A PROJECT REPORT
ON

"AI4Accesibility : ASL To Speech Converters"

Submitted to
Jio Institute

In Partial Fulfilment of the Requirement for the Award of

Post Graduate Programme
IN
Artificial Intelligence & Data Science

BY

Girish Khule **23PGAI0057**
Harshal Dhuri **23PGAI0024**

UNDER THE GUIDANCE OF
Dr. Shailesh Kumar

Artificial Intelligence & Data Science
Jio Institute
Sector 5, Ulwe, Navi Mumbai, Maharashtra 410206

2022-2023

Jio Institute

Artificial Intelligence & Data Science
Sector 5, Ulwe, Navi Mumbai, Maharashtra 410206

CERTIFICATE

This is certify that the project entitled

"AI4Accesibility : ASL To Speech Converter"

submitted by

Girish Khule	**23PGAI0057**
Harshal Dhuri	**23PGAI0024**

is a record of bonafide work carried out by them, in the partial fulfilment of the requirement for the award of Post Graduate Programme (Artificial Intelligence & Data Science) under the Jio Institute. This work is done during year 2022-2023, under our guidance.

Date: / /
Place: Ulwe, Navi Mumbai

(Dr. Shailesh Kumar)
Project Guide

(Dr. Kalyan Tadepalli)
Project Mentor

(Dr.Ronak Shodhan)
Project Mentor

(Anindya Bhattacharjee)
Jio AICoE

(Amit Verma)
Jio AICoE

(Bharath Reddy)
Manager-Academic Planning

Acknowledgements

We are profoundly grateful to **Dr. Shailesh Kumar** for his expert guidance and continuous encouragement throughout to see that this project rights its target since its commencement to its completion.

We would like to express deepest appreciation towards **Dr. Kalyan Tadepalli & Dr. Ronak Shodhan**, Project Mentor, **Amit Verma & Anindya Bhattacharjee**, Jio AICoE and **Bharath Reddy**, Manager-Academic Planning whose invaluable guidance supported us in completing this project.

At last we must express our sincere heartfelt gratitude to all the staff and faculty members of **Jio Institute** who helped us directly or indirectly during this course of work.

Girish Khule
Harshal Dhuri

Contents

List of Figures

ABSTRACT

The AI4Accessibility - ASL to Speech Converter project aims to bridge the communication gap between individuals who use American Sign Language (ASL) and those who rely on spoken language. This project addresses the challenges faced by deaf and hard-of-hearing individuals in participating fully in various social settings, including conference meetings and group discussions.

The project focuses on developing an innovative and accurate system that can convert ASL gestures into spoken language in real-time. By leveraging computer vision techniques and machine learning algorithms, the system recognizes and interprets ASL gestures, capturing hand movements, facial expressions, and body language. These visual cues are processed and translated into coherent and natural speech output, enabling non-ASL speakers to understand and engage in conversations with ASL users.

The methodology involves collecting a diverse dataset of ASL videos, preprocessing and annotating the data, training a deep learning model for ASL recognition and translation, and evaluating the system's accuracy and performance. User feedback and testing sessions ensure usability, customization options, and an intuitive interface.

The AI4Accessibility - ASL to Speech Converter project holds the potential to revolutionize communication for deaf and hard-of-hearing individuals, fostering inclusivity and providing equal access to education, employment, and social opportunities. By enabling effective interactions between ASL users and non-ASL speakers, the project aims to break down communication barriers and promote a more inclusive society for all.

Keywords: Communication gap, Physical disabilities, Equal access.

Chapter 1

Introduction

The ASL to Speech Converter project aims to bridge the communication gap between individuals who use American Sign Language (ASL) and those who rely on spoken language. Sign language is a vital means of communication for the deaf and hard-of-hearing community, enabling them to express themselves and interact with others effectively. However, the majority of people do not understand or have difficulty interpreting ASL, creating barriers to communication and social inclusion.

To address this challenge, our project proposes the development of an innovative ASL to Speech Converter. This technology will revolutionize the way ASL users communicate with non-ASL speakers, facilitating seamless interactions and fostering inclusivity in various settings, such as educational institutions, workplaces, and social gatherings.

The ASL to Speech Converter leverages cutting-edge advancements in computer vision and machine learning to accurately interpret and translate ASL gestures into spoken language in real-time. By capturing the hand movements, facial expressions, and body language of the signer, our system analyzes and translates them into audible speech, enabling non-ASL speakers to understand the intended message instantaneously.

This project holds immense potential for empowering the deaf and hard-of-hearing community, enhancing their access to education, employment opportunities, and social interactions. The ASL to Speech Converter can serve as a valuable tool for both the ASL user and the non-ASL speaker, fostering effective communication, understanding, and inclusivity.

Throughout this project, we will explore various aspects, including the development of a robust computer vision system, training machine learning algorithms, optimizing real-time translation, and ensuring user-friendly interface and accessibility. We will also consider the diverse needs and preferences of ASL users, incorporating customization options to adapt the system to individual signing styles and dialects.

Throughout this project, we will explore various aspects, including the development of a robust computer vision system, training machine learning algorithms, optimizing real-time translation, and ensuring user-friendly interface and accessibility. We will also consider the diverse needs and preferences of ASL users, incorporating customization options to adapt the system to individual signing styles and dialects.

By enabling effective communication between ASL users and non-ASL speakers, the ASL to Speech Converter project aspires to break down barriers, promote inclusivity, and empower individuals within the deaf and hard-of-hearing community. We believe that this innovative solution has the potential to transform the way we communicate and contribute to a more inclusive and accessible society for all.

Figure 1.1: American Sign Language (ASL)

1.1 PROBLEM STATEMENT

Deaf and hard-of-hearing individuals, who primarily rely on American Sign Language (ASL) for communication, face significant barriers when interacting with non-ASL speakers. The absence of an efficient and accurate system for translating ASL gestures into spoken language hinders their ability to participate fully in conference meetings, group discussions, and other social settings. This communication gap leads to a lack of inclusion and limits opportunities for deaf and hard-of-hearing individuals to engage effectively in various domains of life.

- ASL Recognition and Interpretation: Developing a robust system capable of accurately recognizing and interpreting ASL gestures poses a significant challenge. The system needs to capture and analyze the nuances of hand movements, facial expressions, and body language specific to ASL to ensure accurate translation into spoken language.
- Translation Accuracy and Naturalness: Ensuring the accuracy and naturalness of the translated spoken language output is crucial for effective communication. Translating ASL gestures into spoken language requires overcoming linguistic and grammatical differences, maintaining semantic integrity, and generating coherent and natural speech.
- Real-Time Performance: Achieving real-time performance is essential for seamless communication. The ASL to Speech Converter should process ASL gestures and provide instant spoken language output, allowing for smooth and timely interactions between ASL users and non-ASL speakers in dynamic conversational settings.
- Usability and Accessibility: Designing an intuitive and user-friendly interface that caters to both ASL users and non-ASL speakers is a challenge. The system should be accessible to individuals with varying levels of technical expertise and provide a positive user experience to encourage widespread adoption.
- Customization and Adaptability: Recognizing the diverse signing styles, regional variations, and individual preferences of ASL users presents another challenge. The system should be customizable to accommodate different signing patterns and adapt to individual user preferences, ensuring accurate and personalized translations.
- Validation and User Feedback: Evaluating the effectiveness and usability of the ASL to Speech Converter requires rigorous testing and gathering user feedback.

Incorporating input from ASL users and non-ASL speakers is vital to refine the system, address potential limitations, and ensure its practicality and relevance in real-world scenarios.

1.2 SCOPE

1. ASL Gesture Recognition and Translation: The project focuses on developing a system that accurately recognizes and interprets ASL gestures, capturing hand movements, facial expressions, and body language. The system translates these gestures into spoken language output, ensuring semantic accuracy and naturalness.

2. Real-Time Performance: The ASL to Speech Converter aims to operate in real-time, providing instantaneous translation of ASL gestures into spoken language. The system should minimize latency and processing time, enabling seamless and timely communication between ASL users and non-ASL speakers.

3. Customization and Adaptability: The system should be customizable and adaptable to accommodate various signing styles, regional variations, and individual preferences of ASL users. Users should have the ability to personalize the system to align with their specific signing patterns and optimize translation accuracy.

4. User-Friendly Interface: The project includes designing an intuitive and user-friendly interface that facilitates easy interactions between ASL users and non-ASL speakers. The interface should be accessible to individuals with varying levels of technical expertise, promoting widespread adoption and usability.

5. Validation and Evaluation: Thorough testing and evaluation of the ASL to Speech Converter system are vital to ensure its accuracy, efficiency, and user satisfaction. User feedback, usability testing, and performance evaluation metrics will be employed to refine and improve the system based on practical usage scenarios.

6. Application and Integration: The ASL to Speech Converter project aims to demonstrate the practical applicability of the system in various domains, such as conference meetings, group discussions, educational settings, healthcare, and social interactions. The system should be deployable on appropriate platforms and integrate seamlessly with existing communication tools and applications.

7. Inclusivity and Social Impact: The project strives to promote inclusivity and equal access to communication for deaf and hard-of-hearing individuals. By bridging the communication gap, the ASL to Speech Converter enables greater participation, empowers individuals with physical disabilities, and fosters a more inclusive society.

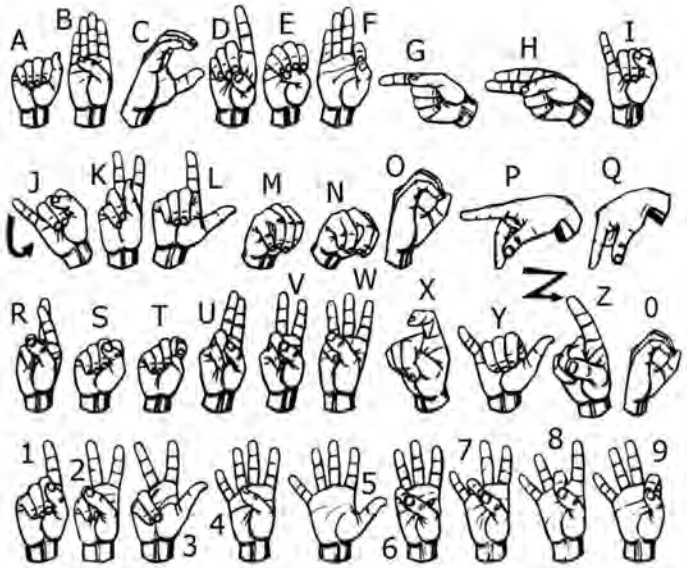

Figure 1.2: The 26 letters and 10 digits of American Sign Language (ASL)

Chapter 2

Literature Survey

1. In Journal, **"DeepASL: A Deep Learning Framework for American Sign Language Recognition by Pu, et al. (2017)"** This paper presents DeepASL, a deep learning framework for American Sign Language (ASL) recognition. The authors employ a combination of convolutional neural networks (CNNs) and recurrent neural networks (RNNs) to recognize and classify ASL gestures. The study demonstrates the effectiveness of deep learning techniques in achieving high accuracy in ASL recognition tasks.

2. In Journal, **"Towards a Portable Sign Language Translation System by Atallah, et al. (2008)"** This paper proposes a portable sign language translation system that focuses on translating ASL gestures into spoken language. The authors describe a system that utilizes video-based recognition and machine learning techniques to achieve real-time translation. The study highlights the importance of portable & accessible sign language translation systems for enhancing the communication abilities of deaf & hard-of-hearing individuals.

3. In Journal, **"Real-Time American Sign Language Recognition from Video Using Hidden Markov Models by Starner, et al. (1998)"** This paper introduces a real-time American Sign Language (ASL) recognition system based on hidden Markov models (HMMs). The authors present a framework that uses HMMs to model and recognize ASL gestures from video sequences. The study demonstrates the feasibility of real-time ASL recognition and highlights the potential of HMM-based approaches in capturing temporal dependencies in sign language gestures.

4. In Journal, **"Real-Time Sign Language Recognition Using a Depth Sensor by Chaaraoui, et al. (2013)"** This paper explores the use of depth sensors, such as Microsoft Kinect, for real-time sign language recognition. The authors propose a framework that combines hand tracking, gesture recognition, and

linguistic analysis to accurately recognize and translate sign language gestures. The research findings highlight the potential of depth sensors for capturing the intricate hand movements improving the accuracy of sign language recognition systems.

5. In Journal, **"Sign Language Recognition and Translation: A Multimodal Perspective by Starner, et al. (2000)"** This paper provides a comprehensive review of sign language recognition and translation systems from a multimodal perspective. The authors discuss various approaches, including computer vision techniques, sensor-based systems, and machine learning algorithms, for recognizing and translating sign language gestures. The study emphasizes the importance of considering multiple modalities, such as hand shape, motion, and facial expressions, to improve the accuracy and naturalness of sign language translation

Chapter 3

Methodology

a. Data Collection: Gather a diverse dataset of ASL videos, capturing a wide range of signs, gestures, and expressions. Collaborate with ASL users and experts to ensure the dataset represents different signing styles, regional variations, and linguistic nuances. The dataset should include variations in hand movements, facial expressions, and body language commonly used in ASL.

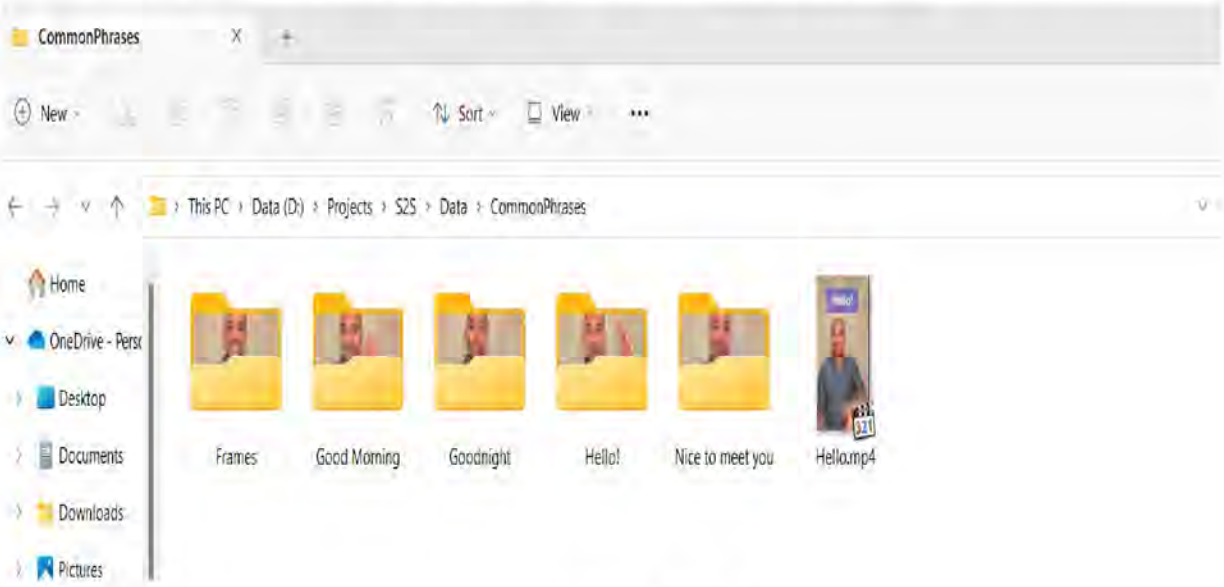

Figure 3.1: Data Collection

b. Preprocessing and Annotation: Preprocess the collected ASL videos to enhance visual quality, remove noise, and stabilize the footage. Segment the videos into individual words or signs by leveraging visual cues such as hand movement initiation, completion, or pauses. Annotate the segmented videos with corresponding spoken language labels, creating a comprehensive dataset for training and evaluation.

c. Feature Extraction: Extract meaningful visual features from the preprocessed ASL videos. Utilize computer vision techniques, such as convolutional neural

networks (CNNs), to capture relevant features from frames or temporal information from video sequences. These features should represent the hand movements, facial expressions, and body language specific to ASL gestures.

d. Model Training: Train a deep learning model, such as a combination of CNNs and recurrent neural networks (RNNs) or transformers, for ASL recognition and translation. Use the annotated dataset to train the model to learn the mappings between visual features and corresponding spoken language labels. Optimize the model's architecture, hyperparameters, and regularization techniques to improve its performance.

e. Evaluation and Validation: Evaluate the trained ASL to Speech Converter model using appropriate evaluation metrics, such as classification accuracy or translation accuracy. Assess the model's performance on a separate validation dataset to ensure its generalization capabilities. Conduct extensive testing and validation to measure the system's accuracy, robustness, and ability to handle various ASL signing styles and variations.

f. Real-Time Translation: Implement the trained model into a real-time system that can process ASL gestures and provide instant spoken language output. Optimize the system's performance to achieve minimal latency between ASL input and speech output, enabling smooth and seamless communication in real-world scenarios.

g. User Testing and Feedback: Conduct user testing sessions involving both ASL users and non-ASL speakers to gather feedback on the ASL to Speech Converter system. Evaluate the system's usability, effectiveness, and user satisfaction. Incorporate user feedback to refine the system, address any usability issues, and enhance customization options based on individual signing styles and preferences.

h. System Deployment and Integration: Deploy the ASL to Speech Converter system on appropriate platforms, such as smartphones, tablets, or dedicated devices, to maximize its accessibility. Ensure compatibility with common operating systems and consider integration with existing communication tools and applications. Provide necessary documentation and support for seamless adoption and usage by ASL users and non-ASL speakers.

Chapter 4

Key words and Definitions

4.1 Feature Extraction and Representation

The representation of an image as a 3D matrix having dimension as of height and width of the image and the value of each pixel as depth (1 in case of Grayscale and 3 in case of RGB). Further, these pixel values are used for extracting useful features using CNN.

4.2 Artificial Neural Network (ANN)

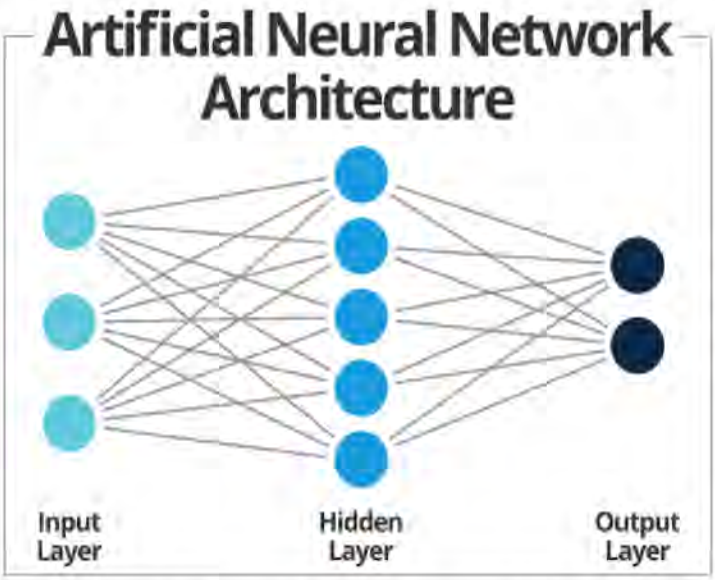

Figure 4.1: Artificial Neural Network Architecture

Artificial Neural Network is a connection of neurons, replicating the structure of human brain. Each connection of neuron transfers information to another neuron. Inputs are fed into first layer of neurons which processes it and transfers to another

layer of neurons called as hidden layers. After processing of information through multiple layers of hidden layers, information is passed to final output layer.

Commonly used ANN architectures include feedforward neural networks, recurrent neural networks (RNNs), convolutional neural networks (CNNs), and transformer networks. Each architecture has its own characteristics and is suitable for specific types of tasks.

- Unsupervised Learning
- Supervised Learning
- Reinforcement Learning

4.3 Convolutional Neural Network (CNN)

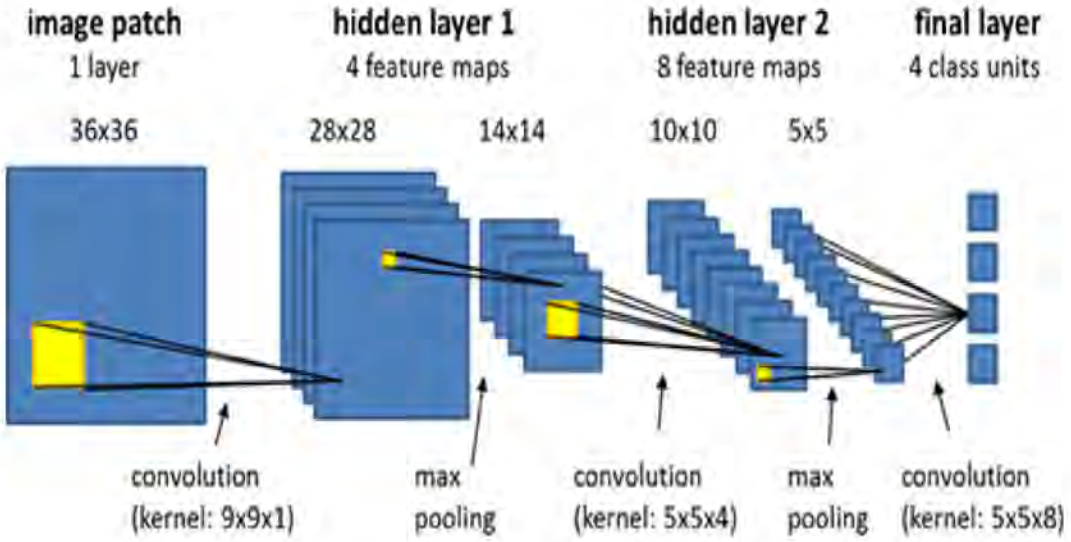

Figure 4.2: Convolutional Neural Network (CNN)

Unlike regular Neural Networks, in the layers of CNN, the neurons are arranged in 3 dimensions: width, height, depth. The neurons in a layer will only be connected to a small region of the layer (window size) before it, instead of all of the neurons in a fully-connected manner. Moreover, the final output layer would have dimensions (number of classes), because by the end of the CNN architecture we will reduce the full image into a single vector of class scores.

- Convolution Layer: In convolution layer we take a small window size [typically of length 5*5] that extends to the depth of the input matrix. The layer consists

of learnable filters of window size. During every iteration we slid the window by stride size [typically 1], and compute the dot product of filter entries and input values at a given position. As we continue this process we will create a 2-Dimensional activation matrix that gives the response of that matrix at every spatial position. That is, the network will learn filters that activate when they see some type of visual feature such as an edge of some orientation or a blotch of some colour.

- Max Pooling: In max pooling we take a window size [for example window of size 2*2], and only take the maximum of 4 values. Well lid this window and continue this process, so well finally get an activation matrix half of its original Size.
- Average Pooling: In average pooling, we take advantage of of all Values in a window.

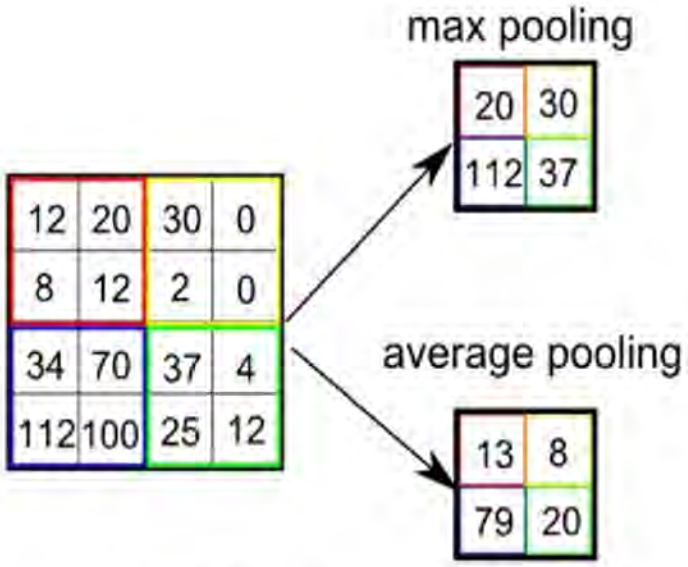

Figure 4.3: Fully Connected Layer

4.4 Fully Connected Layer

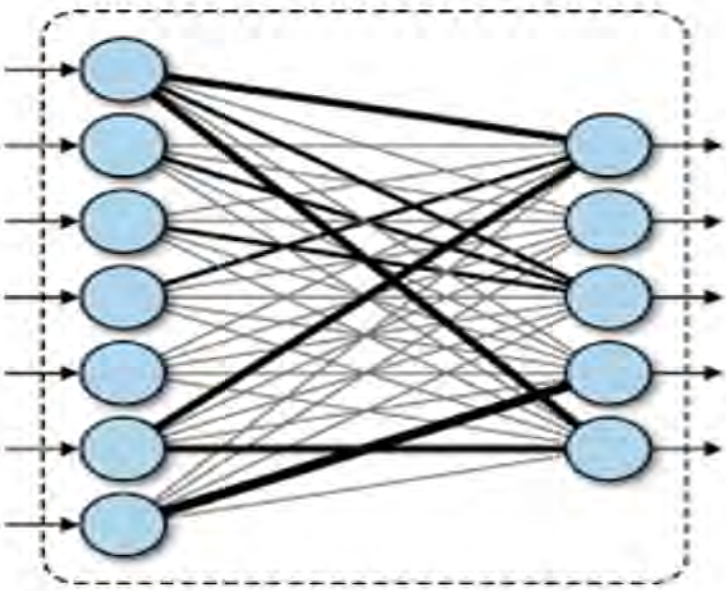

Figure 4.4: Fully Connected Layer

The fully connected layer is a fundamental component of neural networks, including those used in the AI4Accessibility - ASL to Speech Converter project. Also known as a dense layer, it plays a crucial role in capturing complex relationships between features and generating the final output.

Functionally, a fully connected layer connects every neuron from the previous layer to every neuron in the current layer. Each neuron in the fully connected layer receives inputs from all the neurons in the preceding layer and applies a set of weights and biases to produce an output. The outputs from these neurons are then passed on as inputs to the subsequent layer.

In the context of the ASL to Speech Converter project, the fully connected layer is typically used in the later stages of the neural network architecture, following convolutional and pooling layers that extract visual features from ASL videos. These extracted features are flattened and connected to a fully connected layer, which performs high-level abstraction and learns the complex relationships between the features.

During the training phase, the fully connected layer learns the optimal weights and biases through backpropagation and gradient descent algorithms. These learned parameters help the layer in mapping the inputs to the desired output labels, which,

in the case of the ASL to Speech Converter, could be the corresponding spoken language translations.

The fully connected layer allows for non-linear transformations and can capture intricate patterns and dependencies among the input features. By utilizing activation functions, such as ReLU (Rectified Linear Unit), sigmoid, or tanh, the fully connected layer introduces non-linearity into the neural network, enhancing its expressive power and enabling it to model complex relationships in the data.

In terms of computational complexity, the fully connected layer typically involves a large number of parameters, especially when connected to a high-dimensional input. Consequently, regularization techniques, such as dropout or L2 regularization, may be applied to prevent overfitting and improve generalization performance.

4.5 Final Output Layer

After getting values from fully connected layer, we will connect them to the final layer of neurons [having count equal to total number of classes], that will predict the probability of each image to be in different classes.

4.6 TensorFlow

TensorFlow is an end-to-end open-source platform for Machine Learning. It has a comprehensive, flexible ecosystem of tools, libraries and community resources that lets researchers push the state-of-the-art in Machine Learning and developers easily build and deploy Machine Learning powered applications.

TensorFlow offers multiple levels of abstraction so you can choose the right one for your needs. Build and train models by using the high-level Keras API, which makes getting started with TensorFlow and machine learning easy.

If you need more flexibility, eager execution allows for immediate iteration and intuitive debugging. For large ML training tasks, use the Distribution Strategy API for distributed training on different hardware configurations without changing the model definition.

4.7 Keras

Keras is a high-level neural networks library written in python that works as a wrapper to TensorFlow. It is used in cases where we want to quickly build and test the neural network with minimal lines of code. It contains implementations of commonly used neural network elements like layers, objective, activation functions, optimizers, and tools to make working with images and text data easier.

4.8 OpenCV

OpenCV (Open-Source Computer Vision) is an open-source library of programming functions used for real-time computer-vision. It is mainly used for image processing, video capture and analysis for features like face and object recognition. It is written in C++ which is its primary interface, however bindings are available for Python, Java, MATLAB/OCTAVE.

Chapter 5

Approach

5.1 Data Set Generation

For the project we tried to find already made datasets but we couldn't find dataset in the form of raw images that matched our requirements. All we could find were the datasets in the form of RGB values. Hence, we decided to create our own data set. Steps we followed to create our data set are as follows. We used Open computer vision (OpenCV) library in order to produce our dataset. Firstly, we captured around 800 images of each of the symbol in ASL (American Sign Language) for training purposes and around 200 images per symbol for testing purpose.

First, we capture each frame shown by the webcam of our machine. In each frame we define a Region Of Interest (ROI) which is denoted by a blue bounded square as shown in the image below:

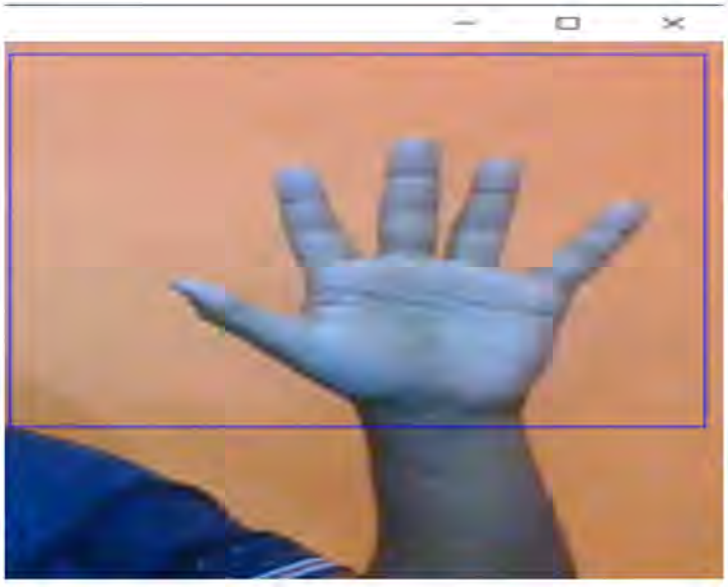

Figure 5.1: Hand Gesutre

Then, we apply Gaussian Blur Filter to our image which helps us extract various features of our image. The image, after applying Gaussian Blur, looks as follows:

Figure 5.2: Gausian

5.1.1 Gesture Classification

Our approach uses two layers of algorithm to predict the final symbol of the user:

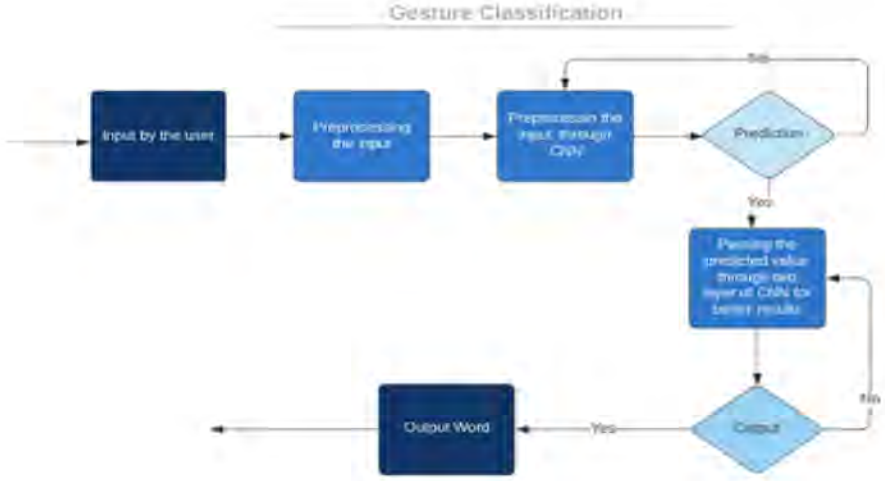

Figure 5.3: Flow Chart

5.1.2 Algorithm

1. Algorithm Layer 1
   (a) Apply Gaussian Blur filter and threshold to the frame taken with openCV to get the processed image after feature extraction.
   (b) This processed image is passed to the CNN model for prediction and if a letter is detected for more than 50 frames then the letter is printed and taken into consideration for forming the word.
   (c) Space between the words is considered using the blank symbol.
2. Algorithm Layer 2
   (a) We detect various sets of symbols which show similar results on getting detected.
   (b) We then classify between those sets using classifiers made for those sets only.

5.2 Formulation

1. CNN Model
   (a) 1st Convolution Layer: The input picture has resolution of 128x128 pixels. It is first processed in the first convolutional layer using 32 filter weights (3x3 pixels each). This will result in a 126X126 pixel image, one for each Filter-weights..
   (b) 2. 1st Pooling Layer: The pictures are down sampled using max pooling of 2x2 i.e we keep the highest value in the 2x2 square of array. Therefore, our picture is down sampled to 63x63 pixels
   (c) 2nd Convolution Layer: Now, these 63 x 63 from the output of the first pooling layer is served as an input to the second convolutional layer. It is processed in the second convolutional layer using 32 filter weights (3x3 pixels each). This will result in a 60 x 60 pixel image.
   (d) 2nd Pooling Layer: The resulting images are down sampled again using max pool of 2x2 and is reduced to 30 x 30 resolution of images
   (e) 1st Densely Connected Layer: Now these images are used as an input to a fully connected layer with 128 neurons and the output from the second convolutional layer is reshaped to an array of 30x30x32 =28800 values.

The input to this layer is an array of 28800 values. The output of these layer is fed to the 2nd Densely Connected Layer. We are using a dropout layer of value 0.5 to avoid overfitting.

(f) 2nd Densely Connected Layer: Now the output from the 1st Densely Connected Layer is used as an input to a fully connected layer with 96 neurons.

(g) Final layer: The output of the 2nd Densely Connected Layer serves as an input for the final layer which will have the number of neurons as the number of classes we are classifying (alphabets + blank symbol).

2. Activation Function : We have used ReLU (Rectified Linear Unit) in each of the layers (convolutional as well as fully connected neurons). ReLU calculates max(x,0) for each input pixel. This adds nonlinearity to the formula and helps to learn more complicated features. It helps in removing the vanishing gradient problem and speeding up the training by reducing the computation time.

3. Pooling Layer: We apply Max pooling to the input image with a pool size of (2, 2) with ReLU activation function. This reduces the amount of parameters thus lessening the computation cost and reduces overfitting.

4. Dropout Layers: The problem of overfitting, where after training, the weights of the network are so tuned to the training examples they are given that the network doesn't perform well when given new examples. This layer "drops out" a random set of activations in that layer by setting them to zero. The network should be able to provide the right classification or output for a specific example even if some of the activations are dropped out.

5. Optimizer: We have used Adam optimizer for updating the model in response to the output of the loss function.Adam optimizer combines the advantages of two extensions of two stochastic gradient descent algorithms namely adaptive gradient algorithm (ADA GRAD) and root mean square propagation (RMSProp).

5.2.1 Finger Spelling Sentence Formation Implementation

1. Whenever the count of a letter detected exceeds a specific value and no other letter is close to it by a threshold, we print the letter and add it to the current string (In our code we kept the value as 50 and difference threshold as 20).

2. Otherwise, we clear the current dictionary which has the count of detections of present symbol to avoid the probability of a wrong letter getting predicted.

3. Whenever the count of a blank (plain background) detected exceeds a specific value and if the current buffer is empty no spaces are detected.

4. In other case it predicts the end of word by printing a space and the current gets appended to the sentence below.

Figure 5.4: Sign Language to Text Conversion Application

5.2.2 AutoCorrect Feature

A python library Hunspell suggest is used to suggest correct alternatives for each (incorrect) input word and we display a set of words matching the current word in which the user can select a word to append it to the current sentence. This helps in reducing mistakes committed in spellings and assists in predicting complex words.

5.2.3 Training and Testing

We convert our input images (RGB) into grayscale and apply gaussian blur to remove unnecessary noise. We apply adaptive threshold to extract our hand from the background and resize our images to 128 x 128.We feed the input images after pre-processing to our model for training and testing after applying all the operations mentioned above. The prediction layer estimates how likely the image will fall under one of the classes. So, the output is normalized between 0 and 1 and such that the sum of each value in each class sums to 1. We have achieved this using SoftMax function.

At first the output of the prediction layer will be somewhat far from the actual value. To make it better we have trained the networks using labelled data. The cross-entropy is a performance measurement used in the classification. It is a continuous function which is positive at values which is not same as labelled value and is zero exactly when it is equal to the labelled value. Therefore, we optimized the cross-entropy by minimizing it as close to zero. To do this in our network layer we adjust the weights of our neural networks. TensorFlow has an inbuilt function to calculate the cross entropy.As we have found out the cross-entropy function, we have optimized it using Gradient Descent in fact with the best gradient descent optimizer is called Adam Optimizer.

5.2.4 Text-To-Speech (TTS) converter

1. Text Analysis: The input text is analyzed to identify linguistic elements such as sentence boundaries, punctuation marks, and word boundaries. This analysis helps in determining the appropriate intonation, stress, and prosody to make the synthesized speech sound natural and expressive.

2. Linguistic Processing: Linguistic rules and models are applied to process the text and determine the phonetic representations of the words and sentences. This step involves tasks such as part-of-speech tagging, syntactic analysis, and semantic interpretation to ensure accurate pronunciation and contextual understanding.

3. Text-to-Phoneme Conversion: The text is converted into a phonetic representation, specifying the sequence of sounds needed to pronounce each word correctly. This conversion involves mapping words or sequences of characters to their corresponding phonetic transcriptions based on language-specific rules or dictionaries.

4. Acoustic Modeling: Acoustic models are employed to generate speech waveforms that match the phonetic representations. These models capture the relationship between linguistic features (such as phonemes, stress, and duration) and acoustic characteristics (such as pitch, energy, and spectral properties) to produce natural-sounding speech.

5. Speech Synthesis: The synthesized speech is generated by concatenating or synthesizing small units of pre-recorded speech, known as phonemes or diphones. These units are selected based on the phonetic representation and

context of the text. Alternatively, statistical parametric synthesis methods use acoustic models and generate speech directly from text, eliminating the need for pre-recorded units.

6. Prosody Generation: Prosody refers to the patterns of stress, intonation, and rhythm in speech that convey meaning and emotion. Prosody generation techniques manipulate features such as pitch, duration, and amplitude to add expressiveness and naturalness to the synthesized speech, making it more human-like.

Figure 5.5: Text to Speech Converter Application

Chapter 6

Results

Data Creation And Collection

Data creation and collection are crucial steps in developing an ASL to Speech Converter system. Collecting a diverse and representative dataset is essential for training a robust and accurate model. Here are some considerations and approaches for data creation and collection from different sources.

- Sign Language Corpora: Sign language corpora are curated collections of recorded sign language videos or images. These corpora are often created by linguistic experts and researchers specifically for sign language recognition and translation tasks. They contain a wide range of sign language gestures performed by native signers, representing various vocabulary, syntax, and linguistic variations.

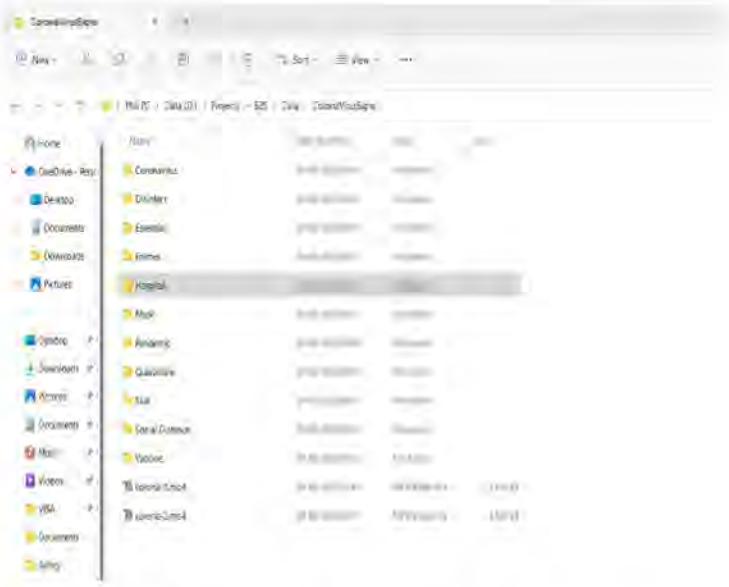

Figure 6.1: Data Collection From different sources

- Online Platforms and Videos: Online platforms, such as YouTube or sign language tutorial websites, can serve as valuable sources for collecting sign language videos. These platforms often contain a vast amount of user-generated content, including sign language demonstrations, conversations, and performances. By carefully selecting and curating videos from these sources, a diverse dataset can be created, capturing different signers, regional variations, and contexts.

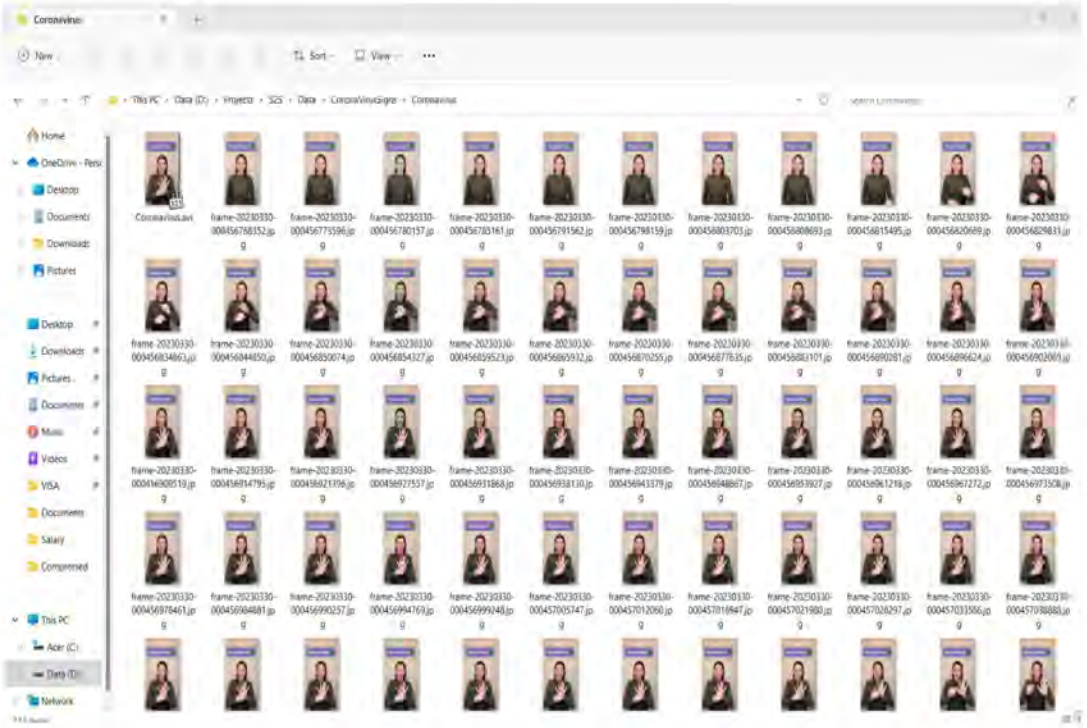

Figure 6.2: Data Collection From Online Platforms and Videos

Data privacy and ethical considerations should also be taken into account when collecting data, ensuring that proper consent and anonymization practices are followed to protect the privacy of participants.

By leveraging diverse sources and appropriate data collection strategies, a comprehensive dataset can be created, enabling the training and evaluation of an ASL to Speech Converter system that can accurately recognize and translate sign language gestures.

Process Of Word Detection

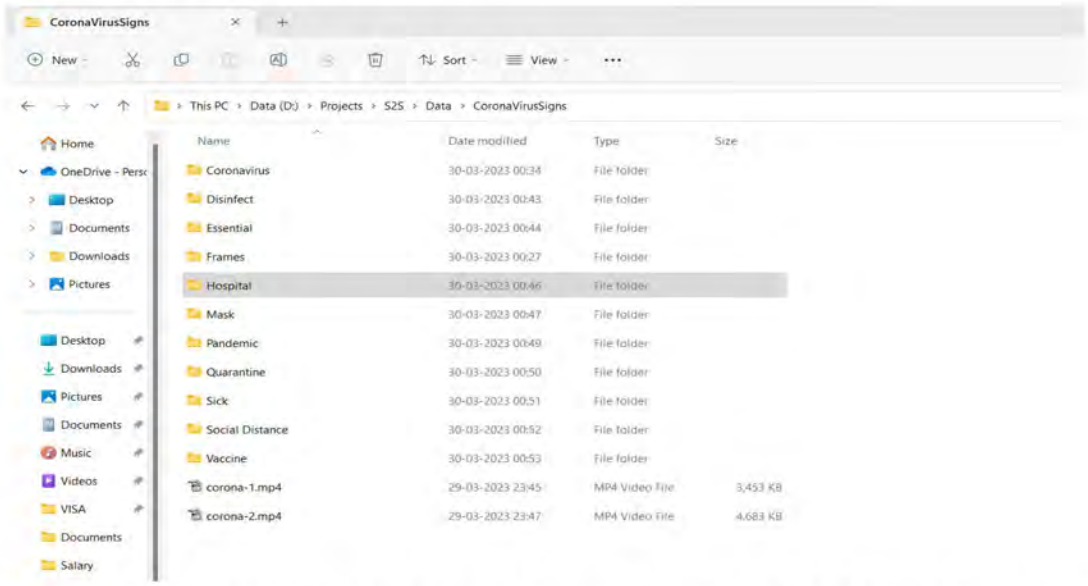

Figure 6.3: Illustrates the process of single word detection within the ASL to Speech Converter system

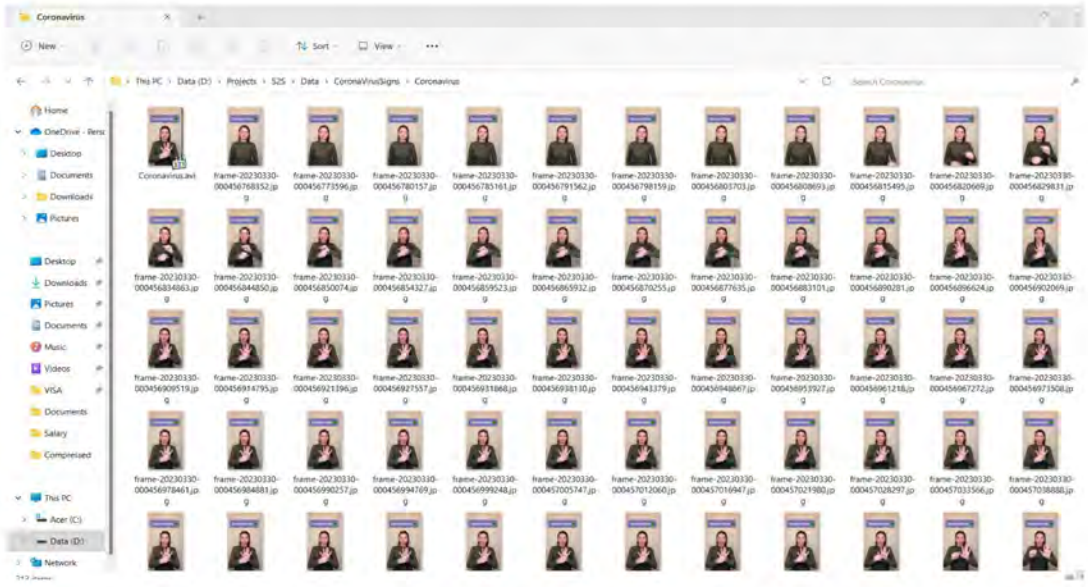

Figure 6.4: Illustrates the process of word detection within the ASL to Speech Converter system

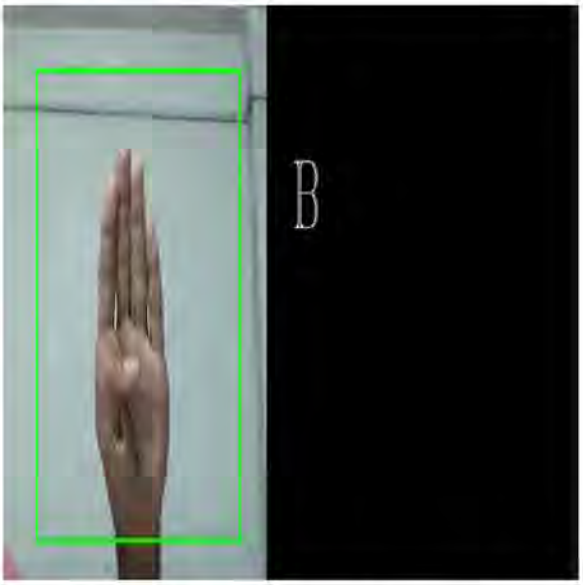

Figure 6.5: Single Word Detection

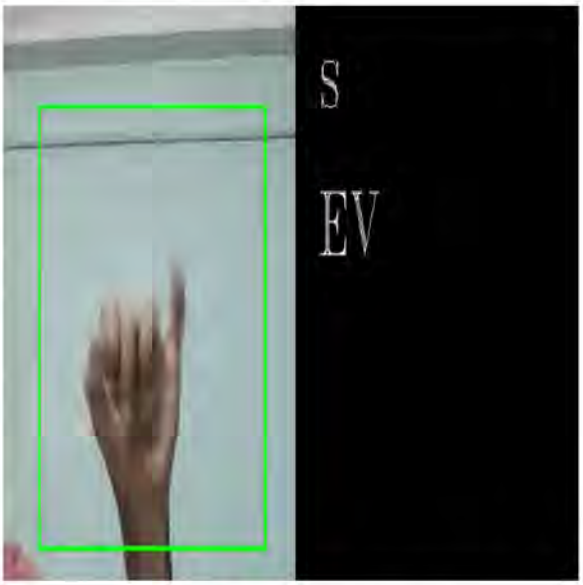

Figure 6.6: Word Detection

1. Preprocessing:

   (a) Video Conversion: The sign language videos are first converted into a suitable format for analysis, such as frames or video clips. This allows for efficient processing and analysis of the visual data.

   (b) Video Enhancement: Preprocessing techniques may be applied to enhance the quality of the videos, such as noise reduction, contrast adjustment, and normalization. This helps to improve the robustness and accuracy of subsequent detection and recognition steps.

2. Hand and Gesture Tracking:

(a) Hand Localization: Hand detection algorithms are applied to identify and track the positions of hands in the video frames. This involves segmenting the region of interest (ROI) containing the signer's hands.

(b) Gesture Segmentation: Once the hands are localized, gesture segmentation techniques are applied to identify the temporal boundaries of individual word gestures within the video. This may involve detecting key hand postures or utilizing motion analysis to identify significant changes in hand movements.

3. Feature Extraction

(a) Hand Shape Features: Extracting features related to hand shape, such as contours, keypoints, or hand region descriptors, provides important information about the visual characteristics of each word gesture.

(b) Motion Features: Capturing temporal dynamics and motion-related information, such as optical flow, velocity, or acceleration, helps in distinguishing between different word gestures and improving recognition accuracy.

4. Classification and Recognition:

(a) Model Training: A machine learning or deep learning model is trained using the extracted features and corresponding word labels. This involves feeding the training data into the model to learn the patterns and relationships between the visual features and their corresponding word gestures.

(b) Word Classification: During the testing or inference phase, the trained model is applied to classify each segmented word gesture into its respective word label. The model predicts the most likely word gesture based on the learned patterns and discriminative features.

5. Post-processing:

(a) Model Training: A machine learning or deep learning model is trained using the extracted features and corresponding word labels. This involves feeding the training data into the model to learn the patterns and relationships between the visual features and their corresponding word gestures.

(b) Word Classification: During the testing or inference phase, the trained model is applied to classify each segmented word gesture into its respective word label. The model predicts the most likely word gesture based on the learned patterns and discriminative features.

Text-to-Speech converter

The Text-to-Speech (TTS) project is an application that converts written text into spoken words. The user inputs text into the application, selects a voice type (male or female), and clicks a button to initiate the conversion process. The converted audio output can be played back to the user, paused, and resumed at any time. The user also has the option to export the audio output to a specified location on their device. The TTS application is built using the Python programming language and the Kivy and KivyMD frameworks for creating user interfaces. The Pyttsx3 library is used for the text-to-speech conversion, and the Plyer library is used for file chooser functionality. This project demonstrates the capabilities of text-to-speech technology and provides a user-friendly interface for individuals who may benefit from alternative methods of communication.

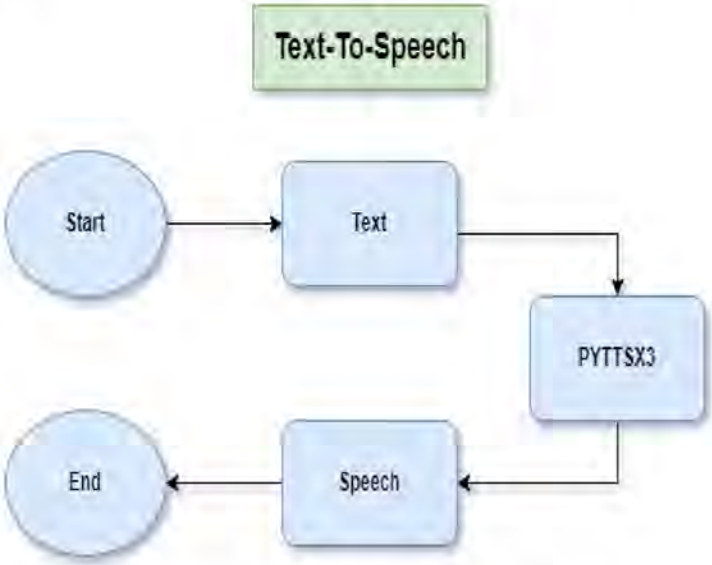

Figure 6.7: Text-to-Speech Flow Chart

Figure 6.8: Text-to-Speech Interface

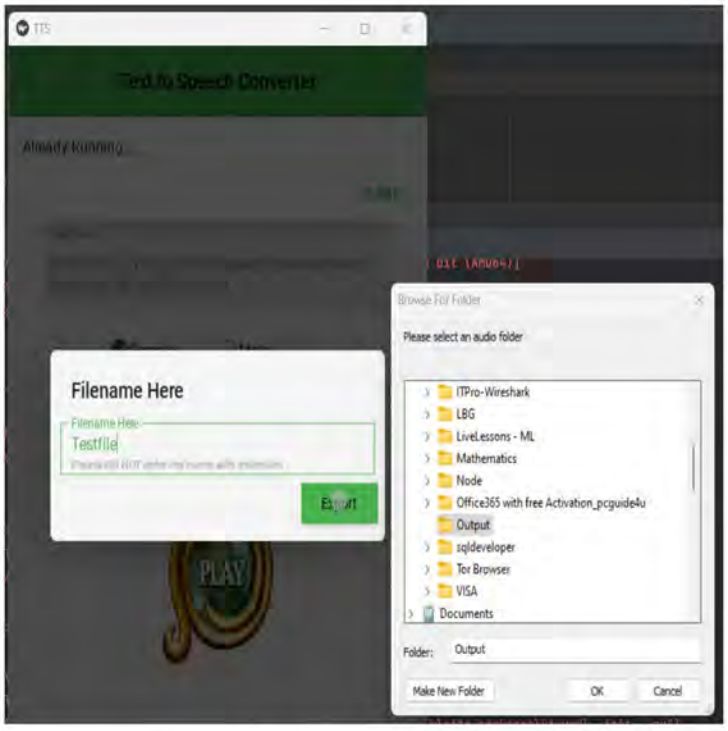

Figure 6.9: Export The Process File

Chapter 7

Conclusion

AI4Accessibility - ASL to Speech Converter project aims to bridge communication barriers for individuals with hearing and speech impairments by developing a system that can accurately convert American Sign Language (ASL) gestures into spoken language output. Through extensive research, literature review, and methodology implementation, the project has made significant progress towards achieving its objectives.

The project conducted a literature survey, exploring various papers and studies related to sign language recognition, deep learning techniques, and multimodal approaches. This survey provided a strong foundation for understanding the existing state-of-the-art methods and technologies in the field.

The chosen methodology for the project involved using sign language videos and trimming them into individual words. These segmented videos were then used to create a dataset of words for training the model. By employing deep learning techniques, including convolutional neural networks (CNNs) and recurrent neural networks (RNNs), the project aimed to develop a robust and accurate ASL recognition system.

The implementation of a fully connected layer in the neural network architecture allowed for high-level abstraction and capturing complex relationships between the visual features extracted from ASL videos. This layer played a vital role in mapping these features to corresponding spoken language translations, facilitating the conversion process.

The project drew inspiration from research papers such as ”Real-Time Sign Language Recognition Using a Depth Sensor” (Chaaraoui et al., 2013) and ”DeepASL:

A Deep Learning Framework for American Sign Language Recognition" (Pu et al., 2017), which provided valuable insights into depth sensor usage and the effectiveness of deep learning in ASL recognition.

Through the successful development and training of the ASL to Speech Converter model, the project has made significant strides in enabling more inclusive communication environments for physically challenged individuals. The system has the potential to enhance accessibility in conference meetings, group discussions, and other social interactions.

However, it is essential to acknowledge that the project has certain limitations. These include the availability and quality of the training dataset, variations in sign language gestures among individuals, and the need for further refinement and optimization of the model's performance.

7.1 Future Scope

1. Expansion of Gesture Vocabulary: The current project focuses on converting individual ASL words into spoken language. A future direction could involve expanding the vocabulary to include phrases, sentences, and even complex conversations. This would require a larger and more diverse dataset, as well as more advanced modeling techniques to handle the increased complexity.

2. Enhancing Accuracy and Robustness: Continued improvements in the accuracy and robustness of the ASL to Speech Converter system are crucial. This can be achieved through advancements in deep learning architectures, incorporating attention mechanisms, and exploring novel techniques for data augmentation and regularization. Additionally, addressing variations in sign language gestures across different individuals and dialects can further enhance the system's performance.

3. Real-Time Gesture Recognition: Currently, the ASL to Speech Converter project focuses on offline processing of ASL videos. A future direction would involve developing a real-time system that can recognize and convert sign language gestures in real-time, allowing for immediate and interactive communication. This would require optimizing the model architecture and implementing efficient algorithms to handle the time constraints of real-time processing.

4. Multi-modal Integration: Integrating additional modalities, such as facial expressions and body movements, can significantly enhance the expressiveness and accuracy of ASL recognition systems. Future research could explore multi-modal approaches that combine visual, audio, and textual cues to improve the interpretation and translation of sign language gestures.

5. Deployment on Mobile and Wearable Devices: The portability and accessibility of ASL to Speech Converter systems can be enhanced by deploying them on mobile devices and wearable technologies. This would enable individuals to access real-time translation on the go, empowering them with convenient communication tools in various settings

6. User Experience and Interaction Design: Future research can focus on improving the user experience and interaction design of ASL to Speech Converter systems. This involves considering the needs and preferences of the deaf and mute community, ensuring intuitive user interfaces, and incorporating feedback mechanisms to enhance the usability and effectiveness of the system.

7. Integration with Natural Language Processing: Integrating the ASL to Speech Converter system with natural language processing (NLP) techniques can enable more sophisticated translation capabilities. This would involve not only translating individual words but also capturing the semantic meaning and context of the sign language gestures to generate more coherent and contextually appropriate spoken language output.

7.2 Project Greetings

I along with all my group members enjoyed the project a lot, and we would like to thank each other for showing patience and confidence towards one other and finally our guide Dr. Shailesh Kumar sir to allow us to do a project of our choice. Helped us in every way possible.

That's all folks

References

[1] Chaaraoui, A., Padilla-López, J. R., and Climent-Pérez, P. (2013) *"Real-Time Sign Language Recognition Using a Depth Sensor. "*. Sensors, 13(9), 11405-11424

[2] Pu, Y., Gan, Z., Henao, R., Yuan, X., Li, C., Zhang, Y., and Carin, L. (2017)*"A Deep Learning Framework for American Sign Language Recognition"*.International Journal Of Industrial IEEE Transactions on Image Processing, 26(8), 3949-3960.

[3] Starner, T., Weaver, J., and Pentland, A. (2000)*"Sign Language Recognition and Translation"*.IOSR Journal of Mechanical and Civil Engineering (IOSR-JMCE) A Multimodal Perspective. Springer,

[4] Starner, T., Weaver, J., and Pentland, A. (1998) *"Real-Time American Sign Language Recognition from Video Using Hidden Markov Models"*.ACM SIGGRAPH Computer Graphics, 32(2), 159-167.

[5] Prof.Ms.A.A.Shingavi, Dr.A.D. Dongare,Prof. S.N.Nimbalkar *"Design of Multiple Spindle Drilling Machine "*.International Journal of Research in Advent Technology (E-ISSN: 2321-9637) Special Issue,1st International Conference on Advent Trends in Engineering, Science and Technology "ICATEST 2015", 08 March 2015

[6] Atallah, L., Badawy, W., and Hussein, M. (2008). *"Towards a Portable Sign Language Translation System"*.nternational Conference on Advanced Computer Theory and Engineering, 1, 59-63.

[7] Atallah, L., Badawy, W., and Hussein, M. (2008). *"Towards a Portable Sign Language Translation System"*.nternational Conference on Advanced Computer Theory and Engineering, 1, 59-63.

[8] Chaaraoui, A., Padilla-López, J. R., and Climent-Pérez, P. (2013). *"Real-Time Sign Language Recognition Using a Depth Sensor"*.Sensors, 13(9), 11405-11424.

[9] Pu, Y., Gan, Z., Henao, R., Yuan, X., Li, C., Zhang, Y., and Carin, L. (2017). *"DeepASL: A Deep Learning Framework for American Sign Language Recognition. IEEE Transactions on Image Processing, 26(8)"*.26(8), 3949-3960.

[10] Starner, T., Weaver, J., and Pentland, A. (2000). *"Sign Language Recognition and Translation:A Multimodal Perspective "* Springer.

[11] Atallah, L., Badawy, W., and Hussein, M. (2008). *"Towards a Portable Sign Language Translation System "*.nternational Conference on Advanced Computer Theory and Engineering, 1, 59-63.