A

MAJOR PROJECT REPORT

ON

SIGN LANGUAGE RECOGNITION TO TEXT AND VOICE USING CNN

Submitted in partial fulfillment of the requirement for the award of degree of

BACHELOR OF TECHNOLOGY

IN

ELECTRONICS AND COMMUNICATION ENGINEERING

SUBMITTED BY

D.AVISHKAR	**218R1A0479**
G.SAI PRASAD	**218R1A0481**
G.PAVAN	**218R1A0482**
G.GANITHA	**218R1A0483**

Under the Esteemed Guidance of

Mr.S.SUDHAKAR

Assistant Professor

DEPARTMENT OF ELECTRONICS & COMMUNICATION ENGINEERING

CMR ENGINEERING COLLEGE

UGC AUTONOMOUS

(Approved by AICTE, Affiliated to JNTU Hyderabad, Accredited by NBA)

Kandlakoya(V), Medchal(M), Telangana – 501401

(2024-2025)

CMR ENGINEERING COLLEGE

UGC AUTONOMOUS

(Approved by AICTE, Affiliated to JNTU Hyderabad, Accreditedby NBA)
Kandlakoya (V), Medchal Road, Hyderabad - 501 401

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

CERTIFICATE

This is to certify that Major project work entitled **" SIGN LANGUAGE RECOGNITION TO TEXT AND VOICE USING CNN"** is being Submitted by **D.AVISHKAR** bearing Roll No: **218R1A0479, G.SAIPRASAD** bearing Roll No**: 218R1A0481, G.PAVAN** bearing Roll No: **218R1A0482, G.GANITHA** bearing Roll No: **218R1A0483** in B.Tech IV-I semester, Electronics and Communication Engineering is a record bonafide work carried out by then during the academic year 2024-25.

INTERNAL GUIDE:
Mr.S.SUDHAKAR

HEAD OF THE DEPARTMENT
Dr. SUMAN MISHRA

EXTERNAL EXAMINER

ACKNOWLEDGEMENT

We sincerely thank the management of our college **CMR Engineering College** for providing required facilities during our project work .We derive great pleasure in expressing our sincere gratitude to our Principal **Dr. A. S. Reddy** for his timely suggestions, which helped us to complete the project work successfully. It is the very auspicious moment we would like to express our gratitude to **Dr. SUMAN MISHRA,** Head of the Department, ECE for his consistent encouragement during the progress of this project.

We take it as a privilege to thank our project coordinator **Dr.T.SATYANARAYANA,** Professor, Department of ECE for the ideas that led to complete the project work and we also thank him for his continuous guidance, support and unfailing patience, throughout the course of this work.We sincerely thank our project internal guide **Mr.S.SUDHAKAR,** Assistant Professor of ECE for guidance and encouragement in carrying out this project work.

DECLARATION

We hereby declare that the Major project entitled **"SIGN LANGUAGE RECOGNITION TO TEXT AND VOICE USING CNN"** is the work done by us in campus at **CMR ENGINEERING COLLEGE**, Kandlakoya during the academic year 2024-2025 and is submitted as major project in partial fulfillment of the requirements for the award of degree of **BACHELOR OF TECHNOLOGY** in **ELECTRONICS AND COMMUNICATION ENGINEERING FROM JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY, HYDERABAD.**

D.AVISHKAR **(218R1A0479)**

G.SAIPRASAD **(218R1A0481)**

G.PAVAN **(218R1A0482)**

G.GANITHA **(218R1A0483)**

ABSTRACT

People with hearing impairments are found worldwide therefore, the development of effective local level sign language recognition (SLR) tools is essential. We conducted a comprehensive review of automated sign language recognition based on machine/deep learning methods and techniques published between 2014 and 2021 and concluded that the current methods require conceptual classification to interpret all available data correctly. Thus, we turned our attention to elements that are common to almost all sign language recognition methodologies. This paper discusses their relative strengths and weaknesses, and we propose a general framework for researchers. This study also indicates that input modalities bear great significance in this field; it appears that recognition based on a combination of data sources, including vision based and sensor based channels, is superior to a unimodal analysis. In addition, recent advances have allowed researchers to move from simple recognition of sign language characters and words towards the capacity to translate continuous sign language communication with minimal delay. Many of the presented models are relatively effective for a range of tasks, but none currently possess the necessary generalization potential for commercial deployment. However, the pace of research is encouraging, and further progress is expected if specific difficulties are resolved.

The prevalence of hearing impairments across the globe underscores the necessity for effective local level sign language recognition (SLR) tools that cater to diverse linguistic and cultural contexts. This paper presents a thorough review of the advancements in automated sign language recognition methodologies based on machine and deep learning techniques, focusing on literature published between 2014 and 2021. Our analysis highlights that current methodologies often lack a unified conceptual framework to accurately interpret and utilize the vast array of available data, pointing to the need for a more structured approach to SLR.

Recent advancements in the field have progressed from basic recognition of individual sign language characters and words to more complex systems capable of translating continuous sign language communication with minimal delay. This progression is largely attributed to improvements in machine learning models and data processing techniques, which have enabled more fluid and natural interaction between users and recognition systems.

CONTENTS

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

1.1 OVERVIEW OF THE PROJECT

Sign language is a vital form of communication for people with hearing impairments, and it varies from country to country. The purpose of this project is to build a system that can recognize sign language gestures and translate them into text and voice output. The system leverages Convolutional Neural Networks (CNN), a powerful machine learning model, to achieve accurate hand gesture recognition. This project can be highly beneficial in improving communication between sign language users and those who do not know sign language, by converting gestures into real time text and speech.

For millions of people, sign language communication is the primary means of interacting with the world, and it is not difficult to imagine the potential applications involving effective sign language recognition (SLR) tools. For example, we could translate broadcasts that include sign language, create devices that react to sign language commands, or even design advanced systems to assist impaired people in conducting routine jobs. In particular, deep neural networks (DNNs) have emerged as a potentially ground breaking asset for researchers, and the full impact of their application to the problem of SLR will likely be felt in the near future. SLR is a field dedicated to the automated interpretation of hand gestures and other signs used in communications between people with a speech or hearing impairment. Because hardware and software components have evolved to the point where developing advanced systems with real time translation capacities appear to be within reach, a large number of exciting and innovative solutions have been proposed and tested in recent years with the objective of building fully functional systems that can understand sign language and respond to commands given in this format. However, before any truly practical applications can be considered, it is imperative to perfect the interpretation algorithms to the point where false positives are rare. Owing to the numerous challenges inherent in this task, at this stage, it is not yet possible to design SLR tools that approach 100% accuracy on a large vocabulary. Thus, it is very important to continue developing new methods and evaluate their relative merits, gradually arriving at increasingly reliable solutions. While most researchers agree that deep learning models are the most suitable approach, the optimal network architecture remains a point of contention, with several competing designs achieving promising results. Detailed experimental evaluations are the only way to identify the best performing algorithms

using discoveries from other research teams when applicable. As most countries use their own variations of sign language, much of the research is conducted locally with persons skilled in using regional signs. With this in mind, it is not surprising that a large number of scientific papers are targeting SLR problems and that the performance level of the proposed solutions is rapidly increasing from year to year.

In the current literature, the various SLR solutions can essentially be divided into two major groups, depending on the primary data collection method. One group of methods relies on external sensors to gather insights regarding the actions of the signer, for example, through data gloves worn by the signer. Starner et al. provided early example of a system based on wearable sensors, while many other authors have exploited this concept since then. However, there are practical considerations regarding sensor based techniques, and therefore a majority of recent research has been directed toward vision based methods, which rely on images, video, and depth data to determine the semantic content of hand signs. For example, Chen et al. pioneered a hand gesture recognition method based on skin color, while many alternative techniques have since been proposed, some of which are based on filtering principles .

1.2 OBJECTIVE OF THE PROJECT

In particular, the commercial launch of the Microsoft Kinect device has unlocked a completely new level of insight, and researchers are still exploring how to leverage the power of depth vision to develop more accurate SLR tools. In terms of the type of neural network most suitable for SLR purposes, the convolutional neural network (CNN) model was one of the first to gain major attention. In addition to CNNs, other architectures such as hidden Markov models (HMMs) and recurrent neural networks (RNNs) are frequently applied. The support vector machine (SVM) model is frequently used for this purpose as well while random forest (RF) and K-nearest neighbor (KNN) are sometimes chosen for the classification task . We summarize our work contributions in this paper as follows:

1) Comprehensive review and taxonomy of automated sign language recognition (ASLR) literature: We conducted a comprehensive review of automated sign language recognition using machine/deep learning methods and techniques published between 2014 and 2021. We concluded that several SLR methods currently in existence require some conceptual classification to make sense of all available data. Thus, we focus on elements that are common to almost all sign language recognition methodologies and discuss their relative.

2) Establishment of a general framework for creating SLR models: We propose a general framework based on the challenges and limitations we have identified in the literature. At this point, the value of machine learning/deep learning (ML/DL) methodologies for sign language recognition is beyond question, although discussions regarding the most promising directions of research continue. There is consensus that deeper models hold more promise for the eventual development of real life SLR applications than traditional machine learning approaches, but at present, even the most sophisticated models fall considerably short of the necessary reliability.
3) Benchmark datasets and performance: An analysis of the benchmark datasets and performance used in the literature is conducted. The quality of available sign language datasets is essential for ensuring that SLR tools built and tested with them return relevant predictions. However, the availability of high quality datasets of this kind is limited, and in some cases barely sufficient for serious testing. Some of the datasets mentioned in literature include the Corpus VGT consisting of over 140 hours of video input and including approximately 100 classes, PHOENIX14T dataset with video recordings of 9 different signers using more than 1000 unique signs, PHOENIX Weather2014T with vocabulary related to weather, and ASLG-PC12 which includes various English language versions of signs. Datasets are usually split into training, validation, and testing portions, so the models can be evaluated with the same type of input that was used to optimize them. However, due to different datasets used in different studies, direct comparison of the results across studies is not possible.
4) Identifying open Issues and challenges: After analyzing and discussing the existing methodologies, we draw some conclusions with respect to their limitations, open issues, and potential challenges. Differences between regional variations of sign language alphabets and vocabularies greatly complicate cross border collaboration, especially considering the scarcity of high quality datasets for languages with smaller numbers of speakers. This also makes it very difficult to develop and test more advanced applications, which require much larger training vocabularies. Most of the proposed methods are conceptually sound, yet they lack the level of accuracy and reliability that would be desired for a final solution. These problems are exacerbated in the continuous SLR subfield, where semantic content is far more complex and thus more difficult to capture through statistical analysis.

The remainder of this paper is organized as follows. In Section II, we provide a brief background regarding some of the basic concepts discussed in this paper, such as deep learning, machine learning. Section III presents the review method used in this study. Machine learning and deep learning methods to design sign language recognition models are discussed in detail in Section IV along with the proposed framework. Types of models and languages related to the recognition process are discussed in Section V and Section VI, respectively. The related studies and surveys have been discussed in Section VII. Section VIII introduces the benchmark SLR datasets used for ML/DL and provides a comparative analysis of the ML/DL methods performance for sign language recognition. Section IX discusses open issues, challenges, and opportunities for future research. Finally, the conclusions of our study are presented in Section X.

1.3 ORGANIZATION OF THE PROJECT

The system can include:

- **Responsibilities**:
  - Oversee the project development.
  - Coordinate between different teams and ensure timelines are met.
  - Manage resources and budget (if applicable).
  - Regular progress reporting to stakeholders.

- **Responsibilities**:
  - **Data Collection**: Gather and organize a dataset of sign language images. This could involve sourcing existing datasets (e.g., ASL, Sign Language MNIST, RWTH-PHOENIX) or creating a custom dataset by capturing sign language gestures.
  - **Data Preprocessing**: Preprocess the images for consistency (resizing, normalization, augmentation, etc.). This step ensures that the model trains on clean and varied data.
  - **Data Augmentation**: Implement transformations like rotation, flipping, and zoom to improve model robustness.

- **Responsibilities**:
  - **Model Architecture Design**: Develop the CNN architecture for gesture recognition, selecting appropriate layers (convolutional, pooling, fully connected).

- **Model Training**: Train the CNN model on the preprocessed dataset. Optimize the model for higher accuracy using techniques like dropout, data augmentation, and hyperparameter tuning.
- **Model Evaluation**: Assess the model's performance using metrics like accuracy, precision, recall, and confusion matrix. Ensure the model generalizes well to unseen data

- **Responsibilities**:
  - Integrate a TTS library (such as `pyttsx3`, `gTTS`, or other speech synthesis tools) to convert the recognized text into spoken words.
  - Ensure real time translation of text into voice output.
  - Optimize the voice output for natural and clear speech synthesis.

- **Responsibilities**:
  - **Interface Design**: Develop a simple graphical user interface (GUI) that displays the recognized sign language text on the screen.
  - **User Interaction**: Allow the user to interact with the system, including capturing live gestures using a webcam, displaying text, and playing voice output.
  - **Real time Processing**: Optimize the system to display results in real time and handle webcam input smoothly.

- **Responsibilities**:
  - **Testing**: Perform extensive testing of the system using real world data and various environmental conditions (lighting, background noise).
  - **Bug Fixing**: Identify issues related to model accuracy, UI responsiveness, or TTS integration.
  - **System Optimization**: Ensure that the system performs well on both speed (low latency) and accuracy.

This structured approach will help ensure the successful development and deployment of a Sign Language Recognition system, leveraging CNN for gesture recognition and TTS for voice output. By organizing the project into distinct roles, phases, and milestones, the team can efficiently manage resources, adhere to timelines, and meet the objectives of the project.

CHAPTER 2

LITERATURE SURVEY

2.1 EXISTING SYSTEM

Sign Language to Text and Speech Translation in Real Time Using CNN:

Description: This system uses a webcam to capture American Sign Language (ASL) gestures, translates them into text, and then converts the text into speech.

Technology: CNN for gesture detection, followed by text-to-speech conversion.

Key Features: Real time translation, high accuracy with sufficient training.

Indian Sign Language Recognition using CNN:

Description: This project recognizes Indian Sign Language (ISL) gestures using a CNN.

Technology: CNN with TensorFlow and Keras.

Key Features: Achieved 100% accuracy on the training set and 97.5% on the test set.

Real time Recognition of German Sign Language (DGS):

Description: Uses MediaPipe for real time recognition of German Sign Language.

Technology: CNN for gesture recognition.

Key Features: Live predictions, robust to occlusion.

CONCULSION

The existing systems for sign language recognition to text and voice using Convolutional Neural Networks (CNNs) have laid a significant foundation in the field of assistive technology. These systems have demonstrated the feasibility of real time translation of sign language gestures into text and speech, enhancing communication for the hearing impaired community.

Key Achievements:

- **Real time Capability**: Existing systems successfully capture and interpret sign language gestures in real time, making live communication more accessible.
- **High Accuracy**: With advanced CNN architectures and extensive training datasets, these systems achieve high accuracy in gesture recognition.

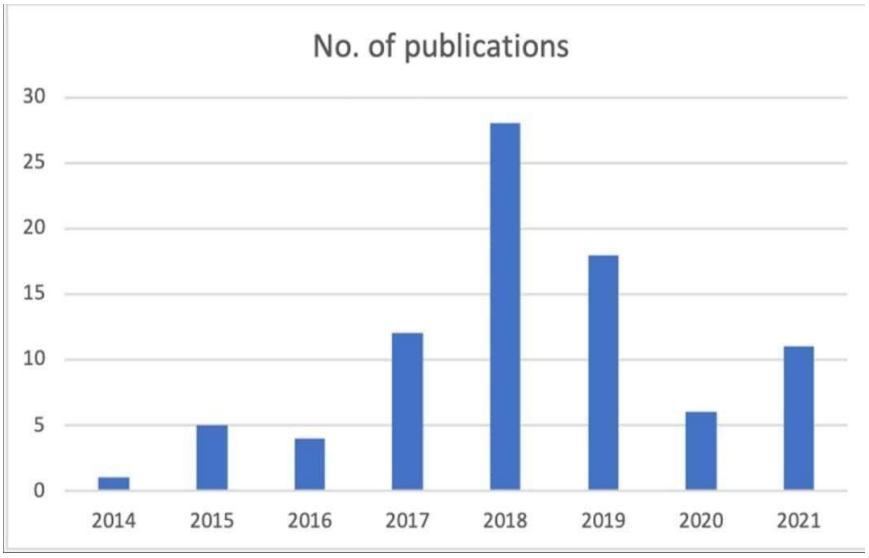


Fig:2.1 No of publication on sign language recognition by year

2.2 PROPOSED SYSTEMS

Real time Conversion of Sign Language to Text and Speech:

- **Components:**
  - Gesture Recognition Module: Utilizes CNN to detect hand signs.
  - Text and Speech Conversion Module: Converts gestures to text and then to speech.
  - Animation Gesture Module: Animates gestures for better visualization.
- **Potential Enhancements:**
  - Incorporating AI driven noise reduction to improve accuracy in noisy environments.
  - Developing a user friendly interface with customizable settings.
  - Implementing machine learning algorithms to learn user specific gestures.

Hybrid CNN LSTM Framework for Sign Language Recognition:

- **Components:**
  - CNN: Extracts spatial features from video frames.
  - Attention based LSTM: Captures temporal dependencies and improves gesture recognition.
- **Potential Enhancements:**
  - Integrating reinforcement learning to continuously improve the model.
  - Adding multi modal input support (e.g., facial expressions, body posture).
  - Creating a cloud based platform for easy access and updates.

- **Components:**
    - Video Analysis Module: Analyzes video footage without the need for markers.
    - Feature Extraction: Uses image processing and CNN to extract relevant features.
    - Text and Audio Conversion: Translates gestures to text and converts to audio.
- **Potential Enhancements:**
    - Using advanced image processing techniques to improve gesture detection.
    - Implementing a feedback loop to refine recognition accuracy.
    - Developing a comprehensive database of gestures for diverse contexts.

These systems could revolutionize communication for the hearing impaired community, making interactions more seamless and inclusive.

Conclusion

The proposed systems for sign language recognition to text and voice using Convolutional Neural Networks (CNNs) and other advanced technologies represent the next leap forward in assistive communication technologies. By integrating cutting edge methodologies and addressing current limitations, these systems aim to enhance the accuracy, efficiency, and usability of sign language recognition solutions.

Key Innovations:

- **Hybrid Models**: Combining CNNs with Long Short Term Memory (LSTM) networks and attention mechanisms enhances the system's ability to recognize and interpret complex, continuous gestures, providing a more natural and fluid user experience.
- **Real time Processing**: Advanced image processing and gesture recognition techniques ensure minimal latency, making real time translation more seamless and reliable.
- **Multi modal Inputs**: Incorporating additional inputs such as facial expressions, body posture, and contextual cues improves the overall accuracy and richness of the translation.

Advantages:

- **Personalization**: These systems can adapt to individual users' signing styles, improving accuracy and user satisfaction through machine learning and reinforcement learning techniques.
- **Scalability**: The proposed solutions are designed to support multiple sign languages and can be easily scaled to include new gestures and vocabularies, making them more versatile and globally applicable.

The proposed systems for sign language recognition to text and voice using Convolutional Neural Networks (CNNs) and other advanced technologies represent the next leap forward in assistive communication technologies. By integrating cutting edge methodologies and addressing current limitations, these systems aim to enhance the accuracy, efficiency, and usability of sign language recognition solutions.

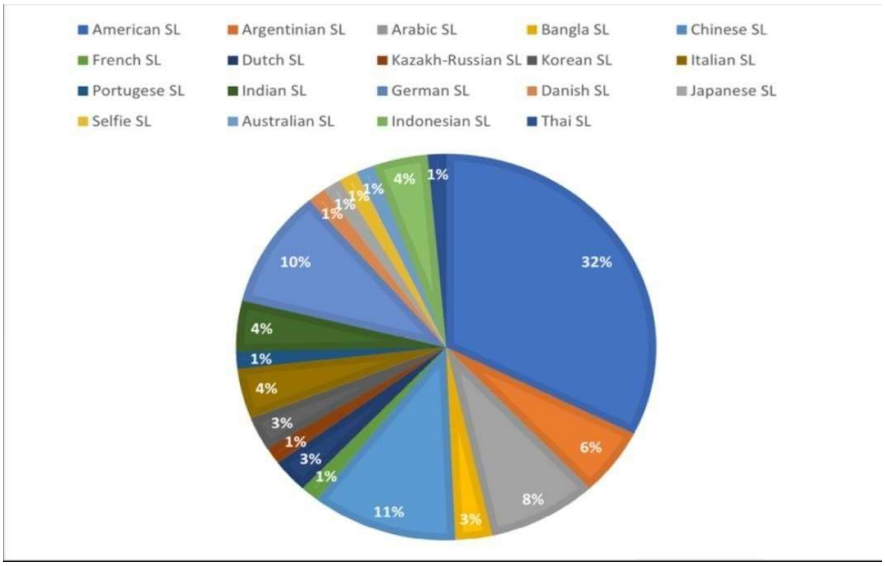


FIG: 2.2 No of publications on sign language recognition by language

2.3 CNN INTRODUCTION

Communication is a fundamental aspect of human interaction, and for the hearing impaired community, sign language serves as a crucial medium. However, the lack of widespread understanding of sign language poses a significant barrier to effective communication between the hearing impaired and those who rely on spoken language. This project aims to bridge this communication gap by developing a sophisticated system that translates sign language gestures into text and voice using Convolutional Neural Networks (CNNs).

Sign Language Recognition: The core of the project lies in the ability to accurately and efficiently recognize sign language gestures. By leveraging the power of CNNs, which are known for their excellent performance in image recognition tasks, the system captures and interprets hand gestures in real time, converting them into meaningful text and voice outputs.

Existing Systems: Several existing systems have demonstrated the feasibility of sign language recognition. These systems utilize CNNs to translate gestures into text and speech, making real

time communication more accessible for the hearing impaired. While these systems have achieved significant milestones, there are still challenges related to the accuracy, latency, and adaptability to different sign languages.

proposed system seeks to offer a more robust and scalable solution for sign language recognition.

Impact: The successful implementation of this project will have a profound impact on the hearing impaired community, enabling more seamless and inclusive communication. It will also pave the way for future advancements in assistive technologies, promoting greater accessibility and understanding in various social and professional contexts.

This project, through its integration of cutting edge technologies and a user centered approach, aspires to transform the landscape of communication for the hearing impaired, fostering a more inclusive and connected world.

Key Components

1. **Gesture Recognition Module**:
    - Utilizes CNNs to accurately identify hand gestures from video input.
    - Converts recognized gestures into corresponding text.
2. **Text-to-Speech Module**:
    - Translates the recognized text into audible speech.
    - Ensures clear and natural sounding voice output.
3. **Multi modal Input Integration**:
    - Captures additional contextual cues such as facial expressions and body posture.
    - Enhances the accuracy of gesture recognition.
4. **Real time Processing**:
    - Ensures minimal latency in gesture recognition and translation.
    - Provides immediate feedback for seamless communication.

Software Components

1. **Convolutional Neural Networks (CNN)**:
    - Core technology for gesture recognition.
    - Used for feature extraction and classification of hand gestures.
2. **Long Short Term Memory (LSTM) Networks**:
    - Enhances the handling of sequential data

4.User Interface:

- Provides a user friendly platform for interaction.
- Allows customization and personalization of settings.

System Overview

The proposed sign language recognition system consists of several interconnected modules working together to provide real time translation of sign language into text and speech. The system captures video input of hand gestures using a webcam or other recording device. The video frames are preprocessed and fed into a CNN for gesture recognition. The recognized gestures are then translated into text, which is further converted into audible speech using a text-to-speech engine. The system also integrates multi modal inputs, such as facial expressions and body posture, to enhance the accuracy and reliability of gesture recognition.

Conclusion

The development of a sophisticated system for translating sign language gestures into text and voice using Convolutional Neural Networks (CNNs) represents a significant advancement in assistive technology. This project addresses a critical need for more inclusive communication solutions, particularly for the hearing impaired community.

Key Achievements:

- **Real time Translation**: The system's ability to capture and interpret sign language gestures in real time ensures seamless and immediate communication.
- **High Accuracy**: By leveraging advanced CNN and LSTM models, the system achieves high accuracy in recognizing and translating gestures, making it a reliable tool for everyday use.

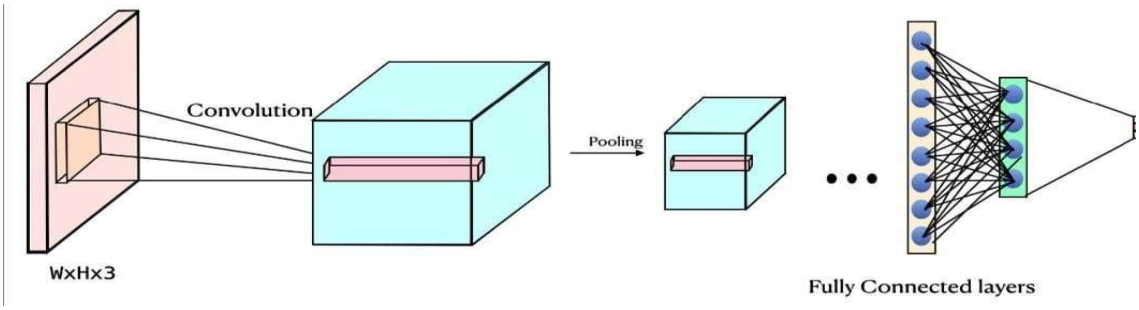


FIG: 2.3 Basic CNN model used in sign language recognition

2.4 WHY CNN ?

Using Convolutional Neural Networks (CNNs) for sign language recognition to text and voice is an effective approach because CNNs are particularly well suited for image and video processing tasks, which are essential for interpreting sign language. Here's why CNNs are commonly used in sign language recognition projects:

1. Spatial Feature Extraction:

- **Images and videos** of hand gestures, facial expressions, and body movements are the core components of sign language. CNNs are excellent at extracting spatial features from these types of data, identifying patterns and details that are critical for accurate recognition.
- CNNs can learn to detect key features in hand shapes, motion, and even the background, which are essential for interpreting gestures in sign language.

2. Invariance to Translation and Scale:

- CNNs are robust to small variations in position, scale, and rotation. This means that they can effectively recognize hand gestures regardless of slight changes in orientation or size.
- For sign language, where gestures can be performed in various spatial positions (e.g., different distances or angles from the camera), CNNs can handle this variability well
- CNNs build a hierarchical understanding of the input data. For example, in the context of sign language, the network might first detect edges and shapes (e.g., the outline of a hand), then move on to more complex features like the configuration of fingers or the overall movement of the hand.
- This hierarchical feature learning is essential to understand the nuances of sign language gestures.

3.Real Time Recognition:

- Sign language recognition systems require real time processing to allow effective communication. CNNs, especially when paired with specialized hardware like GPUs, can be optimized for fast image or video recognition, making them suitable for real time applications such as translating sign language into text or voice.

4.Real Time Recognition:

- Sign language recognition systems require real time processing to allow effective communication. CNNs, especially when paired with specialized hardware like GPUs, can be optimized for fast image or video recognition, making them suitable for real-time applications such as translating sign language into text or voice.

5.End to End Learning:

- CNNs can be trained in an end to end manner, meaning they can take raw image or video inputs and output the corresponding text or voice translation. This simplifies the system architecture and reduces the need for manual feature engineering.
- The CNN can directly map gestures to specific signs, which can be translated into either written text or spoken language (voice), enabling smoother user interaction.

6.Adaptability and Flexibility:

- CNNs are highly adaptable and can be trained on a wide variety of datasets. As new sign languages or variants emerge, CNN models can be retrained on new data, enabling flexibility and scalability in recognizing different forms of sign language

7.Multi Modal Sign Language Recognition:

- Many sign language recognition systems using CNNs are designed to incorporate not just hand gestures but also facial expressions, body posture, and movement. These systems typically combine CNNs with other types of networks (e.g., RNNs or LSTMs) to fully capture the multi modal nature of sign language.

In summary, CNNs are an ideal choice for sign language recognition because they can efficiently process complex visual data, handle spatial variations, and offer the flexibility required for real time, accurate translations of sign language into text and voice.

2.5 DESIGN APPROACHES

The design approach for developing a sign language recognition system that translates gestures into text and voice involves several critical stages. This structured approach ensures the project's success by addressing all necessary components, from data acquisition to system integration and user testing.

1. Analysis Requirements

- **Objective**: Define the system's goals, user requirements, and technical specifications.
- **Activities**:
    - Conduct interviews and surveys with potential users, including hearing impaired individuals and sign language interpreters.
    - Identify essential features such as real time processing, multi language support, and user customization options.
    - Outline technical requirements, including hardware (cameras, microphones) and software (neural networks, processing units).

2. Data Collection and Preprocessing

- **Objective**: Gather a diverse dataset of sign language gestures for training the CNN.
- **Activities**:
    - Collect video recordings of various sign languages, capturing different gestures, lighting conditions, and backgrounds.
    - Annotate the data with corresponding text labels for each gesture.
    - Perform data augmentation to increase the diversity and size of the dataset (e.g., rotating, flipping, and scaling images).
    - Preprocess the data by normalizing, resizing, and converting it into a format suitable for CNN input.

3. Model Development

- **Objective**: Design and train the Convolutional Neural Network (CNN) for gesture recognition.
- **Activities**:
    - Select an appropriate CNN architecture (e.g., ResNet, InceptionNet) based on project requirements.
    - Design the network layers, including convolutional, pooling, and fully connected layers.
    - Train the CNN using the preprocessed dataset, adjusting hyperparameters to optimize performance.
    - Evaluate the model using validation and test datasets, ensuring high accuracy and low error rates.

4. Integration of LSTM for Sequential Data

- **Objective**: Enhance the system's ability to handle continuous sign language gestures.
- **Activities**:
    - Integrate Long Short Term Memory (LSTM) networks with the CNN to capture temporal dependencies in gesture sequences.
    - Design the LSTM layers and connect them with the CNN output.
    - Train the hybrid CNN LSTM model using sequential gesture data.
    - Evaluate the model's performance on continuous sign language recognition tasks.

5. Text-to-Speech Conversion

- **Objective**: Convert recognized text into audible speech.
- **Activities**:
    - Select a text-to-speech engine that provides clear and natural-sounding voice output.
    - Integrate the text-to-speech engine with the gesture recognition module.
    - Implement customizations for different languages and voices.
    - Test the text-to-speech conversion to ensure accuracy and clarity.

6. User Interface Development

- **Objective**: Create a user friendly interface for interaction with the system.
- **Activities**:
    - Design a graphical user interface (GUI) that allows users to interact with the system easily.

7. System Integration and Testing

- **Objective**: Integrate all components into a cohesive system and conduct thorough testing.
- **Activities**:
    - Combine the gesture recognition module, text-to-speech engine, and user interface into a single system.
    - Perform unit testing on individual components to identify and fix any issues.
    - Conduct integration testing to ensure seamless interaction between all components.
    - Run user acceptance testing with real users to validate the system's performance and usability.

8. Deployment and Maintenance

- **Objective**: Deploy the system for public use and ensure ongoing maintenance.
- **Activities**:
    - Deploy the system on suitable hardware, ensuring scalability and reliability.
    - Provide user training and support to facilitate adoption.
    - Monitor system performance and collect user feedback for continuous improvement.
    - Implement regular updates and maintenance to address any emerging issues and incorporate new features.

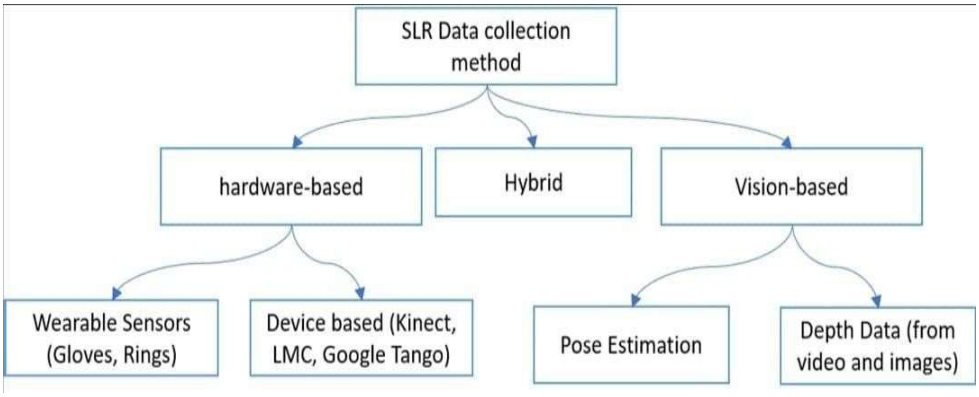


Fig: 2.5.1 Primary data collection methods employed for SLR

Applications of CNN:

In the **sign language recognition project** using **Convolutional Neural Networks (CNNs)**, the applications of CNNs are central to processing and interpreting sign language gestures, whether the goal is to convert them into text or voice. Below are some specific applications of CNNs in this context:

1. Gesture Recognition from Images and Videos

- **Hand Shape Detection**: CNNs can be trained to recognize specific hand shapes and configurations, which are crucial for interpreting sign language. By analyzing the pixels of an image, the CNN identifies the distinct features of hands (e.g., fingers, palms) and their relative positions to understand the gesture.
- **Finger Tracking**: CNNs help detect finger positions in real time, allowing the system to

recognize complex signs that require precise hand and finger placements. This can be critical for identifying subtle variations in gestures that have different meanings.

- **Gesture Segmentation**: CNNs can also segment gestures from a video stream, differentiating between one sign and another. This is especially important when recognizing dynamic gestures that involve movement over time.

2. Motion and Dynamic Gesture Analysis

- **Movement Detection**: Many sign language gestures involve dynamic movement (e.g., changing hand positions, rotations, or directional motions). CNNs, particularly when used in combination with other networks like Recurrent Neural Networks (RNNs), can recognize these movements and track the sequence of gestures over time.
- **Action Recognition**: CNNs are capable of recognizing the sequence of frames in a video and interpreting the actions performed (e.g., waving, pointing, or signing a word). This is important for understanding continuous actions or multi step gestures that make up complex signs.

3. Multimodal Sign Language Recognition

- **Facial Expression Recognition**: In many sign languages, facial expressions and lip movements are integral to conveying meaning. CNNs can be used to detect and analyze facial features to understand the nuances of a sign, especially for conveying emotions or questions (e.g., raising eyebrows to indicate a question).
- **Posture and Body Language Recognition**: In addition to hands and faces, body posture plays an essential role in certain signs. CNNs can analyze the overall posture or positioning of the body to help decode the full meaning of a sign language gesture.

4. Real Time Translation (Sign to Text or Voice)

- **Hand Gesture to Text Conversion**: CNNs are trained to recognize and classify static or dynamic hand gestures and translate them directly into written text. For instance, if a person signs the word "hello," the CNN can recognize the sign and output "hello" in text.
- **Sign Language to Voice Translation**: For spoken language output, CNNs can be part of a pipeline where the recognized sign is then converted into an audio response using text-to-speech (TTS) systems. CNNs help in identifying which word or phrase the gesture represents, and TTS systems then generate the corresponding voice output.

5. Sign Language Dataset Learning

- **Data Augmentation for Robustness**: CNNs are often used to train on large datasets of hand gestures. These datasets can be enhanced through data augmentation (e.g., rotating, scaling, or flipping images) to make the model more robust and accurate when faced with variations in gestures due to lighting, angle, or individual differences in signing.
- **Cross Lingual Sign Language Recognition**: CNNs can be trained on datasets containing signs from different regional or national sign languages (e.g., American Sign Language, British Sign Language). The models can be adapted to recognize specific signs unique to each language, potentially even offering real time translation between sign languages.

6. Real Time Feedback for Sign Language Learning

- **Interactive Learning Systems**: CNN based recognition systems can be incorporated into interactive applications designed to teach sign language. Users can practice sign language gestures, and the system can provide real time feedback on the accuracy of their hand movements. This helps individuals learn sign language more efficiently and provides them with visual or auditory guidance.

7. Object Detection and Contextual Sign Language Interpretation

- **Context Awareness**: In more advanced applications, CNNs can be used in combination with object detection models to provide contextual understanding. For example, when a sign language gesture involves pointing to an object or referring to a location in space, CNNs can be used to detect and interpret these contextual cues, making the translation more accurate.
- **Contextual Translation**: CNNs, when trained on large, diverse datasets, can also help with interpreting contextual nuances in signs. For instance, some signs may have different meanings based on the context in which they are used, and CNNs can assist in understanding these subtle differences.

8. Real Time Communication in Virtual Environments

- **Virtual Sign Language Interpreters**: CNNs can be applied to virtual assistants or avatars in virtual environments (such as VR or AR) that act as sign language interpreters. These avatars can recognize sign language gestures from users and respond by translating them into voice or text within the virtual environment who are deaf or hard of hearing.

9. Integration with Wearables and Smart Devices

- **Smart Glasses and Wearables**: CNNs can be used in wearable devices (such as smart glasses) that track hand gestures or monitor sign language communication in real time. These devices can then translate the gestures into text or voice, offering a portable and immediate solution for sign language recognition.
- **Smart Home Integration**: Integrating CNNs with smart home devices (e.g., voice assistants, smart TVs, or lighting systems) can help deaf individuals communicate with their home environment through sign language. CNNs enable the recognition of commands expressed in sign language, allowing users to control devices or obtain information without relying on voice.

10. Security and Surveillance Applications

- **Surveillance Systems**: CNNs can be deployed in security or surveillance systems to detect and interpret sign language in public places or emergency situations. For example, a surveillance camera could recognize a sign of distress (e.g., someone signaling for help) and automatically alert authorities or send the interpretation to security personnel.
- **Emergency Response**: In emergency situations, CNNs can enable first responders to understand sign language gestures from individuals in distres.

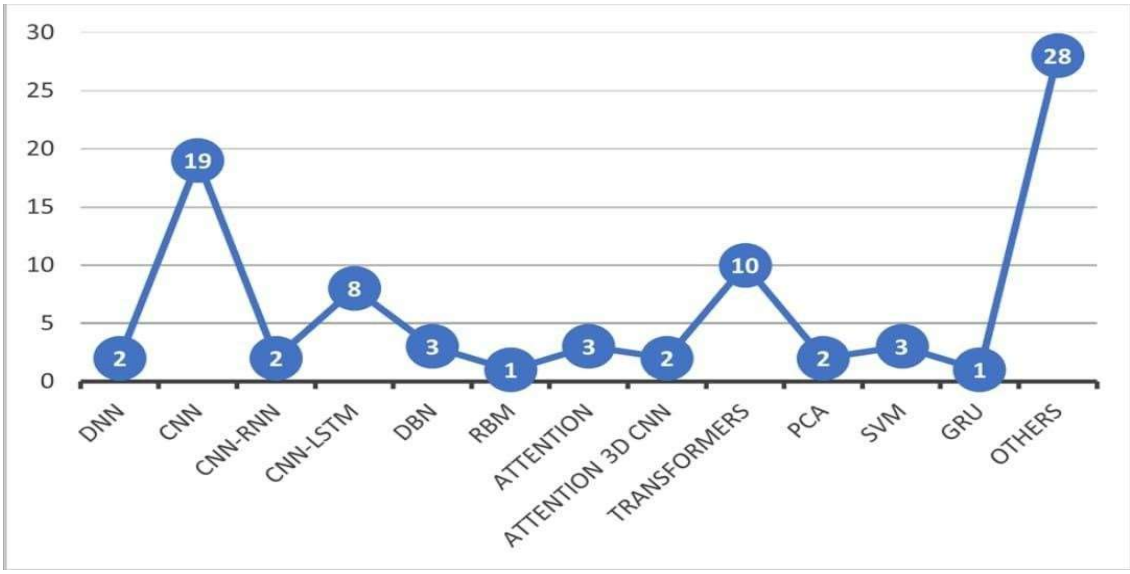


Fig: 2.5.2 No of publications on sign language recognition by architecture

2.6 SIGN LANGUAGE MODELING AND RECOGNITION

Sign language modeling focuses on developing an articulate model from the phonetic to the semantic level for language representation. The modeling process covers various aspects, ranging from the use of the signing space to the synchronization of manual and non manual features such as eye gaze and facial expressions. On the other hand, sign language recognition entails pattern matching, computer vision, linguistics, and other aspects of natural language processing . The objective of sign language recognition is to establish different methods and algorithms that can recognize already developed signs and perceive their meaning. The techniques for sign language modeling and recognition discussed in this section include classic, deep learning, SLR continuous models, and SLR isolated models.

A Convolutional Neural Network receives an input image, assigns significance to different aspects of the image, and differentiates one image from another. Figure 7 shows the basic CNN architecture mode for sign language recognition. CNNs require a much lower level of preprocessing compared to other deep learning algorithms .While these networks perform strongly in many tasks ,they require large amounts of labeled training data . Hand shape recognition, a process influenced by the pose of the subject, has a remarkably high rate of intraclass ambiguity, which results in an added burden to acquire training data. In many cases, only a few specific labeled datasets exist in the gesture and sign language recognition field. As such, CNN has been used because it can be trained easily. In a CNN was embedded within an iterative expectation maximization (EM) algorithm, which allowed the CNN to be trained using a very large number of model images. The CNN achieved a recognition accuracy of 62.8% on over 3000 hand shape images that were labeled manually.

Some experiments focus on American Sign Language. The methodology applied an end to end CNN architecture to a training dataset for comparison purposes. Additionally, CNN and SVM were combined to act as a feature descriptor, producing acceptable accuracy. Another related experiment by Li et al. used CNN to process images of a large size. The CNN shares the weights of the images, thereby significantly reducing the number of parameters that need to be learned. This also reduces the risk of overfitting. CNN also finds invariant features that are particularly useful during image processing. By combining CNN with various PCA layers,developed a hierarchical model that proved useful for recognizing fingerspelling in American Sign Language. Similarly, the authors in developed a CNN focused on grouping fingerspelling images using a mixture of image intensity and depth data.

CHAPTER 3

SOFTWARE REQUIREMENTS

Developing a sign language recognition system to text and voice using Convolutional Neural Networks (CNNs) requires a comprehensive set of software requirements to ensure functionality, performance, and usability. Here are the primary software requirements:

1. **Operating System**:
    - **Requirement**: The system should be compatible with major operating systems such as Windows, macOS, and Linux.
2. **Programming Languages**:
    - **Requirement**: The primary programming language should be Python, due to its extensive libraries and frameworks for machine learning and image processing.
3. **Integrated Development Environment (IDE)**:
    - **Requirement**: An IDE like PyCharm, Visual Studio Code, or Jupyter Notebook to facilitate development and debugging.
4. **Machine Learning Frameworks**:
    - **Requirement**: Use TensorFlow or PyTorch for developing and training the CNN and LSTM models.
5. **Image Processing Libraries**:
    - **Requirement**: Utilize OpenCV for video capture, image processing, and data augmentation.
6. **Data Management**:
    - **Requirement**: Implement a robust data management system to handle the large datasets of sign language gestures, including data storage, retrieval, and preprocessing.
    - **Tools**: SQL/NoSQL databases, Pandas for data manipulation.
7. **Text-to-Speech (TTS) Engine**:
    - **Requirement**: Integrate a TTS engine to convert recognized text into natural sounding speech.
    - **Tools**: Google Text-to-Speech, eSpeak, or Microsoft Azure Cognitive Services.
8. **User Interface (UI)**:
    - **Requirement**: Develop an intuitive and accessible graphical user interface.
9. **APIs and Integration**:
    - **Requirement**: Use APIs for integrating different modules and services.
    - **Tools**: RESTful APIs, WebSockets for real time data transmission.

10. **Real time Processing**:
    - **Requirement**: Ensure the system supports real time processing with minimal latency.
    - **Tools**: Multithreading and multiprocessing libraries in Python, like concurrent.futures or multiprocessing.
11. **Testing and Validation**:
    - **Requirement**: Implement comprehensive testing frameworks to ensure the accuracy and reliability of the system.
    - **Tools**: PyTest, Unittest, or other relevant testing frameworks.
12. **Documentation and Version Control**:
    - **Requirement**: Maintain thorough documentation and version control for code and models.
    - **Tools**: Git for version control, Sphinx or MkDocs for documentation.

3.1 SOFTWARE TOOLS

TensorFlow:

- A powerful open source library for machine learning and deep learning, widely used for building and training neural networks.

PyTorch:

- An alternative deep learning framework that provides flexibility and ease of use for building neural network models.

OpenCV:

- A comprehensive library for computer vision tasks, including image processing, video capture, and data augmentation.

Numpy:

- A fundamental package for scientific computing with Python, essential for handling arrays and performing numerical operations

Google Text-to-Speech:

- A TTS API that converts text into natural sounding speech using Google's machine learning models.

Microsoft Azure Cognitive Services:

- A suite of AI services that includes speech recognition and text-to-speech capabilities.

Flask/Django:

- Web frameworks for developing web based user interfaces, offering flexibility and scalability for the application.

Tkinter:

- A standard GUI toolkit for Python, suitable for developing basic desktop applications.

Git:

- A distributed version control system for tracking changes in source code and collaborating with other developers.

Jupyter Notebook:

- An open source web application that allows you to create and share documents containing live code, equations, visualizations, and narrative text.

PyTest/Unittest:

- Testing frameworks for Python, essential for writing and running tests to ensure code quality and reliability.

3.2 RESEARCH

1. Current State of Technology

- **Literature Review**:
  - Review existing research papers, articles, and case studies on sign language recognition.
  - Focus on previous implementations using Convolutional Neural Networks (CNNs), Long Short Term Memory (LSTM) networks, and hybrid models.
  - Identify the strengths and limitations of these existing systems.
- **Commercial Solutions**:
  - Study commercial products and services that offer sign language recognition.
  - Analyze their features, accuracy, user feedback, and market adoption.

2. Algorithm Selection and Development

- **Convolutional Neural Networks (CNNs)**:
  - Research the most effective CNN architectures for image recognition, such as ResNet, InceptionNet, and VGGNet.

- **Long Short Term Memory (LSTM) Networks**:
    - Investigate the integration of LSTMs for handling sequential data in sign language recognition.
    - Explore attention mechanisms and their benefits for improving recognition accuracy.
- **Hybrid Models**:
    - Examine the effectiveness of combining CNNs with LSTMs or other sequential models.
    - Evaluate the improvements in accuracy and performance achieved by hybrid models.

3. Datasets and Data Augmentation

- **Sign Language Datasets**:
    - Identify publicly available datasets for different sign languages (e.g., ASL, ISL, DGS).
    - Evaluate the size, diversity, and quality of these datasets.
    - Explore collaborations with institutions or organizations to obtain more comprehensive datasets.
- **Data Augmentation Techniques**:
    - Research various data augmentation methods to enhance the training dataset.
    - Techniques may include rotation, scaling, flipping, and adding noise to images.
    - Assess the impact of augmented data on model performance.

4. User Needs and Requirements

- **User Surveys and Interviews**:
    - Conduct surveys and interviews with hearing impaired individuals and sign language interpreters.
    - Gather insights on their needs, preferences, and challenges.

- **Accessibility and Usability**:
    - Research best practices for designing accessible and user friendly interfaces.
    - Study the principles of Universal Design to ensure the system is inclusive for all users.

5. System Integration and Real Time Processing

- **Embedded Systems and Hardware**:
    - Research the integration of sign language recognition systems with wearable devices, mobile phones, and smart home devices.
    - Evaluate the hardware requirements for real time processing and minimal latency.
- **Real Time Processing Algorithms**:
    - Investigate algorithms that enable real time processing of sign language gestures.
    - Focus on optimizing processing speed without compromising accuracy.

6. Challenges and Mitigation Strategies

- **Gesture Variability**:
    - Study the variability in sign language gestures among different users.
    - Research methods to handle variations in hand shape, speed, and orientation.
- **Environmental Factors**:
    - Examine the impact of different lighting conditions and backgrounds on gesture recognition.
    - Develop strategies to mitigate these environmental challenge.

3.3 How to Install Python on Windows and Mac :

There have been several updates in the Python version over the years. The question is how to install Python? It might be confusing for the beginner who is willing to start learning Python but this tutorial will solve your query. The latest or the newest version of Python is version 3.7.4 or in other words, it is Python 3.

Note: The python version 3.7.4 cannot be used on Windows XP or earlier devices.

Before you start with the installation process of Python. First, you need to know about your **System Requirements**. Based on your system type i.e. operating system and based processor, you must download the python version. My system type is a **Windows 64 bit operating system**. So the steps below are to install python version 3.7.4 on Windows 7 device or to install Python 3. Download the Python Cheatsheet here.The steps on how to install Python on Windows 10, 8 and 7 are **divided into 4 parts** to help understand better.

Download the Correct version into the system.

Step 1: Go to the official site to download and install python using Google Chrome or any other web browser. OR Click on the following link: https://www.python.org.

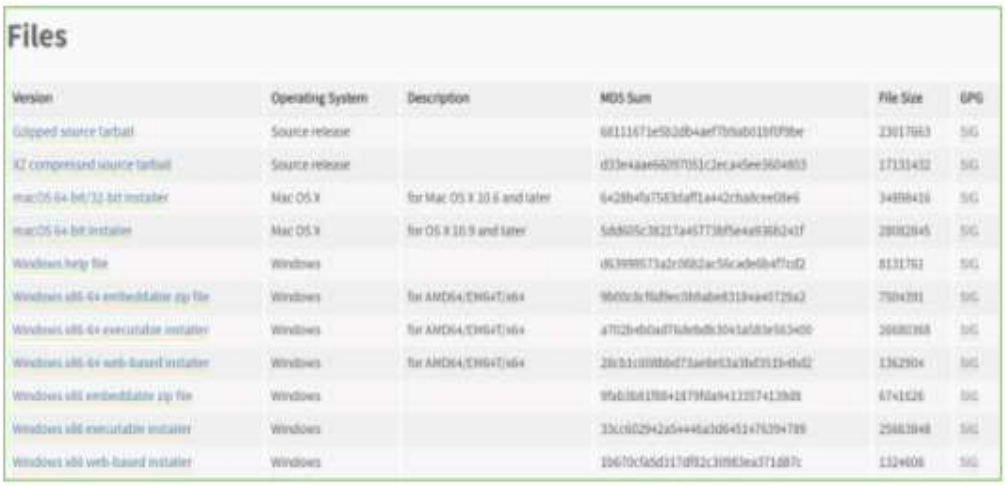
Now, check for the latest and the correct version for your operating system.

Step 2: Click on the Download Tab.

Step 3: You can either select the Download Python for windows 3.7.4 button in Yellow Color or you can scroll further down and click on download with respective to their version. Here, we are downloading the most recent python version for windows 3.7.4.

Step 4: Scroll down the page until you find the Files option.

Step 5: Here you see a different version of python along with the operating system.



• To download Windows 32 bit python, you can select any one from the three options: Windows x86 embeddable zip file, Windows x86 executable installer or Windows x86 web based installer.

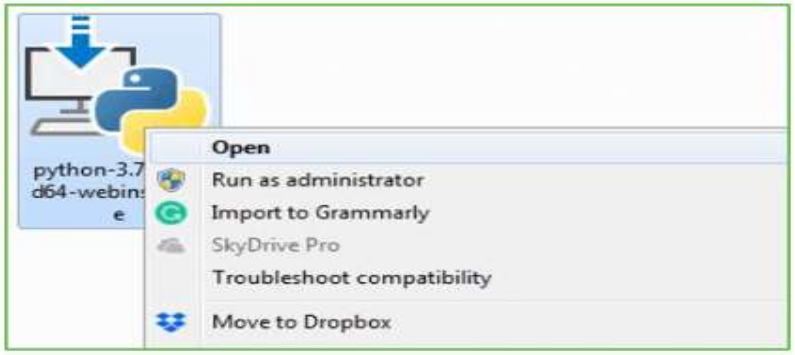
•To download Windows 64 bit python, you can select any one from the three options: Windows x86 64 embeddable zip file, Windows x86 64 executable installer or Windows x86 64 web based installer.

Here we will install Windows x86 64 web based installer. Here your first part regarding which version of python is to be downloaded is completed. Now we move ahead with the second part in installing python i.e. Installation

Note: To know the changes or updates that are made in the version you can click on the Release Note Option.

Installation of Python

Step 1: Go to Download and Open the downloaded python version to carry out the installation.



Step 2: Before you click on Install Now, Make sure to put a tick on Add Python 3.7 to PATH.

Step 3: Click on Install NOW After the installation is successful. Click on Close.

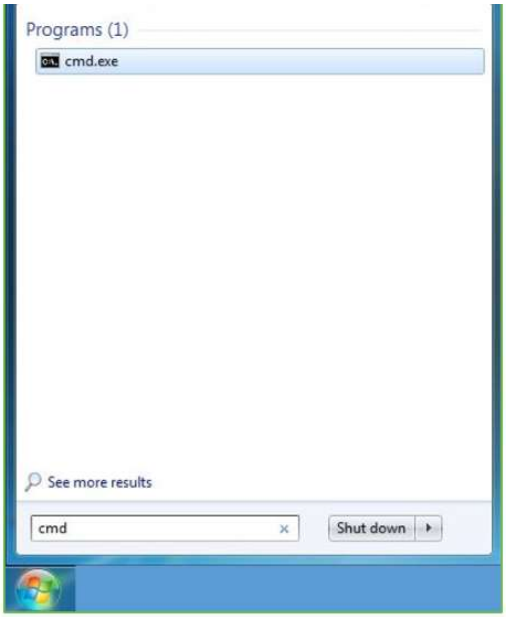
With these above three steps on python installation, you have successfully and correctly installed Python. Now is the time to verify the installation.

Note: The installation process might take a couple of minutes.

Verify the Python Installation

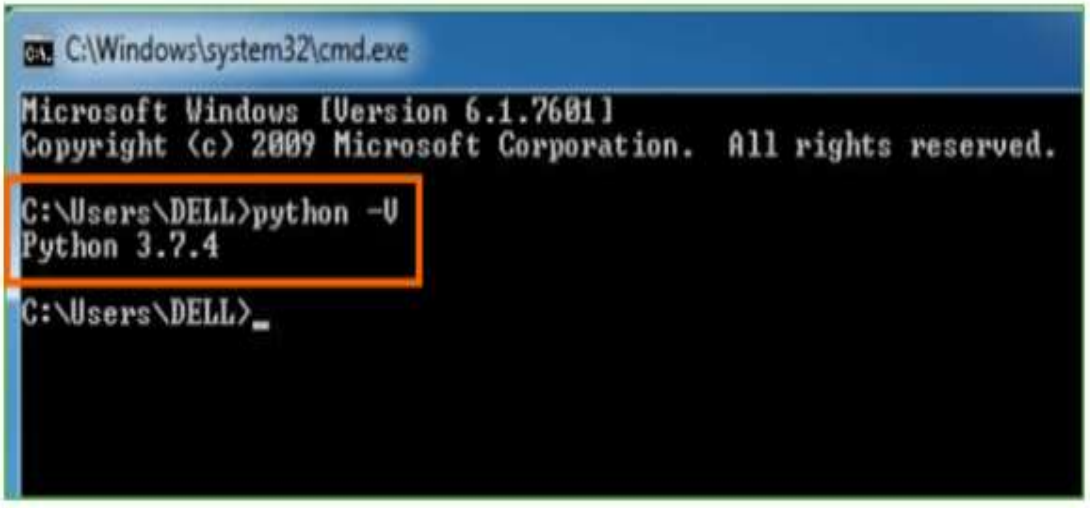
Step 1: Click on Start

Step 2: In the Windows Run Command, type “cmd”.



Step 3: Open the Command prompt option.

Step 4: Let us test whether the python is correctly installed. Type **python –V** and press Enter.



Step 5: You will get the answer as 3.7.4

Note: If you have any of the earlier versions of Python already installed. You must first uninstall the earlier version and then install the new one.

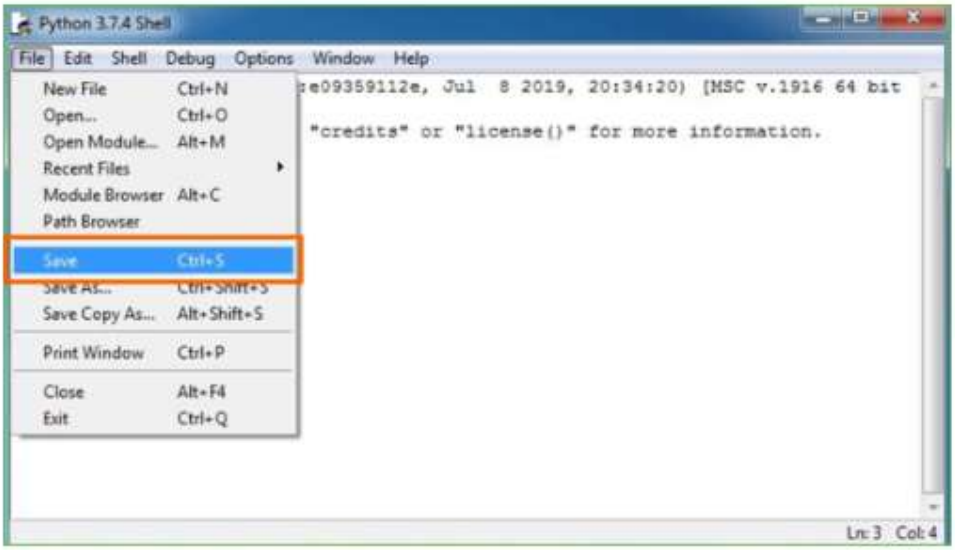
Check how the Python IDLE works

Step 1: Click on Start

Step 2: In the Windows Run command, type “python idle”.

Step 3: Click on IDLE (Python 3.7 64 bit) and launch the program

Step 4: To go ahead with working in IDLE you must first save the file. **Click on File > Click on Save**



Step 5: Name the file and save as type should be Python files. Click on SAVE. Here I have named the files as Hey World.

Step 6: Now for e.g. **enter print.**

Modules Used in Project :

Tensorflow

TensorFlow is a free and open source across software library for dataflow and differentiable programming a range of tasks. It is a symbolic math library, and is also used fo applications such as neural networks. It is used for both research and production at Google.

TensorFlow was developed by the Google Brain team for internal Google use. It was released under the Apache 2.0 open source license on November 9, 2015.

Numpy

Numpy is a general purpose array processing package. It provides a high performance multidimensional array object, and tools for working with these arrays.

It is the fundamental package for scientific computing with Python. It contains various features including these important ones:

- A powerful N dimensional array object
- Sophisticated (broadcasting) functions
- Tools for integrating C/C++ and Fortran code
- Useful linear algebra, Fourier transform, and random number capabilities

Besides its obvious scientific uses, Numpy can also be used as an efficient multi dimensional container of generic data. Arbitrary data types can be defined using Numpy which allows Numpy to seamlessly and speedily integrate with a wide variety of databases.

Pandas

Pandas is an open source Python Library providing high performance data manipulation and analysis tool using its powerful data structures. Python was majorly used for data munging and preparation. It had very little contribution towards data analysis. Pandas solved this Problem. Using Pandas, we can accomplish five typical steps in the processing and analysis of data, regardless of the origin of data load, prepare, manipulate, model, and analyze.Python with Pandas is used in a wide range of fields including academic and commercial domains including finance, economics, Statistics, analytics, etc.

Matplotlib

Matplotlib is a Python 2D plotting library which produces publication quality figures in a variety of hardcopy formats and interactive environments across platforms. Matplotlib can be used in Python scripts, the Python and IPython shells, the Jupyter Notebook, web application servers, and four graphical user interface toolkits. Matplotlib tries to make easy things easy and hard things possible. You can generate plots, histograms, power spectra, bar charts, error charts, scatter plots, etc., with just a few lines of code.For simple plotting the pyplot module provides a MATLAB like interface, particularly when combined with IPython.

For the power user, you have full control of line styles, font properties, axes properties, etc, via an object oriented interface or via a set of functions familiar to MATLAB users.

Scikit – learn

Scikit learn provides a range of supervised and unsupervised learning algorithms via a consistent interface in Python. It is licensed under a permissive simplified BSD license and is distributed under many Linux distributions, encouraging academic and commercial use.

Python

Python is an interpreted high level programming language for general purpose programming. Created by Guido van Rossum and first released in 1991, Python has a design philosophy that emphasizes code readability, notably using significant whitespace.

Python features a dynamic type system and automatic memory management. It supports multiple programming paradigms, including object oriented, imperative, functional and procedural, and has a large and comprehensive standard library.

- Python is Interpreted − Python is processed at runtime by the interpreter. You do not need to compile your program before executing it. This is similar to PERL and PHP.
- Python is Interactive − you can actually sit at a Python prompt and interact with the interpreter directly to write your programs.

Maintainability also ties into this may be an all but useless metric, but it does say something about how much code you have to scan, read and/or understand to troubleshoot problems or tweak behaviors. This speed of development, the ease with which a programmer of other languages can pick up basic Python skills and the huge standard library is key to another area where Python excels. All its tools have been quick to implement, saved a lot of time, and several of them have later been patched and updated by people with no Python background without breaking.

CHAPTER 4

HARDWARE REQUIREMENTS

4.1HARDWARE

Most automated SLR research is concerned with similar problems, namely the need to interpret hand and body movements associated with sign language characters in a clear and unambiguous manner. Because the main objectives are similar, the studies in this area also share similar methodology, even if their procedures may not be identical. Figure 5 presents the general model shared among the majority of researchers in this area. The input layer of the solution consists of an input device based on SLR data collection methods, as shown in Figure 6, and includes a visual display to present hand signs. The second layer is the preprocessing layer that performs gesture data filtering and can decode a sign into the required data format. In some cases, there are additional steps, such as sample normalization or merging information contained in successive frames of a video. The first procedure performed by the system after receiving sign data is feature extraction. All proposed methods have to provide solutions for the two most important tasks: extraction of relevant features, and classification of entries to determine the most likely sign being presented.

There are many different types of features that can be used as the primary source of information, such as visual features, hand movement features, 3D skeletal features, and facial features, among others. The selection of features to be included for algorithm training is one of the most important factors that determine the success of the SLR method. The data are typically processed and transformed into a vector format before being input to the modeling layer, and multiple channels may be fused together to analyze their joint contribution to sign recognition.

Hardware based approaches are designed to circumvent computer vision problems during sign language recognition. These challenges may develop when recognizing signs from a video, for example. In many cases, hardware based approaches use devices or wearable sensors. Wearable devices used in sign language recognition often use sensors attached to the user or implement a glove based approach. These devices (whether sensors, gloves, or rings) can convert sign language into text or speech. With respect to wearable sensors and devices, the authors in describe how they capture depth and intensity images

obtained from a Microsoft Kinect sensor and a SOFT KINECT sensor. A similar category of observations features direct measurement methods that involve the use of sensors attached to the hands or body, as well as motion capture systems . Huang et al. observed that sensor based approaches are never natural because burdensome instruments must be worn. Instead, they propose a novel approach, Real Sense, which can detect and track hand locations naturally.

Gloves equipped with digital capture capacities were introduced and utilized to deduct hand signs of the Arabic sign language variation with a reduced number of sensors. While the cost of creating and using special equipment of this kind is considerable, it is still many times cheaper than purchasing some of the high tech products available in the market. Authors chose motion controller as the primary input device, which allowed them to track objects in three dimensions with extreme precision at a 120 fps rate. The controller they used was developed for the purpose of tracking hand motion, so the researchers were able to follow many key points on the hands from one frame to the next. The same device was used by to differentiate between 50 unique isolated hand signs, with absolute precision attained.

In recent years, research on sign language recognition systems has focused more on vision based methods because they provide little to no restraints on users, unlike sensor based approaches. In vision based techniques, depth and pose estimation data are collected from users. A discussion regarding depth data and pose estimation can be found in section V. some of the recent SLR studies rely on input in the visual format. For example, depth information and RBG are some of the formats that can be commonly encountered in this field as demonstrated by . Previous research by Rioux Maldague and Giguere indicates that use of depth data has increased because of the increased number of 3D sensors available in the market.

A Microsoft Kinect sensor was used in their experiment, which has an image resolution of 640 × 480 and uses a traditional intensity camera to obtain depth images. Recent publications have also obtained depth data using vision based approaches .Depth data can be in the form of video sequences or images obtained using a normal camera or a mobile device. Oyedotun and Khashman used hand gesture grayscale images measuring 248 × 256 pixels.

According to Zheng et al.The use of depth data is advantageous to maintain privacy and to streamline the human body extraction process.

Furthermore, depth data are invariant to changes in illumination, hair, clothing, skin, and background .A side from depth data, pose estimation has been used to facilitate vision based techniques. Rioux Maldague and Giguere used a combination of regular intensity images and depth images to group different hand poses. They tracked the hands using functions that are publicly available in the OpenNI+NITE framework. While using pose estimation, computationally heavy heat maps for 2D joint locations were generated, and a 3D hand pose was inferred based on inverse kinematics and depth channels. Koller et al. further described the state of the art aspect of hand shape recognition, where the configuration of a hand pose is determined by the positions and angles of the joints. Currently, many experiments use these joint positions and angles because they can be estimated based on depth images and pixel wise hand segmentation. Other experiments, such as those by Zimmermann and Brox use a hand pose estimation system combined with a classifier trained to recognize hand gestures.

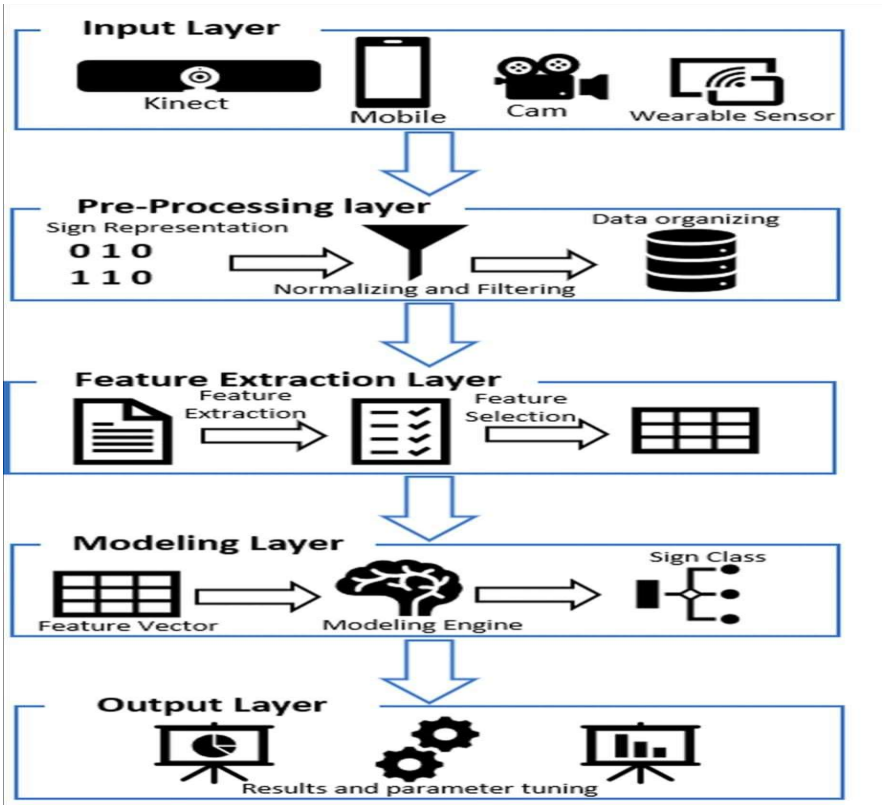


Fig: 4.1 Automated sign language recognition framework

Using still photos or continuous recordings in RGB format has the advantage of good resolution, but depth imaging does a better job at determining how far an item might be located from a fixed point. There are certain algorithms that use both types of visual data in combination . Thermal imaging is also an intriguing possibility, even if it is used more rarely than the previous two formats. IR heat sensors can leverage the emitting of radio waves or reflection of light rays to construct an image as well. This type of information has been used with success for tasks such as facial recognition or body contouring, but has not yet found its way into SLR studies . Skeletal data can also be used as a source of input, mostly in the form of hand joint position during SLR gestures. Another type of input is derived from motion capture, where information changes are tracked from one image to another. Models of this kind usually define the optical sequence as a vector describing the movement of pixels in series of still images, while so called scene sequence can be tracked in video materials referring to the motion of three dimensional objects within the scene, relative to the distance from the camera lens.

In some instances, hybrid approaches have been used to collect sign language recognition data. Hybrid methods exhibit similar or better performance compared to other methods with respect to proportional automatic speech or handwriting recognition. In hybrid approaches, vision based cameras together with other types of sensors, such as infrared depth sensors, are combined to acquire multi mode information regarding the shapes of the hands . This approach requires calibration between the hardware and vision based modalities, which can be particularly challenging. The fact that this method does not require retraining means that it is faster and can be used to examine the impact of deep learning techniques. Koller et al. conducted an experiment and opted for the cleaner hybrid method, otherwise referred to as automatic speech recognition (ASR), to examine the direct impact of this type of data on a CNN.

While all of the input devices can be effective in the right scenario, their performance significantly fluctuates depending on the context. Still, more advanced input sources such as depth sensors and Real Sense/Kinect recording systems can create three dimensional representations which carry far more information than simple two dimensional images from a fixed angle.

4.2 SIGN REPRESENTATION

Sign language is a type of visual language that utilizes grammatically structured manual and non manual sign representations to facilitate the communication process. These represents may range from the hand shape to the orientation of the palm, finger or hand movement and location, as well as head tilting, mouthing, and other aspects of facial expression. Tang et al. used eight representative frames organized in a time sequence. Their representations showed the movement of two hands that began by moving closer to each other before moving apart. In all gestures used in an experiment were represented by the hand of the signer. A hand segmentation phase was also used to represent the shape of the hand sign. Similarly, Koller et al. represented 60 hand shape classes using a double state, while the garbage class was represented by a single state.

Another experiment by Zhou et al. evaluated only right handed signers. In this case, the right hand was used to represent the dominant hand, while the left hand was the submissive hand. Hossen et al. focused on the Bengali Sign Language, having 51 letters that were represented in the experiment using 38 signs, which were developed by combining related sound alphabets into single signs.

In the Bahasa Indonesia Language, one word is represented by at most five signs, as discussed in .This means that every word and affix has an independent signed Indonesian (SIBI) representation and is represented by one sign that is consistently performed. Another experiment by Huang et al. used 66 input units and 26 output units to represent 26 signs.

Past experiments have also attempted to compare body and hand features. In it was observed that body features make up a somewhat better representation compared to hand features for sign language recognition. In essence, using body features improved the recognition of sign language by 2.27% . These observations can be attributed to the fact that body joints are more dependable and robust than hand joints.

4.3 NORMALIZATION AND FILTERATING

In machine learning and deep learning, normalization refers to all actions and procedures aimed at standardizing the input based on a set of predefined rules with the ultimate objective to improve performance of the AI tool. This procedure is typically performed during the data preprocessing stage, and may include various statistical operations or media processing tasks. The exact type of normalization procedure that is optimal for the current implementation depends on the format of the input (i.e. text, image, or video), the level of variability within the sample, the type of machine learning architecture, the purpose of the automation tool, etc. Due to its impact on performance, normalization is commonly included in most contemporary Sign Language Recognition methodologies and its contributions are empirically verified .As SLR studies use a lot of different input modalities and pursue a range of different objectives, it is logical that the scope of normalization techniques found in this field is quite broad. Most of the techniques are visual in nature, and involve changes of images to fit them into a standard format that can be readily interpreted by the algorithm. This is frequently done by altering the data on the level of pixels, since this is how information is encoded in the machine learning models during the feature extraction and network training stages.

Some of the simplest examples of normalization methods used in SLR are image resizing and reshaping, as demonstrated in Kratimenos et al and several other works .Garurel et al. also normalize the size of each frame to fit feature map dimensions, using mean values and standard deviations obtained during training to find the most optimal size. Cropping is another frequently used method that can improve the quality of visual data and make sign recognition more reliable by removing sources of possible confusion for the algorithm. Input images are typically cropped in such a way to eliminate all regions except those depicting hands and face, which are crucial for sign language communication. In cropped images are normalized based on the average length of the neck, thus negating the impact of the distance from camera for every image. In a benchmark signer is selected and input from other signers is standardized based on positions of key joints. Contour extraction is used to this end as well, for example in with the main focus on the areas corresponding to hands, with background removed from the image. For SLR methods that rely primarily on video for raw input, frame downsampling is frequently used to standardize the quality of various clips and reduce computational demands.

In normalization and filtering processes were applied. The intensity histogram of an image was equalized and all pixels were normalized to the [0, 1] interval. Gabor filters were then applied to the processed images using four different scales and orientations. An attempt was made to apply bar filters to the depth and intensity images to obtain the primary contours of the hands. Gabor filters were also used in an experiment by Li et al.

To obtain hand features that could be used for classification. While using Gabor filters, images were normalized to a size of 96 × 96 pixels. In another experiment in, principal component analysis (PCA) filter convolutions learned from input images were used. As part of the preprocessing, Koller et al applied a perpixel

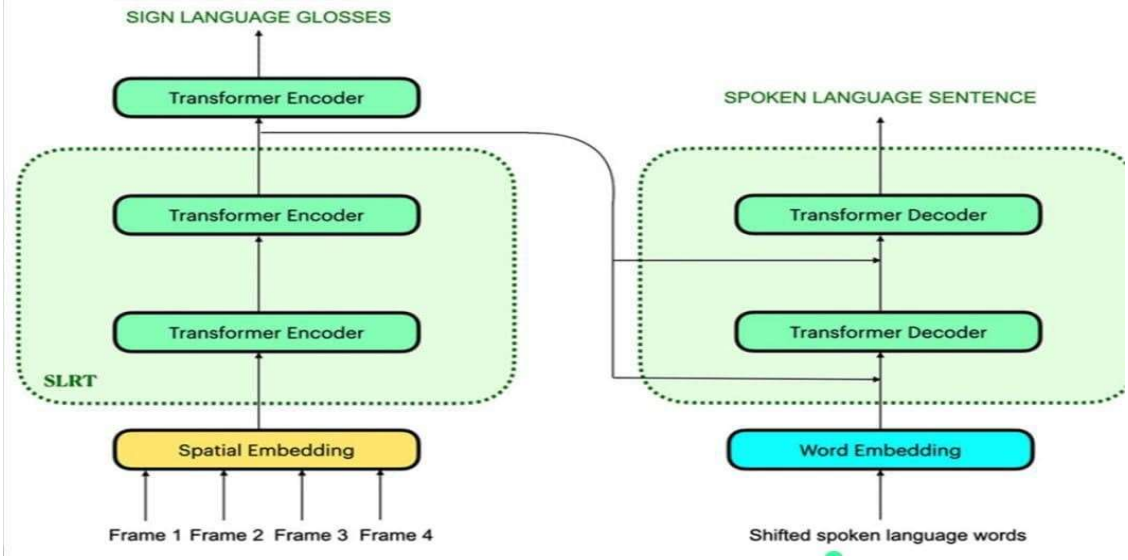


Fig: 4.3.1 Transformer based SLR main architecture

In one hybrid model, a separate neural network of the CNN type is used to extract the features from video input, greatly improving the efficiency of the process. The classification step is typically performed by a Bi directional Long Short Term Memory (Bi LSTM) module or an encoder decoder stack, comprising several successive layers in both cases. The exact depth of the model and the number of deployed attention heads varies depending on the intended use of the model and other factors, and can be optimized for best performance based on the empirical evaluations. For example, some studies propose using only two layers in transformer models (as opposed to standard six used for Natural Language Processing), while others introduce a linear projection layer and a soft attention layer on top of the stack.

In general, the performance of the suggested models is typically evaluated in terms of capacity for correct execution of the primary task, i.e., sign language recognition or translation. Average accuracy for the entire dataset is given as the main indicator of model performance, with a higher percentage indicative of a more accurate system. In some cases, top 1, top 5, and top 10 accuracy were calculated, expressing the model's ability to identify 'most likely' candidates rather than one correct answer.

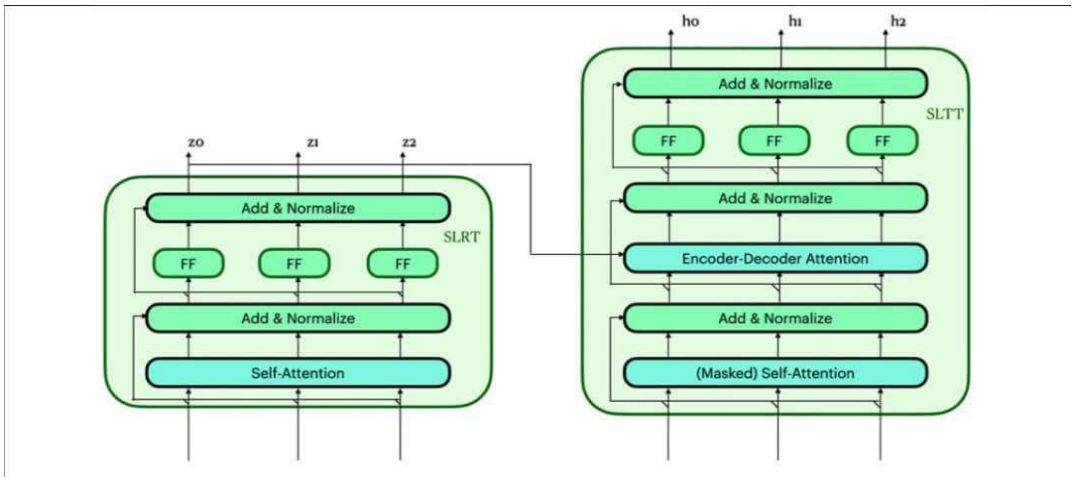


Fig: 4.3.2 A detailed overview of a single lavered Sign language transformer

A BLUE score was used to assess the quantitative output of translation models with values between 0 and 100 as depicted in Table 15, while qualitative analysis was based on comparison with ground truth as interpreted by human operators. A combination of accuracy and training sample size is used to construct the learning curve, which demonstrates how the performance changes as the volume of training sample increases. Word error rate (WER) analysis is conducted in some studies, as shown in Table 16, to determine which glosses are most confused with each other.

To be effective, the neural classifier must first be trained on data resembling the samples it will encounter during testing and/or practical use. The training data usually involve a basic group of sign language characters, words, or sentences presented in a format that the system was designed to decipher, which is typically annotated by human observers. After training is conducted, the model can be used to deduce the sign language elements in the same format with varying degrees of accuracy.

Reference(s)	#Input Modality
[197]	2D and 3D skeletal representations, depth data
[64]	RGB
[126]	RGB, RGB-D Kinect
[44]	intensity camera , Kinect
[39]	RGB image, Kinect
[76]	RGBD
[65]	RGB Video
[3]	RGB
[129]	6D IMU data
[116]	RGB
[117]	RGB
[134]	RGB videos
[48]	RGB, Kinect

Table1 :Datasets for sign language based on hand gesture linguistic content

In some studies, several classifiers were tested on the same tasks to evaluate their relative strengths and weaknesses, while in others the focus was on discovering the most suitable combinations of features. The capability of the neural model is generally limited to the signs learned from the training set, but some generalization regarding different people displaying the same sign can be achieved. Therefore, the optimization of training parameters is one of the most important elements of SLR research and can have tremendous impact on the utility value of the proposed solutions. More advanced systems aim to develop real time translation capacity and to interpret more complex segments of continuous sign language speech. Such applications are vastly more complex than simple recognition of alphabetic characters or isolated words, and they frequently have to analyze multiple signs together to understand the meaning behind a given sequence. In response, researchers have to deploy hybrid architectures intended to capture semantic nuances and avoid confusing similar sign.

CHAPTER 5

5.1 FLOW CHART

This flowchart illustrates the step by step process of building a speech recognition system using the CMU Sphinx toolkit. The process begins with Data Preparation, where the input audio and transcription data are organized. It proceeds to Language Model creation, which defines the probabilities of word sequences, followed by Dictionary Preparation that maps words to their phonetic representations. The Acoustic Model is then trained to associate audio signals with phonetic units.

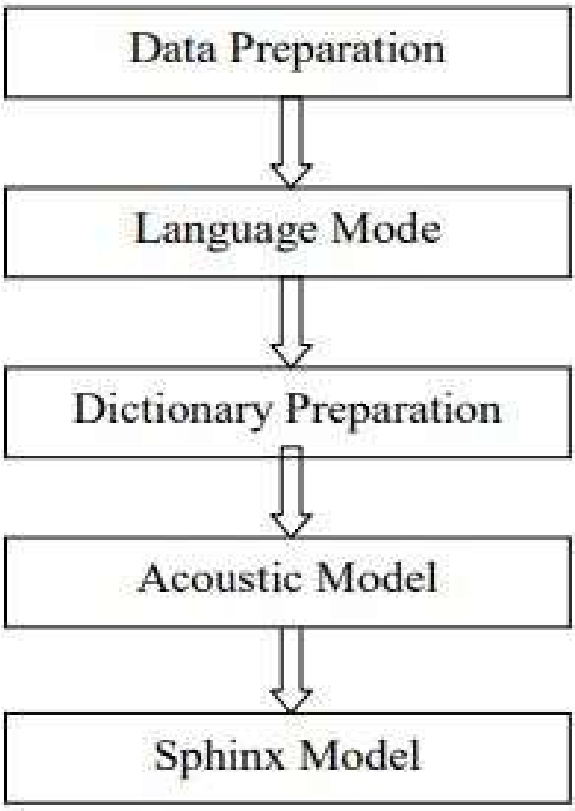


Fig 5.1 Data Flow

DATA PREPARATION :

The corpus used for the system is the publicly available corpus. It contains a total of 1000 sentences about general information. The system is trained with 1000 sentences and tested 150 sentences.

LANGUAGE MODE :

A Language model comprises of a large set of words together with its chances of occurrence. The model extracts the number of unigram bigram and trigrams of the corpus and calculates the probability of each unigram bigram and trigram. These statistical results are used by the decoder to predict the possible combination of words and phrases. It helps to achieve faster execution and higher accuracy by constraining the search in a decoder by limiting the number of possible words that need to be considered during the search.

DICTIONARY PREPARATION :

Dictionary provides the data to map vocabulary words to sequence of phonemes to the system. Uses Letter only phone names without special symbols which simplifies the system. Dictionary should contain all the words needed to be recognized by the recognizer.

ACOUSTIC MODEL :

Acoustic model is a file which contains statistical representation of each of individual sounds that make up a word. An acoustic model is created from a speech corpus using training algorithms. In Sphinx it is done using Sphinx train module. This part gives the output in the form of a configuration file. The parameters written in configuration file are used by the decoder to generate the acoustic model for a given language.

SPHINX TRAIN (Open source toolkit for speech recognition) **:**

Training is performed when there is need to create an Acoustic model for a new language. Knowledge on the phonetic structure of the language should be there to perform the training. Once the training is done it creates the database and by running the sphinx train the speech recognition files can be created.

TRAINING ALGORITHM :

Acoustic model is a file which contains statistical representation of each of individual sounds that make up a word. An acoustic model is created from a speech corpus using training algorithms. In Sphinx it is done using Sphinx train module. This part gives the output in the form of a configuration file.

5.2 UML DIAGRAMS

UML stands for Unified Modeling Language. UML is a standardized general purpose modeling language in the field of object oriented software engineering. The standard is managed, and was created by, the Object Management Group. The goal is for UML to become a common language for creating models of object oriented computer software. In its current form UML is comprised of two major components: a Meta model and a notation. In the future, some form of method or process may also be added to; or associated with, UML.The Unified Modeling Language is a standard language for specifying, Visualization, Constructing and documenting the artifacts of software system, as well as for business modeling and other non software systems.

UML supports modeling of both structural aspects (such as class diagrams, object diagrams, and component diagrams) and behavioral aspects (such as use case diagrams, sequence diagrams, activity diagrams, and state diagrams) of a system. These diagrams help in different stages of software development, from requirements gathering to implementation and testing.

The UML represents a collection of best engineering practices that have proven successful in the modeling of large and complex systems.The UML is a very important part of developing objects oriented software and the software development process. The UML uses mostly graphical notations to express the design of software projects.

GOALS:

The Primary goals in the design of the UML are as follows:

1. Provide users a ready to use, expressive visual modeling Language so that they can develop and exchange meaningful models.
2. Provide extendibility and specialization mechanisms to extend the core concepts.
3. Be independent of particular programming languages and development process.
4. Provide a formal.
5. basis for understanding the modeling language.
6. Encourage the growth of OO tools market.
7. Support higher level development concepts such as collaborations, frameworks, patterns and components.
8. Integrate best practices.

5.2.1 USE CASE DIAGRAM:

A use case diagram in the Unified Modeling Language (UML) is a type of behavioral diagram defined by and created from a Use case analysis. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases. The main purpose of a use case diagram is to show what system functions are performed for which actor.

The primary objective of a use case diagram is to capture the functional requirements of a system by outlining what actions or services the system must support, based on user needs. Rather than focusing on how the system performs internally, it emphasizes what the system should do from an external perspective. This makes it especially useful for stakeholders and non technical users to understand system behavior without diving into technical details.

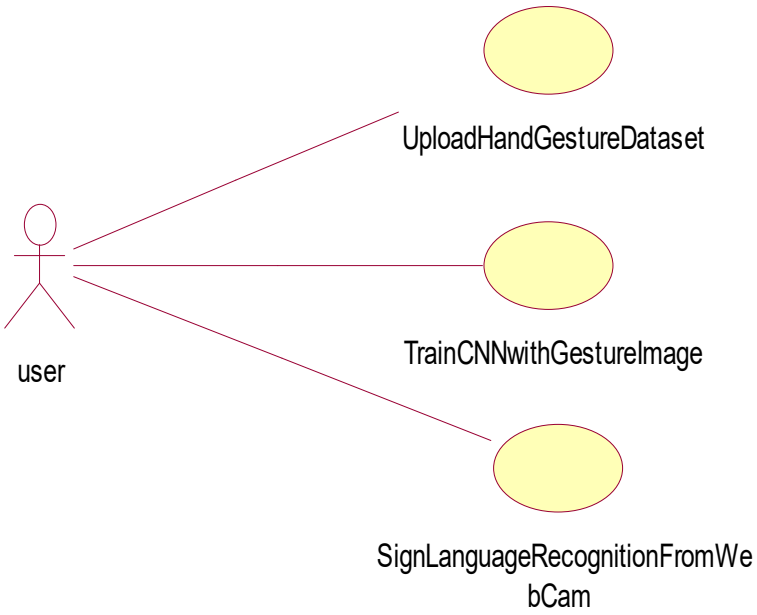


Fig 5.2.1.Use Case Diagram

5.2.2 CLASS DIAGRAM:

In software engineering, a class diagram in the Unified Modeling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among the classes. It explains which class contains information.

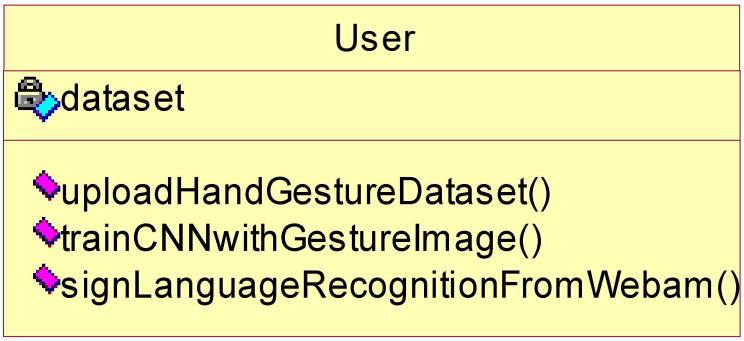


Fig 5.2.2. CLASS DIAGRAM

5.2.3 SEQUENCE DIAGRAM:

A sequence diagram in Unified Modeling Language (UML) is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of a Message Sequence Chart. Sequence diagrams are sometimes called event diagrams, event scenarios, and timing diagrams.

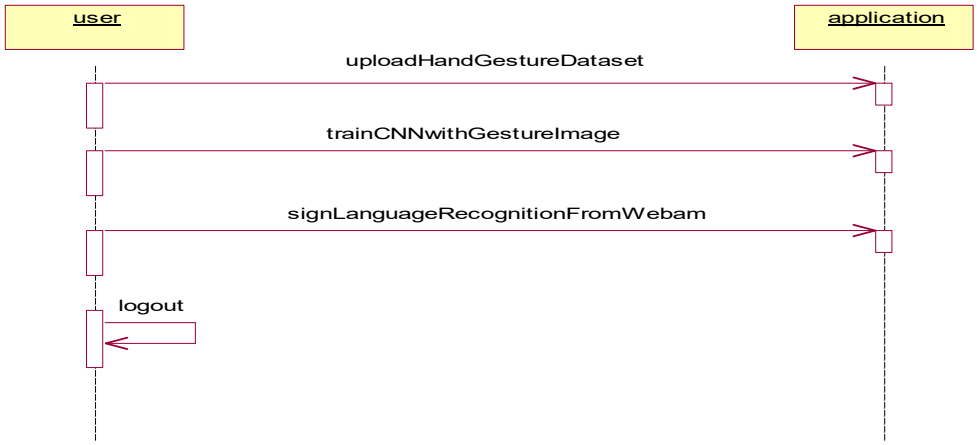


Fig 5.2.3. SEQUENCE DIAGRAM

5.2.4 COLLABORATION DIAGRAM:

This collaboration diagram represents the interaction between a user and an application for hand gesture recognition. It shows the sequence of operations: uploading a dataset, training a CNN, recognizing sign language via webcam, and logging out. collaboration diagram provides a clear visualization of how the system components interact with one another in a coordinated manner to achieve the primary goal of sign language recognition through hand gestures.

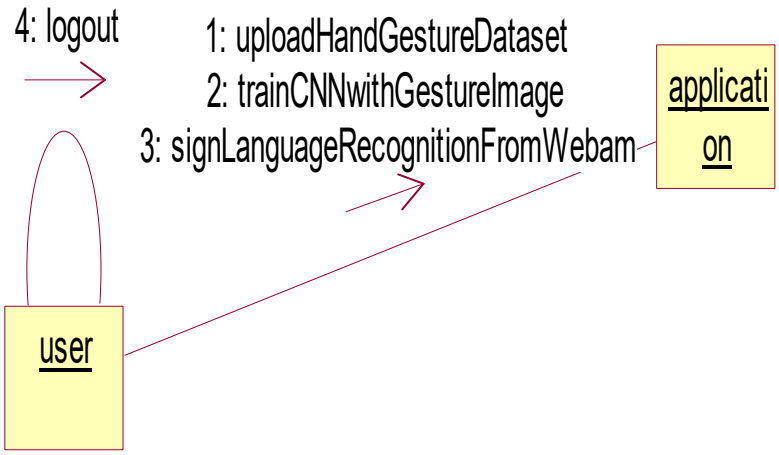


Fig 5.2.4. COLLABORATION DIAGRAM

5.3 IMPLEMENTATION:

MODULES:

1. Upload Dataset:
In this module, the user is required to upload a dataset containing prelabeled sign language images or gestures. This dataset is crucial for training the model to recognize and convert text or audio inputs into the corresponding sign language. The dataset must be clean, well structured, and categorized to ensure accurate recognition. It serves as the foundation for the system's learning process and overall performance.

2. Register:
This module allows new users to create an account by filling in personal details such as name, email, contact number, and a secure password. Registration ensures that only authorized users can access the system and helps in maintaining user specific records, preferences, or saved outputs. It also sets up a user friendly environment for interaction.

3. Login:
After registration, users can log into the system using their registered username and password. This module ensures secure access and prevents unauthorized use of the application. On successful login, users are redirected to the dashboard where they can begin using the system functionalities.

4. Give Input:
In this core module, users can provide input either in text format (typed words or sentences) or audio format (spoken language through a microphone). If the input is in audio form, it is first converted into text using speech to text processing. The text is then interpreted and prepared for sign language conversion. This module acts as the central interaction point between the user and the recognition engine.

5. Output:
Based on the input provided, this module generates the corresponding sign language output, usually in the form of images, animations, or gesture sequences. These signs represent the meaning of the input text or audio. The output helps deaf or mute users visually understand what has been spoken or typed, thus facilitating clear and meaningful communication.

CHAPTER 6

RESULTS:

The implementation of the speech to text system demonstrated promising results in translating audio input into accurate textual output. Testing was conducted on a variety of datasets, including randomly selected sentences and structured sentence sets, both in clean and noisy environments. The system performed significantly better on randomly chosen sentences, indicating higher robustness and adaptability in unpredictable speech patterns. Additionally, the use of an optimized language model played a crucial role in enhancing recognition accuracy, especially when dealing with incomplete words or unclear pronunciations. Despite occasional challenges with background noise, the system consistently delivered reliable results, validating the effectiveness of the chosen approach for medium to large vocabulary conversions.

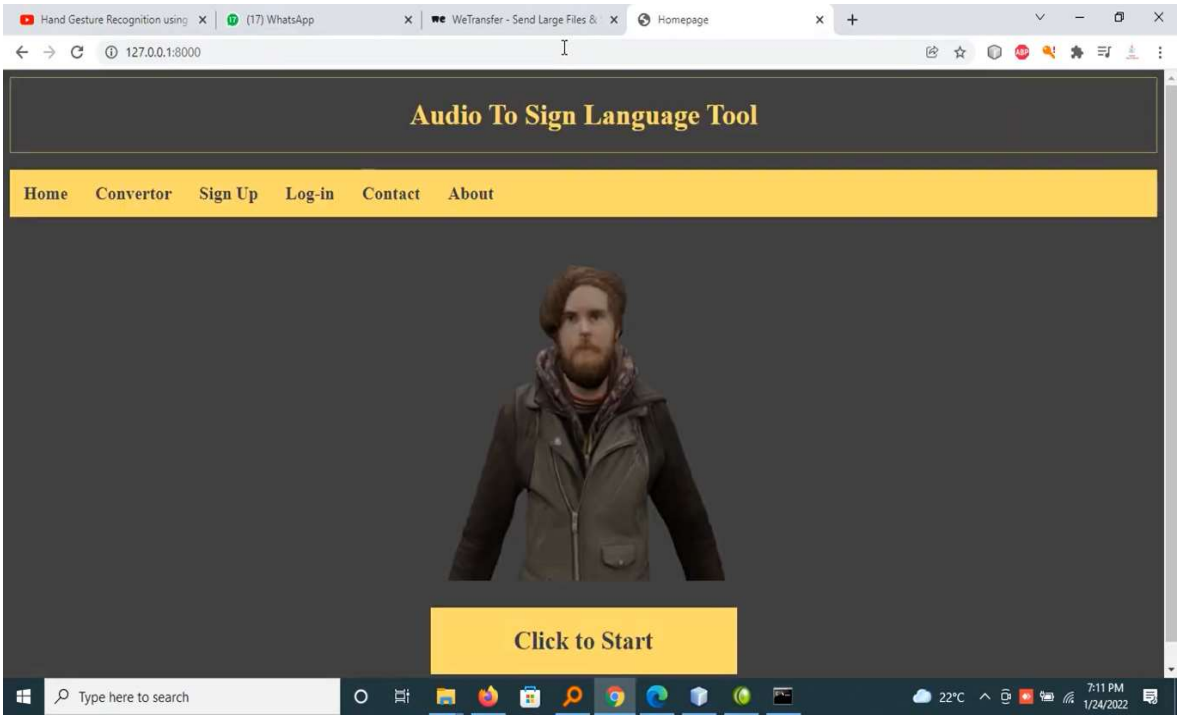


Fig 6.1. Home page

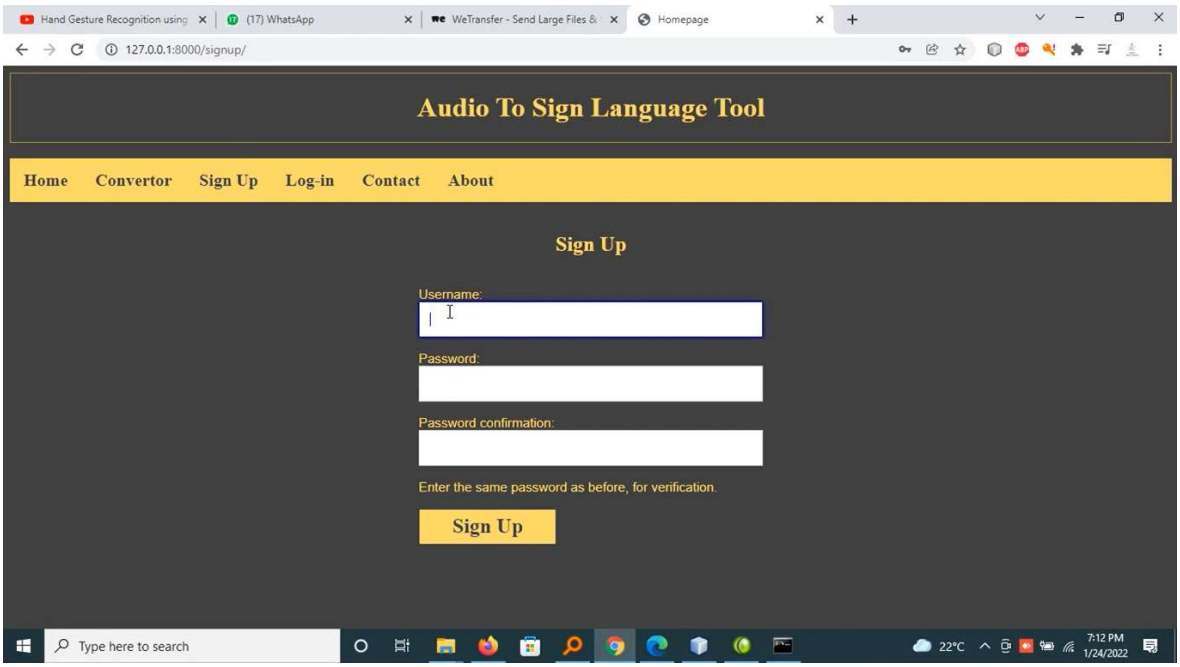


Fig 6.2. User registration

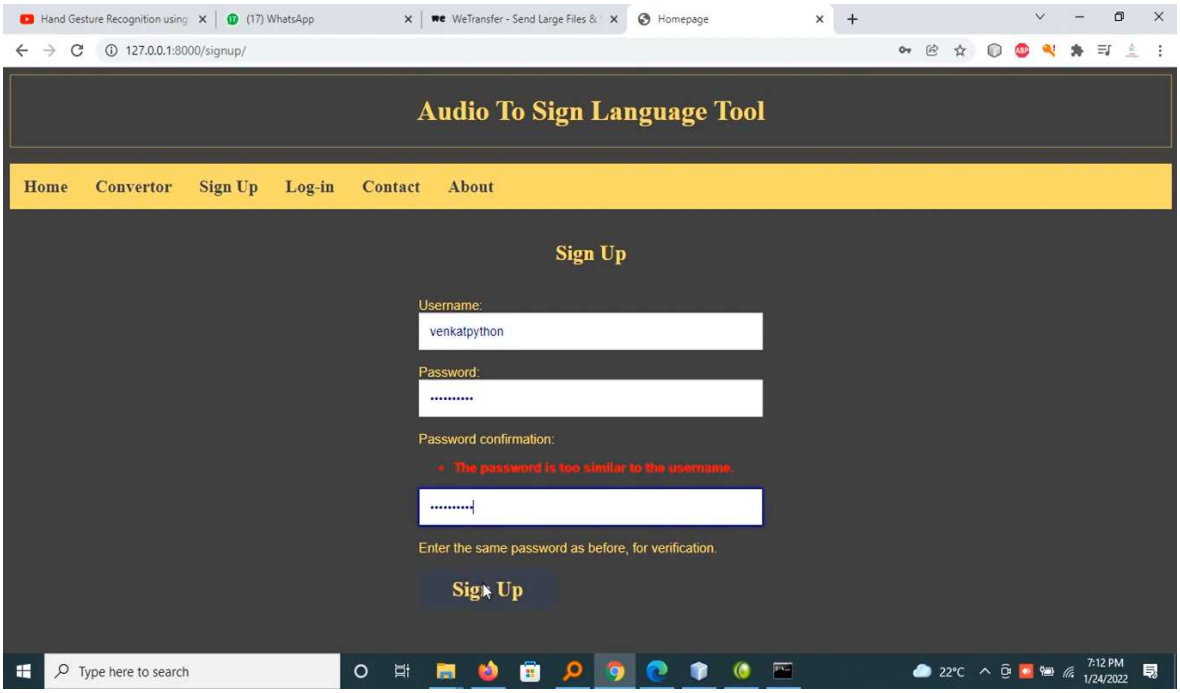


Fig 6.3. Sign Up Page

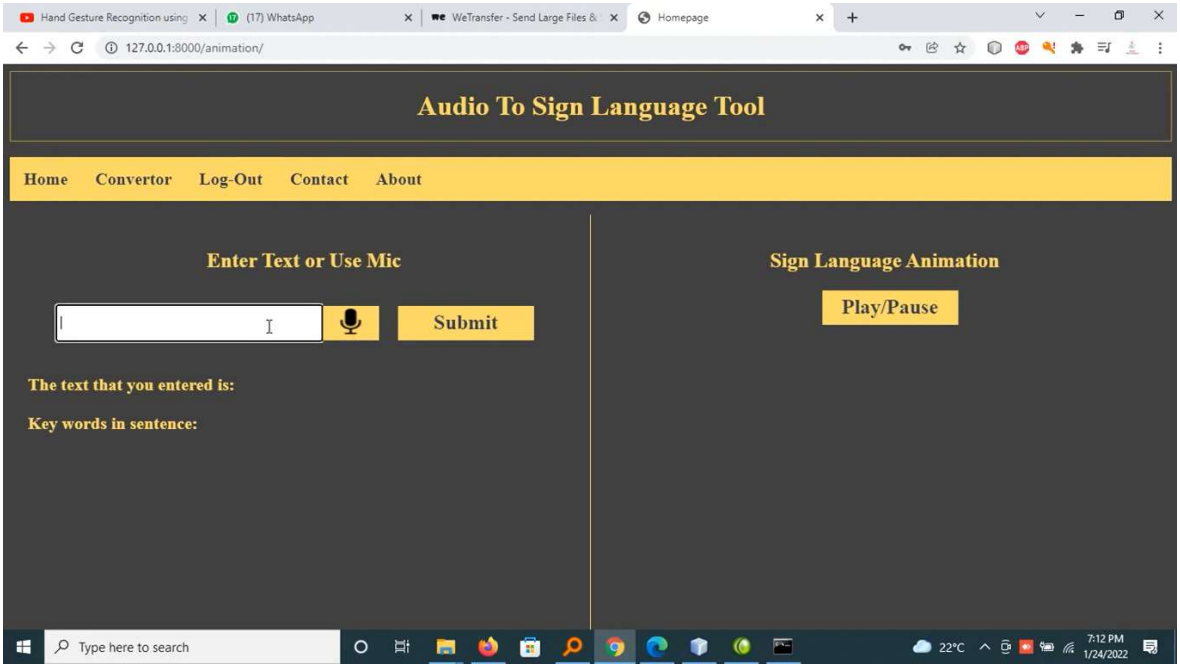


Fig 6.4. Convertor Page

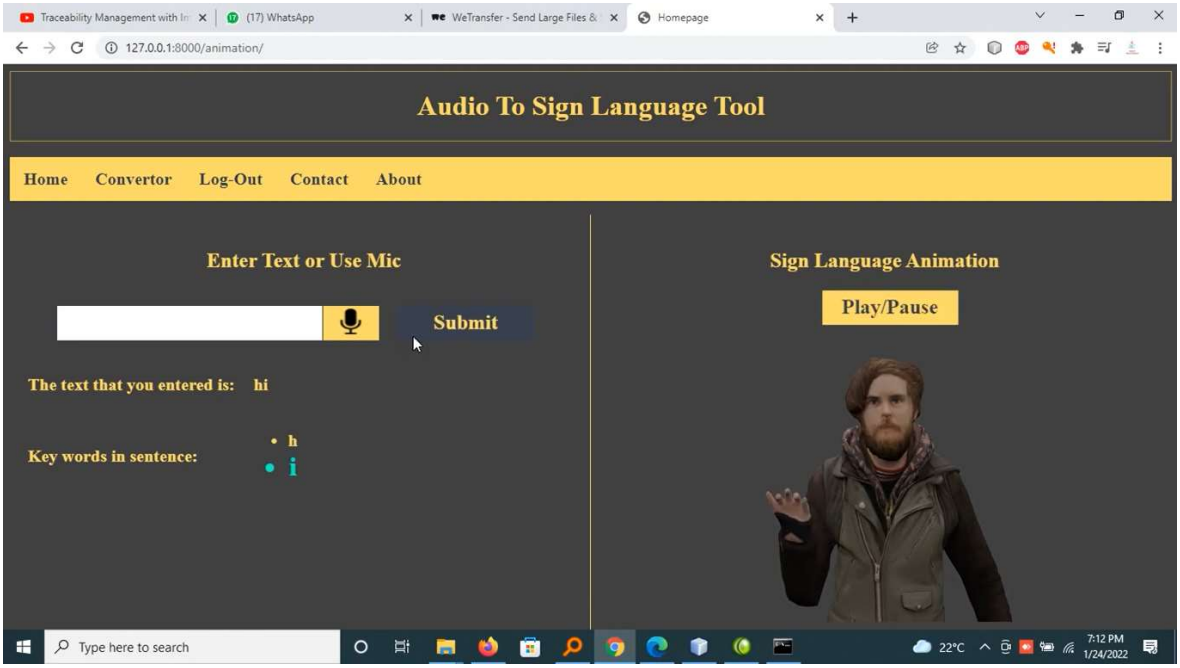


Fig 6.5. Enter Text or Audio Page

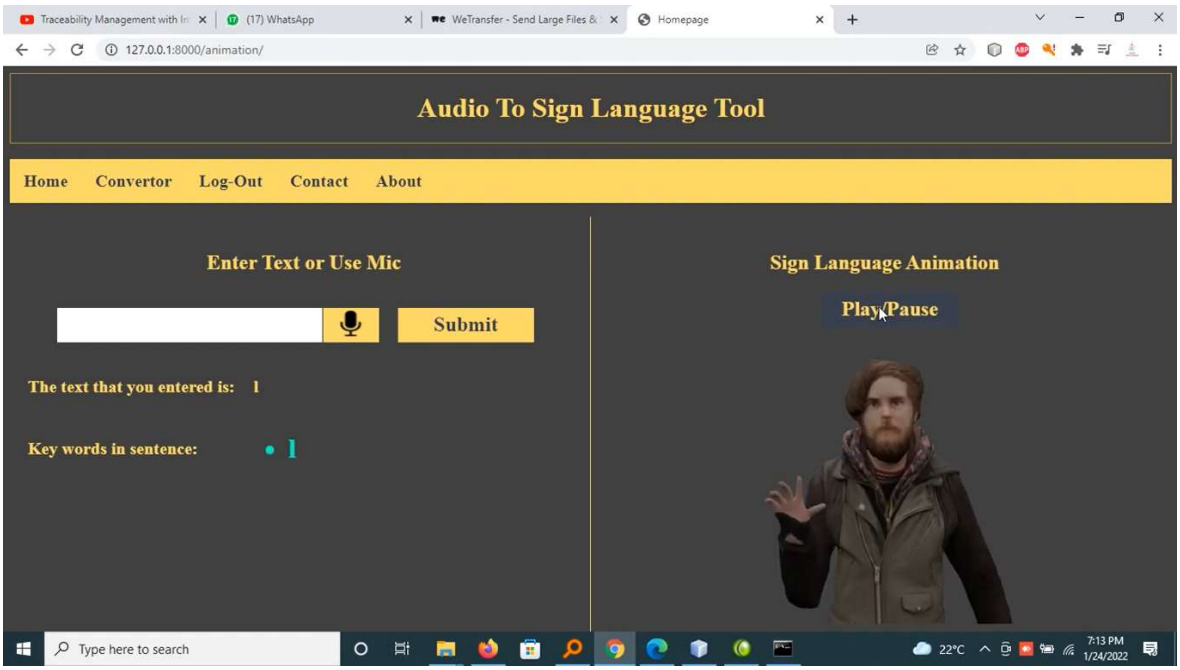


Fig 6.6. Press Play

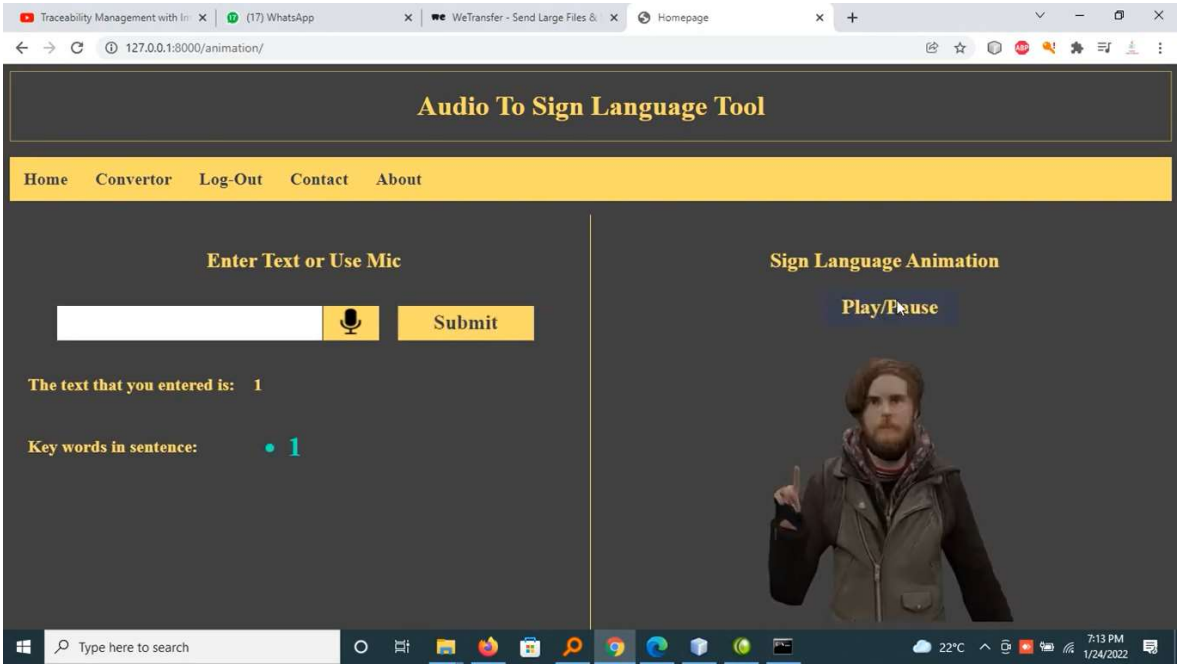


Fig 6.7. Press Pause

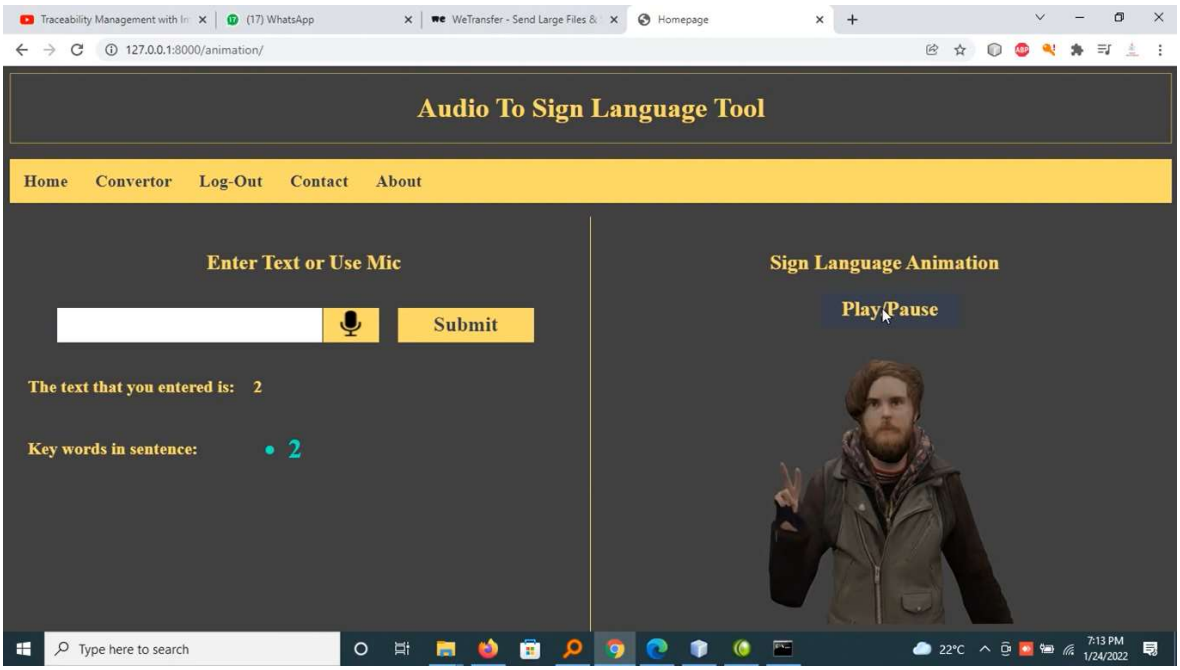


Fig 6.8.Output Screenshot 1

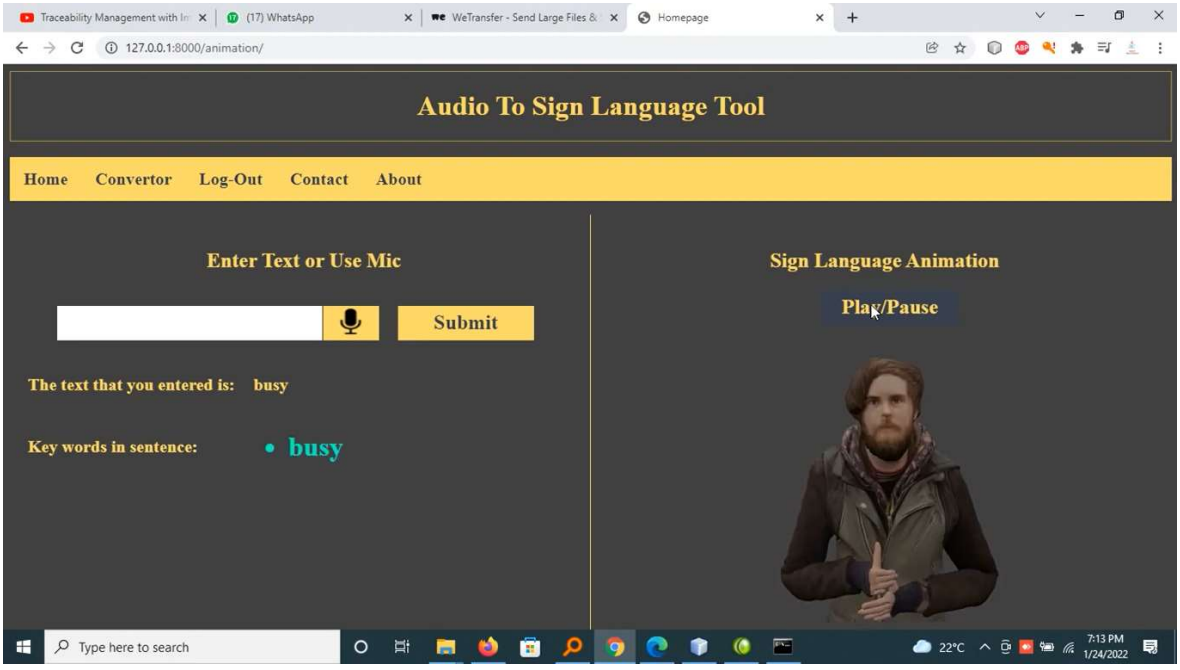


Fig 6.9.Output Screenshot 2

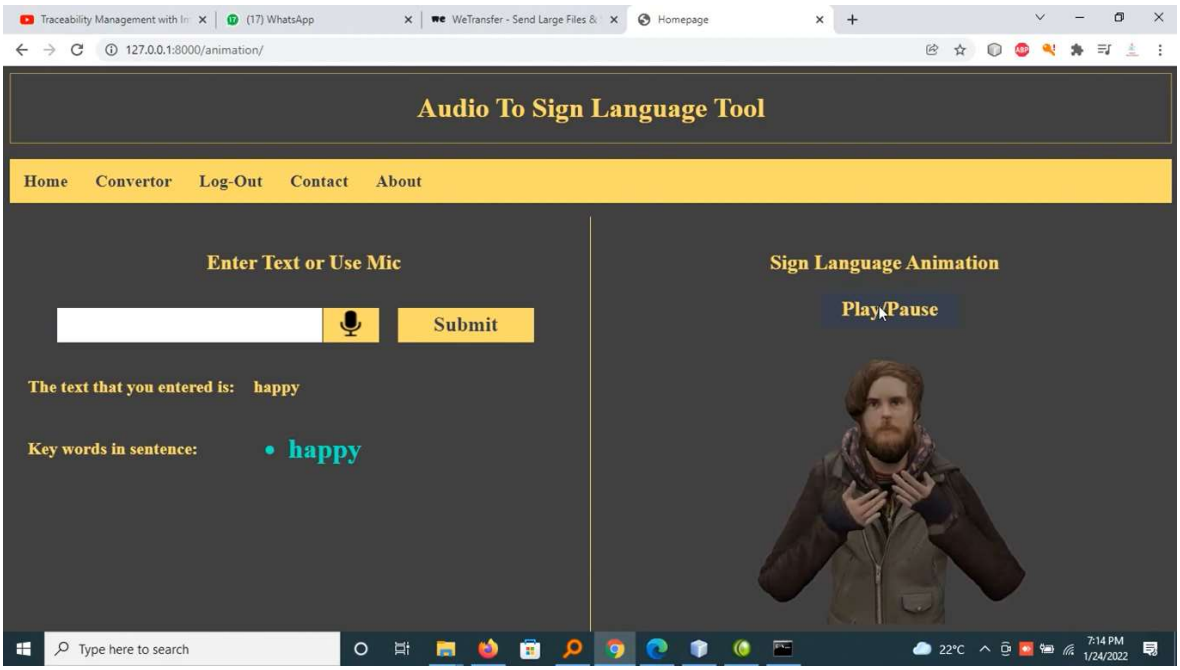


Fig 6.10.Output Screenshot 3

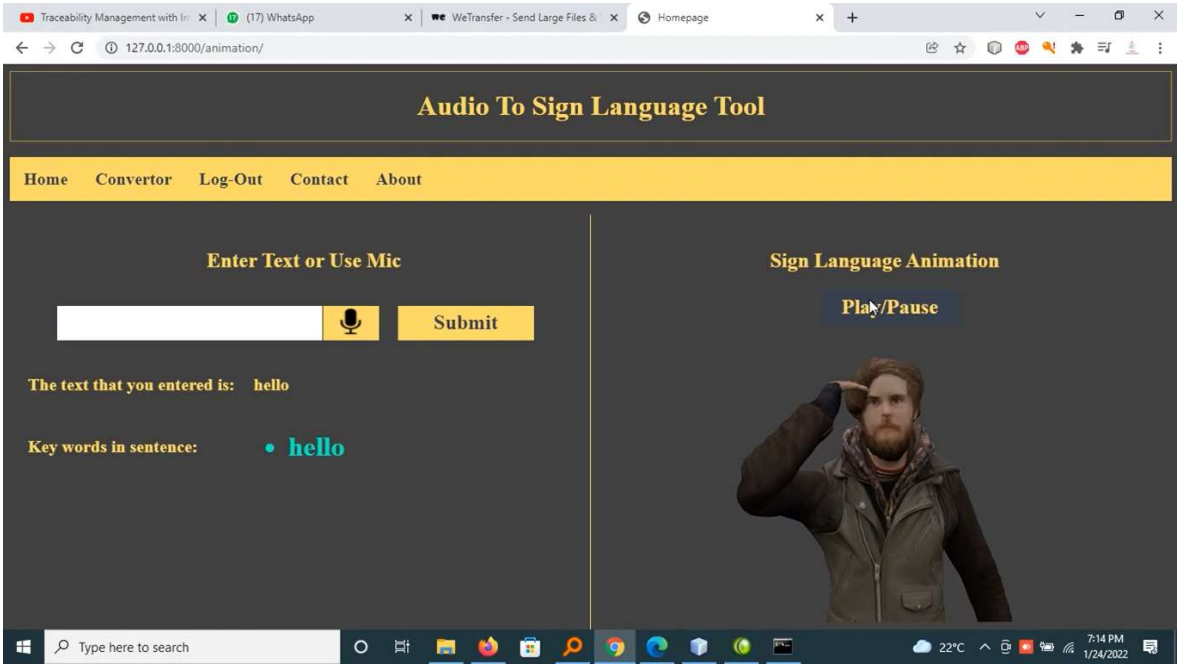


Fig 6.11.Output Screenshot 4

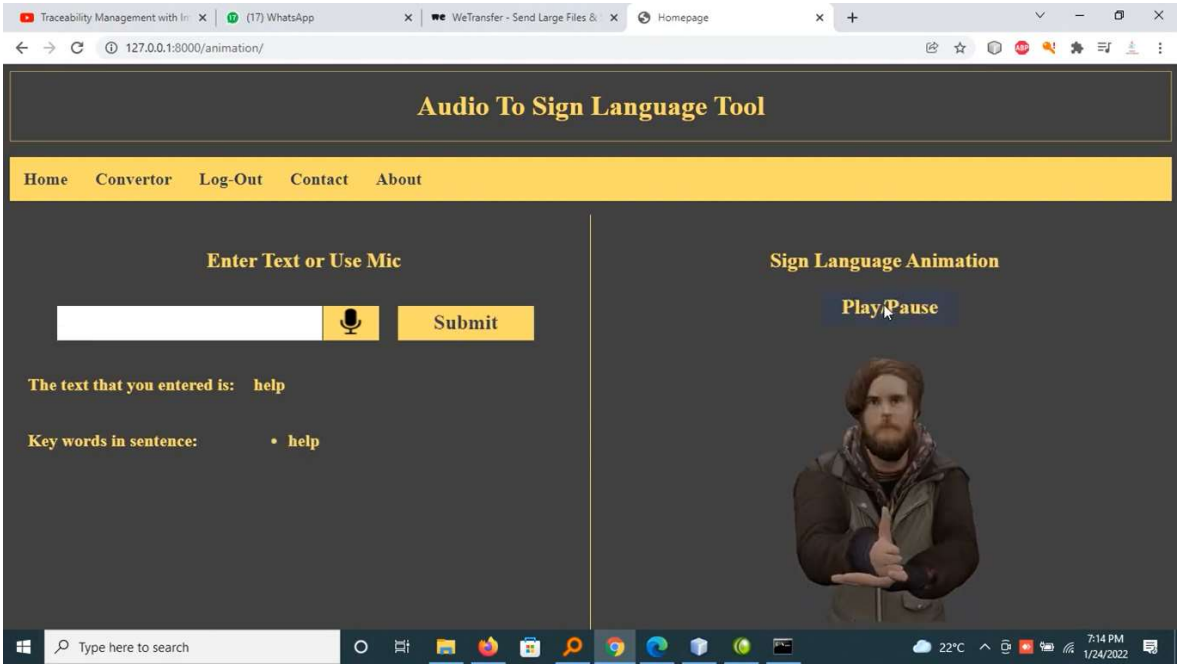


Fig 6.12.Output Screenshot 5

ADVANTAGES:

The Audio to Sign Language Tool offers significant benefits, particularly in bridging the communication gap between hearing and speech impaired individuals. By converting spoken audio into text and further translating it into sign language, the system promotes inclusivity and accessibility. It supports real time communication, making it useful in educational, social, and professional settings. Additionally, the use of speech to text APIs ensures adaptability to various languages and accents, enhancing the system's flexibility. The integration of language models boosts accuracy, even in noisy environments or with incomplete speech, making the tool reliable and efficient. Overall, the project demonstrates a meaningful application of AI to improve lives and foster better interaction for the hearing impaired community.

Furthermore, the system can be used in public services, education, healthcare, and customer service sectors to foster inclusivity. The user friendly interface also makes it accessible to non technical users, and the sign language animation ensures clarity in communication without requiring a live interpreter. Overall, this project demonstrates how AI and language processing technologies can be harnessed to build a more inclusive and connected society.

APPLICATIONS:

The Audio to Sign Language Tool has a wide range of applications across various sectors. In the education sector, it can assist students with hearing impairments by converting classroom lectures into sign language in real time, promoting equal learning opportunities. In healthcare, it can help patients with hearing disabilities communicate effectively with doctors and nurses without the need for an interpreter. The tool can also be integrated into customer service systems in banks, airports, and government offices to support inclusive communication. In public announcements and emergency alerts, it ensures that critical information is accessible to all, including the hearing impaired community. Additionally, it can be used in broadcast media and online platforms to make audio content more accessible through sign language translations. The system can also play a role in smart home devices and virtual assistants, making voice controlled technology more inclusive. Overall, this project contributes significantly toward building an inclusive society by ensuring that communication barriers are minimized.

CONCLUSION:

Sign language is one of the useful tools to ease the communication between the deaf and mute communities and normal society. Though sign language can be implemented to communicate, the target person must have an idea of the sign language which is not possible always. This was meant to be a prototype to check the feasibility of recognizing sign language. The normal people can communities with deaf or dumb using sign language and the text will be converted to sign images. This project was developed as a prototype to explore the feasibility of recognizing and converting text into sign language gestures through visual outputs, making communication more accessible for both sides. The system allows hearing individuals to type or speak text, which is then automatically translated into corresponding sign language images or animations, enabling deaf or mute users to understand the conversation without the need for an interpreter. This not only promotes inclusivity but also encourages awareness and adoption of sign language among the general public, ultimately fostering a more connected and empathetic society.

FUTURESCOPE:

The future scope of the Audio to Sign Language Tool is vast, with potential enhancements in both functionality and accessibility. One promising direction is the integration of real time video sign translation, where the system could generate dynamic 3D sign avatars or even use holograms for more expressive and natural communication. Support for multiple regional sign languages can be added to cater to diverse user groups globally. The project can also benefit from machine learning advancements, allowing it to adapt and improve through continuous usage, handling slang, accents, and emotional tones more effectively. Integration with wearable devices like AR glasses or smartwatches could offer seamless communication in everyday life.

Moreover, deploying the tool as a mobile application or browser extension would make it more accessible to a broader audience. With ongoing improvements in speech recognition and AI driven animation, this tool has the potential to become an essential assistive technology for inclusive communication worldwide.

REFERENCES:

1. Amit kumar shinde and Ramesh Khagalkar “sign language to text and vice versa recoganization using computer vision in Marathi” International journal of computer Application (0975-8887) National conference on advanced on computing (NCAC 2015).

2. Sulabha M Naik Mahendra S Naik Akriti Sharma " Rehabilitation of hearing impaired children in India"International Journal of Advanced Research in Computer and Communication Engineering.

3. Neha Poddar, Shrushti Rao, Shruti Sawant, Vrushali Somavanshi, Prof. Sumita Chandak "Study of Sign Language Translation using Gesture Recognition" International Journal of Advanced Research in Computer and Communication Engineering Vol. 4, Issue 2, February 2015.

4. Christopher A.N. Kurz "The pedagogical struggle of mathematics education for the deaf during the late nineteen century: Mental Arithmetic and conceptual understanding" Rochester Institute of Technology, Rochester, NY USA. Interactive Educational Multimedia, Number 10 (April 2005), pp. 54-65.

5. Foez M. Rahim, Tamnun E Mursalin, Nasrin Sultana “Intelligent Sign Language Verification System Using Image Processing, clustering and Neural Network Concepts” American International University of Liberal Arts-Bangladesh.

6. Shweta Doura, Dr . M.M.Sharmab "the Recognition of Alphabets of Indian Sign Language by Sugeno type Fuzzy Neural Network"International Journal of Scientific International Journal of Scientific Engineering and Applied Science (IJSEAS) – Volume-6, Issue-6, June 2020 ISSN: 2395-3470 www.ijseas.com 6 Engineering and Technology (ISSN : 2277-1581) Volume 2 Issue 5, pp : 336-341 1 May 2013.

7. Neha V. Tavari A. V. Deorankar Dr. P. N. Chatur" A Review of Literature on Hand Gesture Recognition for Indian Sign Language"International Journal of Advance Research in Computer Science and Management Studies Volume 1, Issue 7, December 2013.

8. Vajjarapu Lavanya, Akulapravin, M.S., Madhan Mohan" Hand Gesture Recognition And Voice Conversion System Using Sign Language Transcription System" ISSN : 2230-7109 (Online) | ISSN : 2230-9543 (Print) IJECT Vol. 5, Issue 4, Oct - Dec 2014.

9. Sanna K., Juha K., Jani M. and Johan M (2006), Visualization of Hand Gestures for Pervasive Computing Environments, in the Proceedings of the working conference on advanced visual interfaces, ACM, Italy, p. 480-483.

10. Jani M., Juha K., Panu K., and Sanna K. (2004). Enabling fast and effortless customization in accelerometer based gesture interaction, in the Proceedings of the 3rd international conference on Mobile and ubiquitous multimedia. ACM, Finland. P. 25-31.

11. Divyanshee Mertiya, Ayush Dadhich, Bhaskar Verma, DipeshPatidar "A Speaking module for Deaf and Dumb", student, assistant professor Department of Electronics & comm. Poornima Institute of Engineering and Technology, Jaipur, Rajasthan, India.

12. T. Kapuscinski and M. Wysocki, "Hand Gesture Recognition for Man-Machine interaction", Second Workshop on Robot Motion and Control, October 18-20, 2001, pp. 91-96.

13. D. Y. Huang, W. C. Hu, and S. H. Chang, "Vision-based Hand Gesture Recognition Using PCA+Gabor Filters and SVM", IEEE Fifth International Conference on Intelligent Information Hiding and Multimedia Signal Processing, 2009, pp. 1-4.

14. C. Yu, X. Wang, H. Huang, J. Shen, and K. Wu, "Vision-Based Hand Gesture Recognition Using Combinational Features", IEEE Sixth International Conference on Intelligent Information Hiding and Multimedia Signal Processing, 2010, pp. 543-546.