the strategy of focusing on increasing testability, diagnostics and maintainability and not on reliability. Improving maintainability during the early design phase is generally easier than reliability (and Testability & diagnostics). Maintainability estimates (item Repair by replacement rates) are also generally more accurate.

### 4. Reliability:

The system should not crash and should identify invalid input and produce suitable error message.

### 5. Usability:

The interface should be intuitive and easily navigable and user friendly.

**6. Integrity:**

The software does not store any cache data or doesn't use system resources in background.

**7. Authentication:**

Only authorized nodes can communicate with others.

**4.7 Feasibility Study**

The feasibility study which helps to find solutions to the problems of the project. The solution which is given that how looks as a new system look like.

**4.7.1 Technical Feasibility**

The project entitled “**smart translation for physically challenged people**” is technically feasible because of the below mentioned features. The project is developed in Python. The web server is used to develop translation on local serve. The local server very neatly coordinates between design and coding part. It provides a Graphical User Interface to design an application while the coding is done in python. At the same time, it provides high level reliability, availability and compatibility.

**4.7.2 Economic Feasibility**

In economic feasibility, cost benefit analysis is done in which expected costs and benefits are evaluated. Economic analysis is used for effectiveness of the proposed system. In economic feasibility the most important is cost-benefit analysis. The system is feasible because it does not exceed the estimated cost and the estimated benefits are equal.

**4.7.3 Operational Feasibility**

The project entitled smart translation for physically challenged people is technically feasible because of the below mentioned features. The system identifies the sign language recommendation behaviour and its stages based on data. The performance of the Data mining techniques is compared based on their execution time and displayed it through graph.

## **CHAPTER 5:**

# **METHODOLOGY**

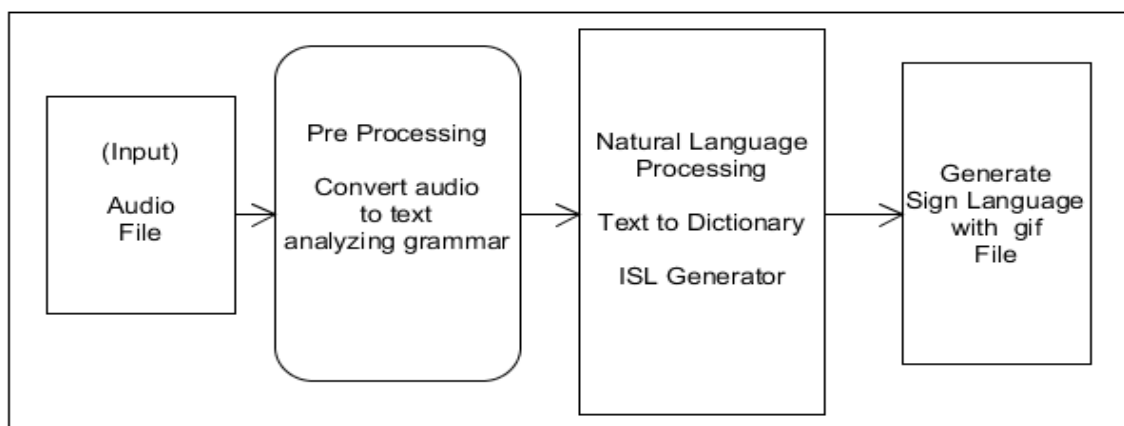


## CHAPTER 5

### METHODOLOGY

#### 5.1 Methodology

This project is divided into two sections. It translates an audio message into sign language, and in the audio to sign language conversion we are going to consider audio as input and displaying the gif file as output. In this project we are going to collect the datasets of gif file. Gui is developed for taking Audio as input to our project. The audio is Converted to text using microphone. The generated text are need to be split and store it in dictionary file. With help of Dependency parser sentence and obtaining relationship between words. Each obtained words are considered as text. Text Pre-processing is done using NLP. Dictionary based Machine Translations of input sentence using ISL grammar rules Generation of Sign language with GIF file.



*Figure 5.1 Audio to sign language conversion*

In this approach we are going to collect the deaf and dumb people Indian sign language datasets. Applying Pre Processing Techniques(resize, noise remove, grayconversion) for the collected datasets. Splitting the data into train and test. Applying Cnn Algorithms for training datasets. For the trained model we are going to Build Model and store in system. The model we have build we are going to pass video as input to the model and recognising the language based on the user hand movement.

**5.1.1 Collection datasets:**

- We are going to collect datasets for the prediction from the kaggle.com
- The data sets consists of many Classess

**5.1.2 Data Pre-Processing:**

- In data pre-processing we are going to perform some image pre-processing techniques on the selected data
- Image Resize
- And Splitting data into train and test

**5.1.3 Data Modelling:**

- The splitted train data are passed as input to the CNN algorithm, which helps in training.
- The trained image data evaluated by passing test data to the algorithm
- Accuracy is calculatee

**5.1.4 Build Model:**

- Once the data is trained and if it showing the accuracy rate as high, then we need to build model file

**5.1.5 Input Video:**

- We are going to pass input as image or video
- Videos are converted into frames
- Each frame is passed as input to the build model
- Model going to detect the sign images

**5.2 CNN Algorithms:****5.2.1 Convolutional Neural Networks have the following layers:**

- Convolutional
- ReLU Layer
- Pooling

- Fully Connected Layer

### Step1: Convolution Layer:

- Convolutional neural networks apply a filter to an input to create a feature map that summarizes the presence of detected features in the input.

### Step2: ReLU Layer

- In this layer, we remove every negative value from the filtered images and replaces them with zeros. It is happening to avoid the values from adding up to zero.
- **Rectified Linear unit (ReLU)** transform functions only activates a node if the input is above a certain quantity. While the data is below zero, the output is zero, but when the information rises above a threshold. It has a linear relationship with the dependent variable.

### Step3: Pooling Layer

- In the layer, we shrink the image stack into a smaller size. Pooling is done after passing by the activation layer. We do by implementing the following 4 steps:
  - Pick a **window size** (often 2 or 3)
  - Pick a **stride** (usually 2)
  - **Walk** your Window **across** your **filtered** images
  - From each **Window**, take the **maximum** value

### Step4: Fully Connected Layer

- The last layer in the network is **fully connected**, meaning that neurons of preceding layers are connected to every neuron in subsequent layers. This **mimics high-level reasoning** where all possible pathways from the input to output are considered. Then take the shrunk image and put into the single list, so we have got after passing through two layers of convolution reLO and pooling and then converting it into a single file or a vector.

### 5.2.2 CNN

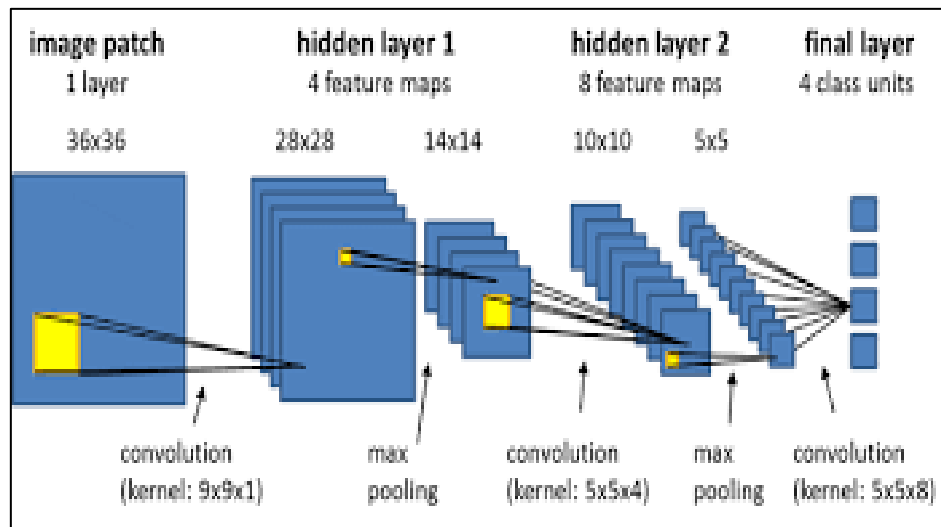


Figure 5.2 CNN layers

### 5.3 Working:

Some researchers divide the deep learning algorithms into four categories: Convolutional Neural Networks, Restricted Boltzmann Machines, Auto encoders and Sparse Coding. In this work of images classification, we opted for the Convolutional Neural Networks (CNNs) because they are the widely adapted for image classification with good performances by making convolutional networks fast to train.

Convolutional Neural Networks (CNN) approaches are commonly used with architectures having multiple layers that are trained. There are two parameters (weights and bias) in each layer. The first layers find low-level features for instance edges, lines and corners and the other layers find mid-level and high-level features for instance structures, objects, and shapes. Then the prediction output is used to compute the loss cost to the ground truth labels. Second, based on the loss cost, the backward stage computes the gradient of each parameter with chain rules. All the parameters are updated based on the gradients and are prepared for the next forward computation. After a number of iterations of the forward and backward stages, the network learning can be stopped.

Clarification of what was meant by the term “convolution” would be useful, since it is used to describe mathematical principles and ideas about the feature transformation

approach and process. In mathematics, specifically in algebraic topology, convolution is a mathematical transformation on two functions ( $u$  and  $v$ ); it generates a third function, which is normally viewed as a transformed version of one of the initial functions. In the case of two real or complex functions,  $u$  and  $v$  the convolution is another function, which is usually denoted  $u * v$  and which is defined by (Hirschman and Widder, 2017):

$$(u * v)(x) = \int_{-\infty}^{+\infty} u(x-t)v(t) dt \quad (1)$$

The convolution satisfies some algebraic properties (e.g. commutativity, associativity, distributivity and multiplicative identity) in order to ensure that the characteristics of their corresponding geometric pictures are preserved. Commutativity

$$u * v = v * u \quad (2)$$

Associativity

$$u * (v * h) = (u * v) * h \quad (3)$$

Associativity with scalar multiplication

$$a * (u * v) = (a * u) * v \quad (4)$$

Distributivity

$$u * (v + h) = (u * v) + (u * h) \quad (5)$$

In artificial intelligence, convolutional neural networks apply multiple cascaded convolution kernels with deep learning applications. Formally, the filtering operation performed by a feature map is a discrete convolution. Discrete convolution can be seen as multiplication by a matrix. Though, the matrix has several entries with constraints to be equal to other entries.

For functions defined on the set of integers, the convolution of two finite sequences is determined by extending the sequences to finitely applicable functions on the set of integers (e.g. a finite summation may be used for a finite support). The discrete convolution can be defined by following formula.

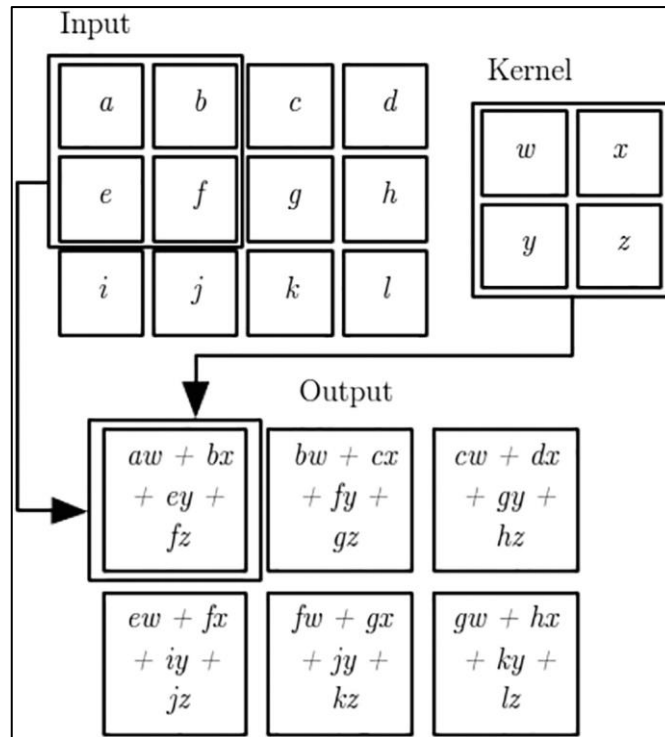


Figure 5.3 CNN functionality

$$(u \vee n) \left( \int_{-\infty}^{\infty} u(n - m) v(m) dt \right) = m \vee m dt$$

$M = (6)$

In image processing, the convolution is done by accomplishing a form of mathematical operation between matrices representing respectively a kernel and an image. This convolution operation is made by adding each element of the image to its local neighbors, weighted by the kernel.

The values of a given pixel in the output image are determined by multiplying each kernel value by the matching input image pixel values. This can be represented algorithmically with the pseudo-code illustrated below:

```
for each image row in input image:
    for each pixel in images row:
        set accumulator to zero
        for each kernel row in the kernel:
            for each element in the kernel row:
                if element position corresponding to pixel position then
                    multiply element value corresponding to pixel value
                    add result to accumulators
                endif
            Endfor
        Endfor
    Endfor
    set output image pixel to accumulator
```

*Figure 5.4 Pseudo Code*

The basic CNN architecture includes Convolution layer, Pooling layer, ReLU layer and Fully connection layer.

The real power of deep learning, especially for image recognition, comes from convolutional layers. It is the first and the most important layer. In this layer, a CNN uses different filters to convolve the whole image as well as the intermediate feature maps, generating various feature maps. Feature map consists of a mapping from input layers to hidden layers.

Depth of the output volume controls the number of neurons in the layer that connect to the same region of the input volume. All of these neurons will learn to activate for different features in the input. For instance, if the first Convolutional Layer takes the raw image as input, then different neurons along the depth dimension may activate in the presence of various oriented edges, or blobs of color.

There are three main advantages of the convolution operation:

- The weight sharing mechanism in the same feature map reduces the number of parameters
- Local connectivity learns correlations among neighboring pixels
- Invariance to the location of the object.

One interesting approach to handling the convolutional layers is the Network In Network (NIN) method where the main idea is to substitute the conventional layer with a small multilayer perceptron consisting of multiple fully connected layers with nonlinear activation functions, thereby replacing the linear filters with nonlinear neural networks. This method achieves good results in image classification.

### 5.3.1 ReLU Layer

- It is the Rectified Linear Units Layer. This is a layer of neurons that applies the non-saturating non-linearity function or loss function:

$$f(x) = \max(0, x) \quad (7)$$

- It yields the nonlinear properties of the decision function and the overall network without affecting the receptive fields of the convolution layer. We have saturated nonlinear functions that are much slower (Krizhevsky et al., 2012). Also, we have the tanh function:

$$f(x) = \tanh(x) \quad (8)$$

or the logistic sigmoid function:

$$f(x) = \frac{1}{1 + e^{-x}} \quad (9)$$

- ReLU results in the neural network that is training several times rapidly, without making a significant difference to generalization accuracy.

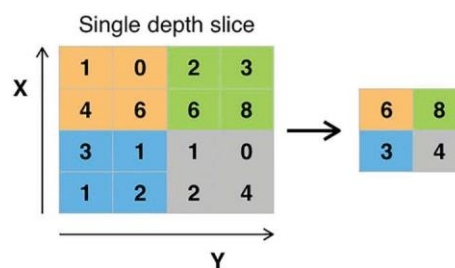
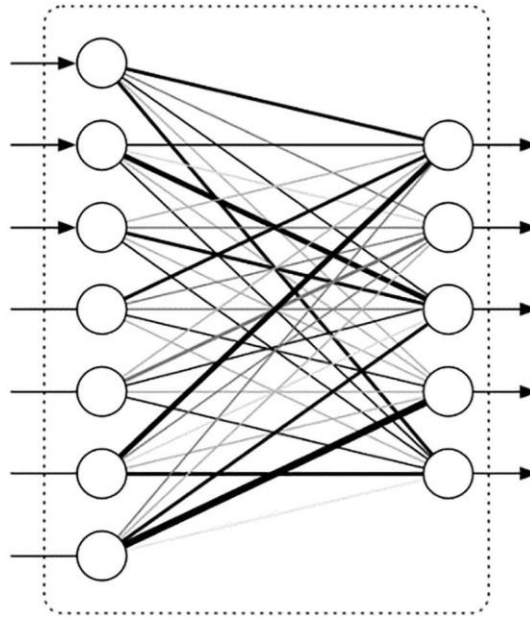


Figure 5.5 Max pooling process





### 5.3.2 Pooling layer

Its task consists to simplify or reduce the spatial dimensions of the information derived from the feature maps. We have three types of pooling: the first one is the average pooling, the second is the L2-norm pooling and finally the most popular use is the max pooling (because of its speed and improved convergence. This basically takes a filter (normally of size  $2 \times 2$ ) and a stride of the same length. It then applies it to the input volume and outputs the maximum number in every sub region that has the filter which convolves around.

### 5.3.3 Fully connection layer

The input to this layer is a vector of numbers. Each of the inputs is connected to every one of the outputs hence the term “fully connected”. It is the last layer, in general after the last pooling layer in CNN process. Fully connected layers perform like a traditional neural network and contain about 90% of the parameters in CNN. This layer basically takes as input the output of the last pooling layer and outputs an N dimensional vector where N is the number of classes that the program has to choose from. It allows us to feed forward the neural network into a vector with a predefined length. For image classification, we could feed forward the vector into certain number categories. The output is also a vector of numbers.

### 5.3.4 Loss layer

Loss layer uses functions that take in the model's output and the target, and it computes a value that measures the model's performance. It can have two main functions:

- Forward (input, target): calculates loss value-based input and target value.
- Backward (input, target): calculates the gradient of the loss function associated with the criterion and return the result.
- Backpropagation is a principle used to compute the gradient of the loss function (calculates the cost associated with a given state) with respect to the weights in a CNN.

## **CHAPTER 6:**

# **TESTING AND RESULTS**

## CHAPTER 6

### TESTING AND RESULTS

#### 6.1 Introduction

Testing is the way toward running a framework with the expectation of discovering blunders. Testing upgrades the uprightness of the framework by distinguishing the deviations in plans and blunders in the framework. Testing targets distinguishing blunders – prom zones. This aides in the avoidance of mistakes in the framework. Testing additionally adds esteems to the item by affirming the client's necessity.

The primary intention is to distinguish blunders and mistake get-prom zones in a framework. Testing must be intensive and all around arranged. A somewhat tried framework is as terrible as an untested framework. Furthermore, the cost of an untested and under-tried framework is high. The execution is the last and significant stage. It includes client preparation, framework testing so as to guarantee the effective running of the proposed framework. The client tests the framework and changes are made by their requirements. The testing includes the testing of the created framework utilizing different sorts of information. While testing, blunders are noted and rightness is the mode.

#### 6.2 Objectives of Testing

- Testing in a cycle of executing a program with the expectation of discovering mistakes.
- A effective experiment is one that reveals an up 'til now unfamiliar blunder.

Framework testing is a phase of usage, which is pointed toward guaranteeing that the framework works accurately and productively according to the client's need before the live activity initiates. As expressed previously, testing is indispensable to the achievement of a framework. Framework testing makes the coherent presumption that if all the framework is right, the objective will be effectively accomplished. A progression of tests are performed before the framework is prepared for the client acknowledgment test.

### **6.3 Testing methods**

System testing is a stage of implementation. This helps the weather system works accurately and efficiently before live operation commences. Testing is vital to the success of the system. The candidate system is subject to a variety of tests: online response, volume, stress, recovery, security, and usability tests series of tests are performed for the proposed system are ready for user acceptance testing.

#### **6.3.1 White Box Testing**

The test is conducted during the code generation phase itself. All the errors were rectified at the moment of its discovery. During this testing, it is ensured that

- All independent module have been exercised at least one
- Exercise all logical decisions on their true or false side.
- Execute all loops at their boundaries.

#### **6.3.2 Black Box Testing**

It is focused around the practical necessities of the product. It's anything but a choice to white box testing; rather, it is a reciprocal methodology that is probably going to reveal an alternate class of blunders than White Box strategies. It is endeavored to discover mistakes in the accompanying classes.

- Incorrect or missing capacities
- Interface blunders
- Errors in an information structure or outside information base access

#### **6.3.3 Unit Testing**

Unit testing chiefly centers around the littlest unit of programming plan. This is known as module testing. The modules are tried independently. The test is done during the programming stage itself. In this progression, every module is discovered to be working acceptably as respects the normal yield from the module.

#### **6.3.4 Integration Testing**

Mix testing is an efficient methodology for developing the program structure, while simultaneously leading tests to reveal blunders related with the interface. The goal is to take unit tried modules and manufacture a program structure. All the modules are joined and tried in general.

#### **6.3.5 Output Testing**

Subsequent to performing approval testing, the following stage is yield trying of the proposed framework, since no framework could be valuable on the off chance that it doesn't create the necessary yield in a particular configuration. The yield design on the screen is discovered to be right. The organization was planned in the framework configuration time as indicated by the client needs. For the printed copy likewise, the yield comes according to the predefined prerequisites by the client. Subsequently yield testing didn't bring about any amendment for the framework.

#### **6.3.6 User Acceptance Testing**

Client acknowledgment of a framework is the vital factor for the achievement of any framework. The framework viable is tried for client acknowledgment by continually staying in contact with the imminent framework clients at the hour of creating and making changes at whatever point required.

### **6.4 Validation**

Toward the consummation of the reconciliation testing, the product is totally amassed as bundle interfacing blunders have been revealed and adjusted and a last arrangement of programming tests starts in approval testing. Approval testing can be characterized from multiple points of view, however a straightforward definition is that the approval succeeds when the product work in a way that is normal by the client. After approval test has been directed as follows:

- The capacity or execution qualities adjust to detail and are acknowledged.
- A deviation from the particular is revealed and a lack list is made.

- Proposed framework viable has been tried by utilizing an approval test and discovered to be working acceptably.

## 6.5 Test Reports

The users test the developed system when changes are made according to the needs. The testing phase involves the testing of the developed system using various kinds of data. An elaborate testing of data is prepared and system is tested using the test data. Test cases are used to check for outputs with different set of inputs.

## 6.6 Test Cases

Test Case	Test Purpose	Test condition	Expected outcome	Actual result	Pass or Fail
Load Audio data	Upload audio data	Load audio file	Uploaded successfully	Audio is loaded Successfully.	Pass
Pre - Processing	Apply methods like, noise removal, gray conversion	Pre-processing for audio	Audio is cleaned successfully	As Expected.	Pass
Audio to text	Sentence Formation	Pre-processing for audio	Audio is converted to text using pyaudio	As Expected.	Pass

Text to word to dictionary	Sentence data	Text into word dictionary using mlp	Converted succesfully	Display gif file	Pass
Input video	Convert videos into frames	Opencv is used to convert data to frames	Converted successfully	Display the result	Pass
Detecting object in frames	Train model	Pass frame data to the model to predict the result	Frames are passed successfully	Frames are successfully	Pass
Predict the result	Frames to the model	Model display the result based trained data	Class label prediction	Display the signs	Pass

Table 6.1: Test Cases

## 6.7 Experimental Results and Discussion

The framework is intended to perceive the hand motions made by the dumb people. The proposed framework is basic and the subject isn't needed to wear any gloves or any electromechanical gadget. The speech is eared by the system and is converted into alphabets. And accordingly the hand gesture is made visible for deaf peoples.eg: Riya is passed as a speech to system. The system coverts it to alphabet like R, I, Y, A where system matches the signs in database and provides output.

The proposed web application seeks to develop a translating mechanism or automation that includes a parser element that converts the incoming speech data or English text to a phrase structure grammar representation, which is then used by another module that

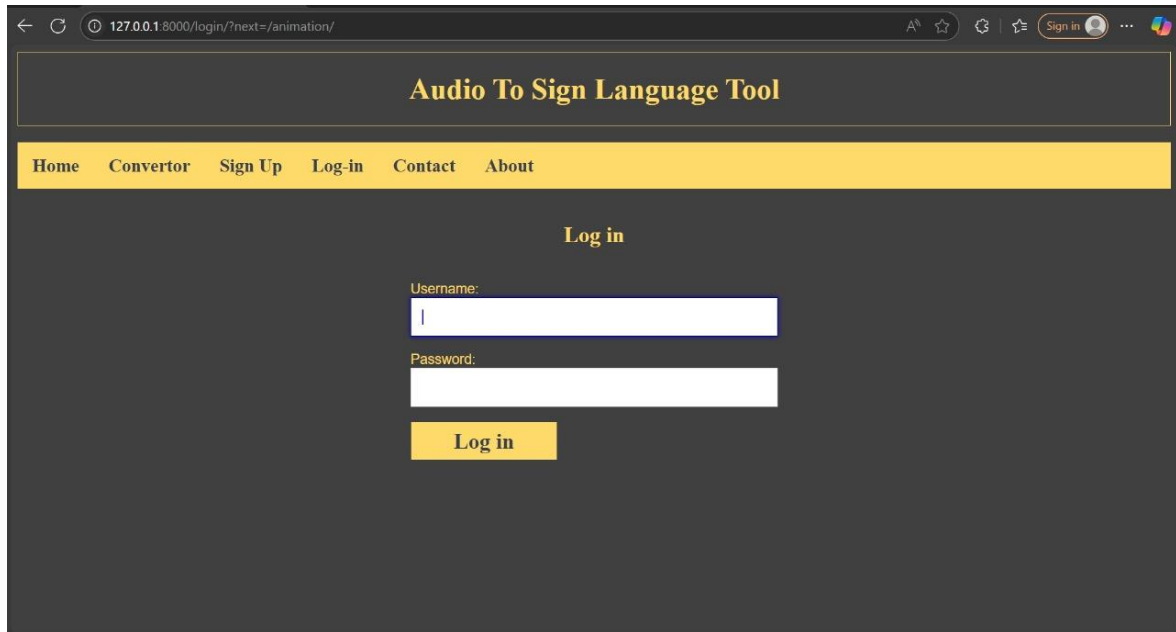


contains Indi Sign language grammatical format. This is accomplished through the means of removing stop-words from the reordered input format. Because Indian sign language does not provide word inflections, stemming and lemmatization are used to turn words into their root form. Following sentence filtration, all words are tested against the words in the database, which is represented as a dictionary comprising video representations of each word. If the words are missing from the database, the algorithm will then look for its related synonym and replace it with that term. In many ways, the proposed system is more innovative and efficient than existing systems, because Existing methods can only convert words directly into Indi sign language, and they were not as efficient as this system, whereas this in the actual world, the system tries to translate these phrases into Indian sign language grammatical order. Because this is a web-based programmed, it is straightforward to access and use. This technology is platform agnostic and more versatile to use, and it transforms phrases to sign language in real time.

## **CHAPTER 7: SNAPSHOTS**

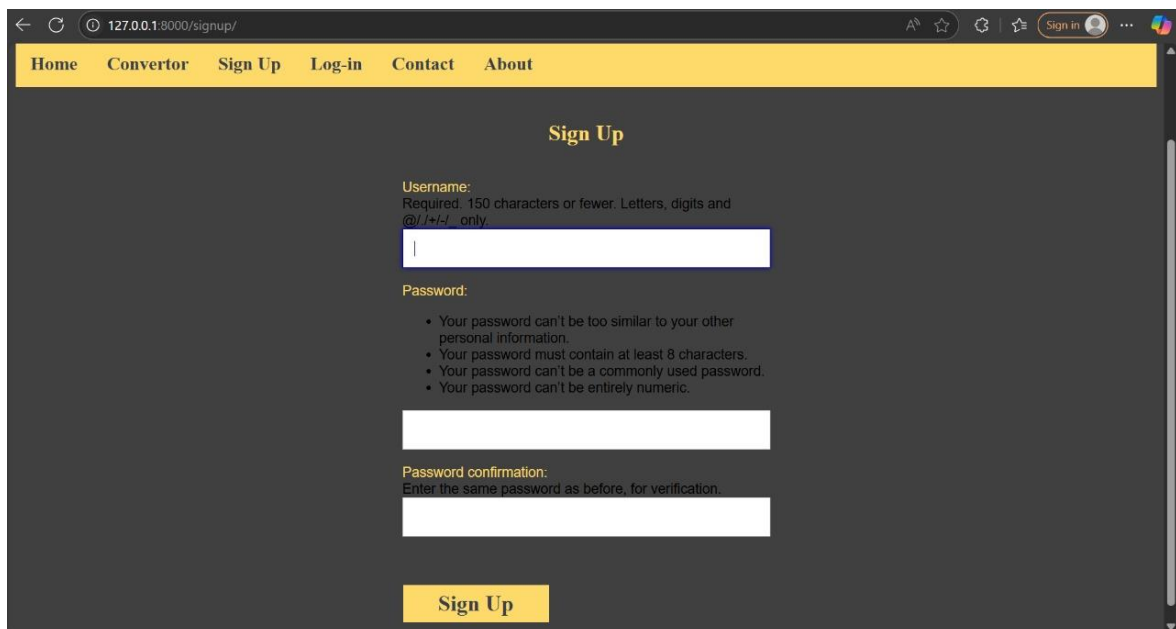
## CHAPTER 7

### SNAPSHOTS



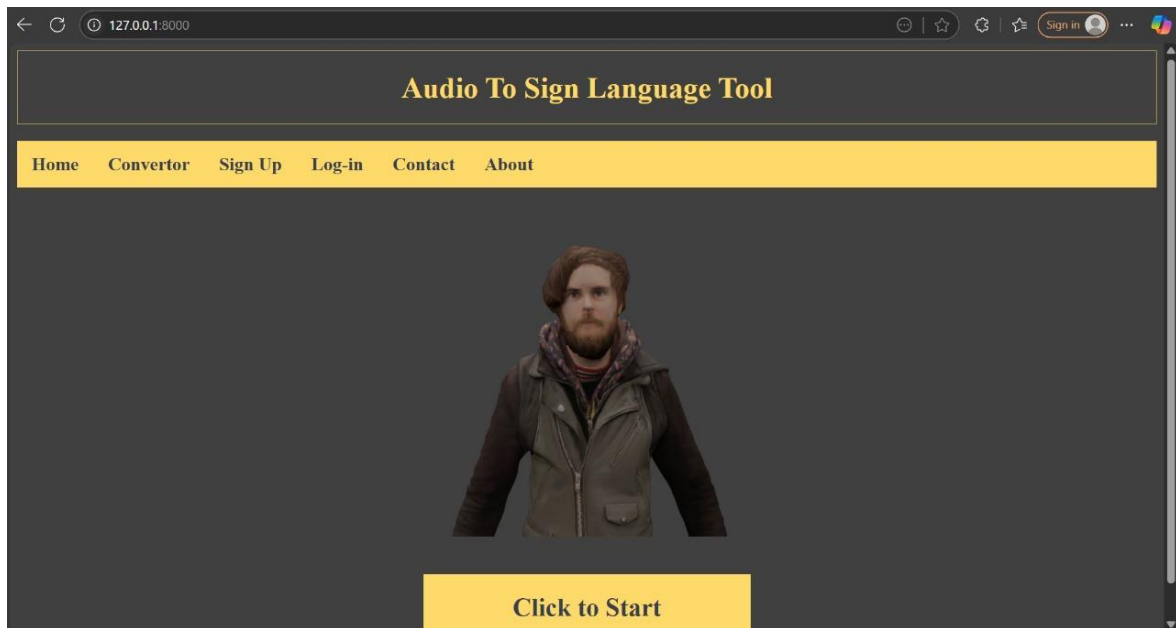
The screenshot shows a web browser window with the URL `127.0.0.1:8000/login/?next=/animation/`. The page title is "Audio To Sign Language Tool". A yellow navigation bar contains links: Home, Convertor, Sign Up, Log-in, Contact, and About. The main content area is titled "Log in" and features a form with "Username:" and "Password:" labels, each followed by a white input field. Below the fields is a yellow "Log in" button.

Figure 7.1: Login Page

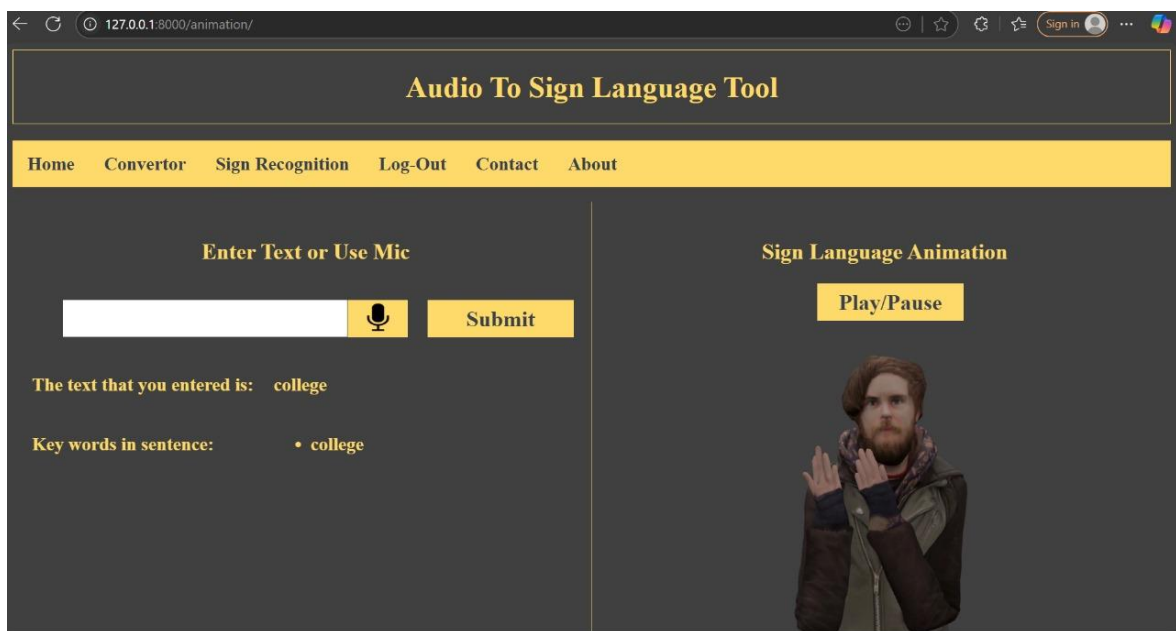


The screenshot shows a web browser window with the URL `127.0.0.1:8000/signup/`. The page title is "Audio To Sign Language Tool". A yellow navigation bar contains links: Home, Convertor, Sign Up, Log-in, Contact, and About. The main content area is titled "Sign Up" and features a form. The "Username:" label is followed by a white input field and a note: "Required: 150 characters or fewer. Letters, digits and @/./+/-/\_ only". The "Password:" label is followed by a white input field and a list of requirements: "Your password can't be too similar to your other personal information.", "Your password must contain at least 8 characters.", "Your password can't be a commonly used password.", and "Your password can't be entirely numeric.". Below the password field is a "Password confirmation:" label with the instruction "Enter the same password as before, for verification" and another white input field. A yellow "Sign Up" button is at the bottom.

Figure 7.2: Sign Up Page



*Figure 7.3: Start of the Tool*



*Figure 7.4: Text/Speech to sign conversion*

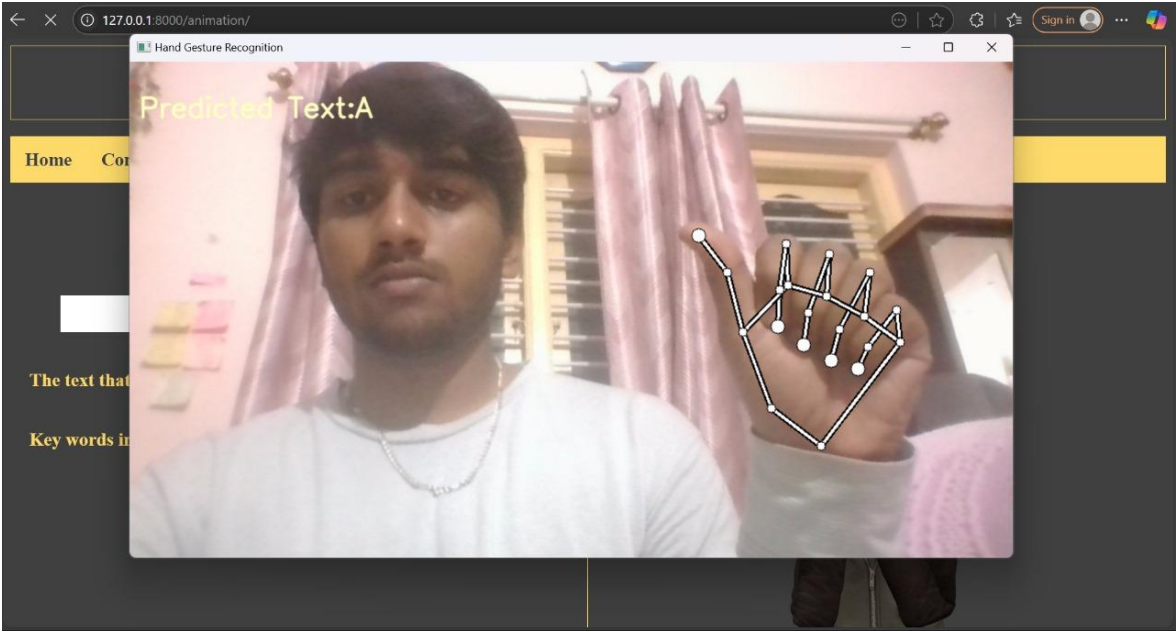


Figure 7.5: Sign language to text conversion

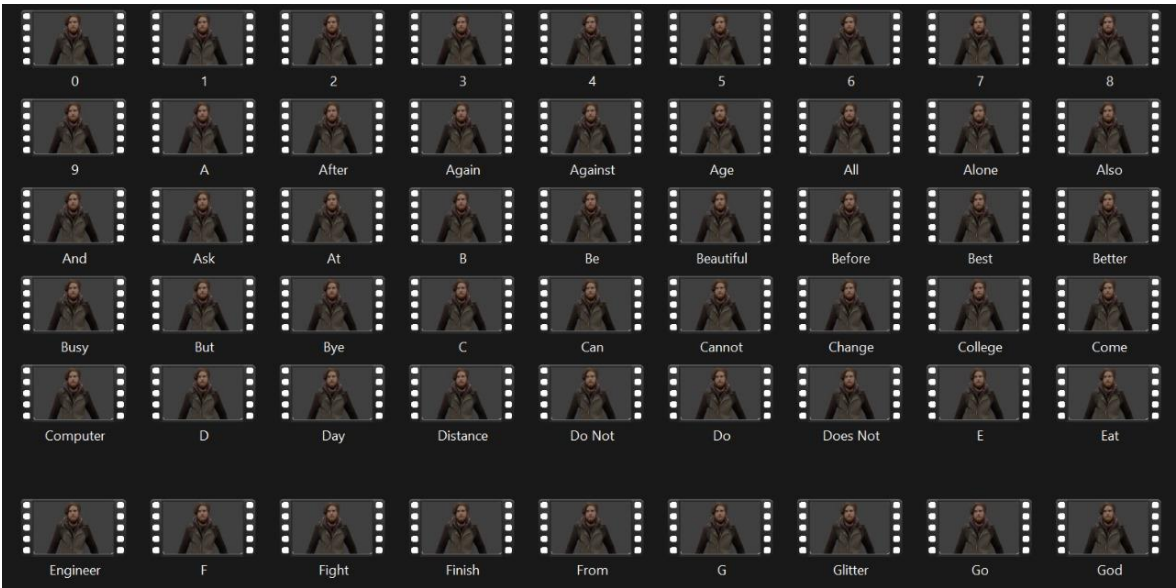


Figure 7.6: Trained Data - GIFs

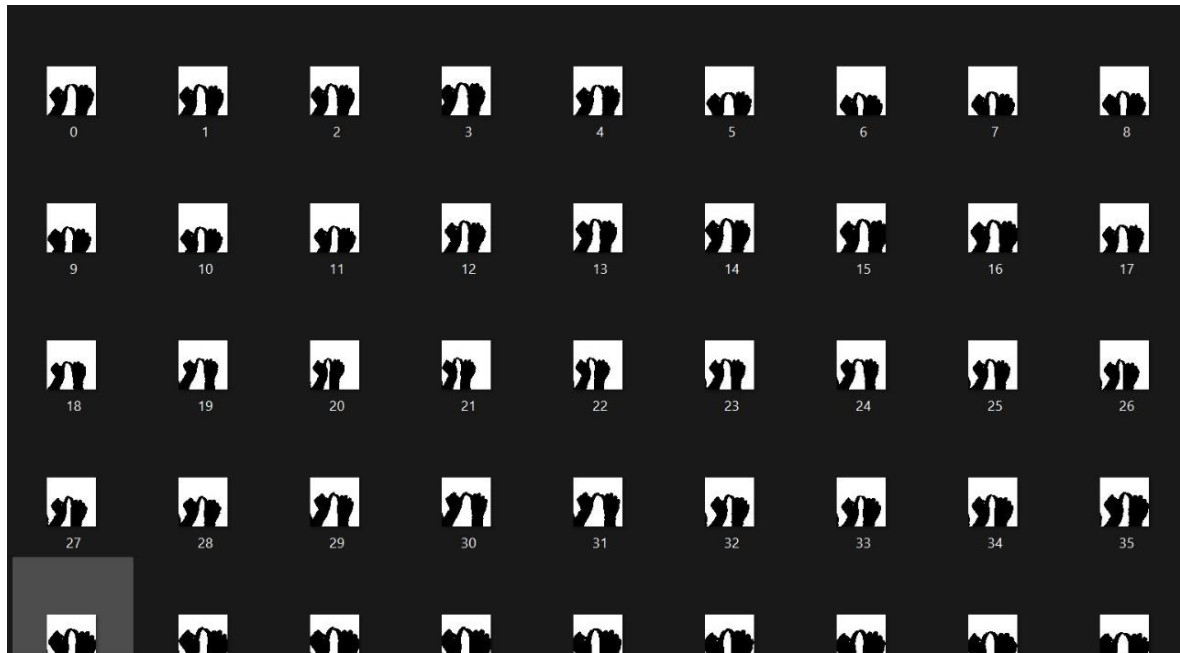


Figure 7.7: Hand Sign Images

A1	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
1	0	0	0	-0.32877	-0.13699	-0.50685	-0.49315	-0.58904	-0.79452	-0.75342	-1	-0.30137	-0.73973	-0.35616	-0.91781	-0.34247	-0.71233	-0.30137	-0.50685
2	0	0	0	-0.31944	-0.13889	-0.48611	-0.52778	-0.56944	-0.81944	-0.77778	-1	-0.26389	-0.73611	-0.30556	-0.91667	-0.31944	-0.69444	-0.27778	-0.48611
3	0	0	0	-0.34247	-0.15068	-0.50685	-0.50685	-0.60274	-0.79452	-0.79452	-1	-0.30137	-0.73973	-0.34247	-0.89041	-0.32877	-0.67123	-0.28767	-0.45205
4	0	0	0	-0.32877	-0.15068	-0.50685	-0.50685	-0.58904	-0.79452	-0.76712	-1	-0.30137	-0.71233	-0.34247	-0.89041	-0.35616	-0.71233	-0.32877	-0.50685
5	0	0	0	-0.33333	-0.125	-0.51389	-0.5	-0.61111	-0.80556	-0.80556	-1	-0.30556	-0.75	-0.375	-0.90278	-0.36111	-0.69444	-0.31944	-0.47222
6	0	0	0	-0.32877	-0.13699	-0.50685	-0.49315	-0.58904	-0.79452	-0.76712	-1	-0.30137	-0.72603	-0.35616	-0.87671	-0.35616	-0.68493	-0.32877	-0.49315
7	0	0	0	-0.34247	-0.12329	-0.52055	-0.49315	-0.60274	-0.79452	-0.78082	-1	-0.30137	-0.72603	-0.36986	-0.87671	-0.38356	-0.68493	-0.34247	-0.47945
8	0	0	0	-0.31507	-0.15068	-0.47945	-0.52055	-0.56164	-0.80822	-0.73973	-1	-0.24658	-0.73973	-0.30137	-0.90411	-0.30137	-0.69863	-0.26027	-0.49315
9	0	0	0	-0.32	-0.16	-0.50667	-0.52	-0.6	-0.8	-0.77333	-1	-0.32	-0.73333	-0.34667	-0.90667	-0.32	-0.69333	-0.29333	-0.48
10	0	0	0	-0.32394	-0.14085	-0.50704	-0.50704	-0.60563	-0.80282	-0.78873	-1	-0.30986	-0.73239	-0.35211	-0.92958	-0.33803	-0.73239	-0.32394	-0.52113
11	0	0	0	-0.33333	-0.13889	-0.51389	-0.51389	-0.61111	-0.80556	-0.79167	-1	-0.31944	-0.72222	-0.375	-0.91667	-0.36111	-0.73611	-0.31944	-0.54167
12	0	0	0	-0.32394	-0.12676	-0.50704	-0.50704	-0.59155	-0.80282	-0.77465	-1	-0.29577	-0.74648	-0.35211	-0.92958	-0.33803	-0.71831	-0.30986	-0.50704
13	0	0	0	-0.36111	-0.13889	-0.54167	-0.5	-0.65278	-0.79167	-0.81944	-1	-0.34722	-0.73611	-0.40278	-0.90278	-0.38889	-0.69444	-0.34722	-0.48611
14	0	0	0	-0.30986	-0.12676	-0.47887	-0.49296	-0.56338	-0.78873	-0.74648	-1	-0.26761	-0.73239	-0.32394	-0.91549	-0.32394	-0.70423	-0.29577	-0.49296
15	0	0	0	-0.33333	-0.13889	-0.5	-0.51389	-0.59722	-0.80556	-0.77778	-1	-0.30556	-0.73611	-0.34722	-0.90278	-0.33333	-0.69444	-0.30556	-0.48611
16	0	0	0	-0.36111	-0.125	-0.54167	-0.5	-0.63889	-0.79167	-0.83333	-1	-0.34722	-0.70833	-0.40278	-0.88889	-0.40278	-0.70833	-0.34722	-0.5
17	0	0	0	-0.31429	-0.12857	-0.48571	-0.51429	-0.57143	-0.81429	-0.75714	-1	-0.27143	-0.77143	-0.32857	-0.91429	-0.31429	-0.68571	-0.28571	-0.47143
18	0	0	0	-0.32877	-0.15068	-0.50685	-0.52055	-0.60274	-0.80822	-0.78082	-1	-0.28767	-0.75342	-0.32877	-0.91781	-0.32877	-0.71233	-0.28767	-0.52055
19	0	0	0	-0.32877	-0.16438	-0.49315	-0.52055	-0.57534	-0.79452	-0.75342	-1	-0.27397	-0.73973	-0.31507	-0.93151	-0.31507	-0.71233	-0.28767	-0.49315
20	0	0	0	-0.34247	-0.15068	-0.52055	-0.50685	-0.61644	-0.78082	-0.79452	-1	-0.31507	-0.72603	-0.35616	-0.90411	-0.35616	-0.69863	-0.32877	-0.49315
21	0	0	0	-0.35714	-0.14286	-0.54286	-0.5	-0.64286	-0.78571	-0.81429	-1	-0.34286	-0.72857	-0.38571	-0.91429	-0.37143	-0.7	-0.32857	-0.5
22	0	0	0	-0.33333	-0.14493	-0.50725	-0.50725	-0.5942	-0.7971	-0.76812	-1	-0.28986	-0.73913	-0.34783	-0.92754	-0.34783	-0.72464	-0.30435	-0.50725

Figure 7.8: Coordinates of the hand movement

## CONCLUSION

## CONCLUSION

The development of a Two-Way Communication System for Deaf People using Machine Learning marks a significant advancement in assistive technology, bridging the communication gap between the hearing and non-hearing communities. This project successfully integrates machine learning algorithms, computer vision (OpenCV & MediaPipe), and natural language processing (NLP) to enable seamless real-time communication.

Through the use of sign language recognition, speech-to-text conversion, and text-to-sign language animation, the system provides a robust solution for individuals with hearing impairments. By leveraging deep learning models such as CNNs, LSTMs, and Transformer-based NLP models, the system achieves high accuracy in recognizing and translating sign language gestures. The inclusion of speech recognition models like Wav2Vec ensures efficient voice-to-text conversion, further enhancing usability.

Despite the numerous benefits, challenges such as gesture variability, real-time processing speed, and dataset limitations were addressed through data augmentation, model optimization, and GPU acceleration. The system's adaptability to various devices, including mobile platforms and AR glasses, ensures greater accessibility for users. Moreover, privacy concerns were mitigated by implementing on-device processing and secure data handling techniques.

The success of this project paves the way for further improvements, such as incorporating multi-language sign recognition, real-time 3D avatars, and AI-driven contextual understanding. With continuous advancements, this system has the potential to become a standard assistive tool in education, workplaces, healthcare, and public services, significantly improving the quality of life for the deaf community.

In conclusion, this project demonstrates the power of machine learning and artificial intelligence in creating an inclusive world, ensuring that communication is no longer a barrier for the hearing-impaired population.



## **FUTURE ENHANCEMENT**

## FUTURE ENHANCEMENT

While the current system provides a robust foundation for two-way communication, several enhancements can be implemented to further improve its performance and accessibility:

1. **Multi-Language Support:** Expanding the system to recognize and translate multiple sign languages, such as ASL, BSL, and ISL, to cater to a global audience.
2. **Enhanced Gesture Recognition:** Implementing **3D hand tracking and full-body motion capture** for more accurate recognition of complex sign language expressions.
3. **AI-Driven Personalization:** Developing adaptive models that learn user-specific gestures and improve accuracy based on individual communication styles.
4. **Integration with Wearable Devices:** Extending support for **smart glasses, haptic feedback gloves, and AR-based sign language visualization** to enhance accessibility.
5. **Cloud-Based Real-Time Processing:** Leveraging cloud computing to enable **faster processing, seamless updates, and collaborative learning models** for continuous improvement.
6. **Real-Time Translation with Voice Assistants:** Integrating with **AI-powered voice assistants** like Alexa, Google Assistant, or Siri for interactive real-time conversations.
7. **Offline Mode for Remote Accessibility:** Enhancing the system to function **without an internet connection**, ensuring accessibility in remote areas.
8. **Improved Natural Language Processing (NLP):** Implementing **advanced contextual AI models** to better understand sentence structures and improve text-to-sign translations.

With these enhancements, the **Two-Way Communication System for Deaf People** can evolve into a highly sophisticated and widely adopted assistive technology, further breaking down communication barriers and fostering inclusivity.

## **BIBLIOGRAPHY**

## BIBLIOGRAPHY

### REFERENCES

- [1] Real Time Sign Language Recognition Using Deep Learning, Sanket Bankar, Tushar Kadam, Vedant Korhale, Mrs. A. A. Kulkarni, Apr 2022.
- [2] American Sign Language Recognition Using Computer Vision, Nihar Joshi, Ryan Gudal, Tianyu Jiang, Samantha Clark, 2021.
- [3] Translating Speech to Indian Sign Language Using Natural Language Processing, Purushottam Sharma, Devesh Tulsian, Chaman Verma, Pratibha Sharma, Nancy Nancy, 2022.
- [4] Real-Time Speech to Sign Language Translation Using Machine and Deep Learning, Deepak Rai, Niharika Rana, Naman Kotak, Manya Sharma, 2024.
- [5] Vision-based hand gesture recognition using deep learning for the interpretation of sign language, Sakshi Sharma, Sukhwinder Singh, 2021.
- [6] Real-Time Hand Gesture Recognition Using Fine-Tuned Convolutional Neural Network, Jaya Prakash Sahoo, Allam Jaya Prakash, Paweł Pławiak, Saunak Samantray, 2022.
- [7] A Two-Way Integrated Communication System for the Deaf and Mute, Godson Thomas, Gokul Rejithkumar, P. Sreevidya, Beenu Riju, 2022.
- [8] A Study on Communication Platforms and Services Needed and Available for Mute and Deaf Community, Snehal Patil, Yash Shah, Payal Narkhede, Abhinav Thakare, Rahul Pitale, 2021.

### BOOKS

- [9] François Chollet, *Deep Learning with Python*, Manning Publications, 2017.
- [10] Richard Szeliski, *Computer Vision: Algorithms and Applications*, Springer, 2022.
- [11] Aurélien Géron, *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow*, O'Reilly Media, 2019.
- [12] Sebastian Raschka, *Python Machine Learning*, Packt Publishing, 2019.

[13] William S. Vincent, *Django for Beginners*, Independently Published, 2018.

## WEBSITE

[14] TensorFlow Official Documentation – <https://www.tensorflow.org>

[15] OpenCV Documentation – <https://docs.opencv.org>

[16] MediaPipe Documentation (Google) – <https://developers.google.com/mediapipe>

[17] NumPy Documentation – <https://numpy.org/doc/>

[18] Django Documentation – <https://docs.djangoproject.com>