CPSC 597 Project In Computer Science

Sign Language Recognition

Submitted By

Lency Lakhani (CWID - 885196055)

Professor Lidia Morrison

Department of Computer Science

California State University, Fullerton

Spring 2024

Abstract

In this era, Communication is the main concern in day-to -day life. In this world, there are many individuals who have disabilities such as hearing and speaking. They can not speak and hear. As a result they ended up as deaf and mute people. So, this project offers the solution for these individuals who can not communicate with normal people. Here, I am planning to make one web application which can overcome this issue. I am using one OpenCv software library to capture the hand sign of deaf and mute people through the webcam. Moreover, this sign will be recognized by the convolutional neural network machine learning model which is trained by the Sign Language dataset. Finally, it will display the text output of a given sign on the screen.The user has the capability to engage in conversations across various languages, and they also utilise a feature that enables voice translation for seamless communication. So, the normal people can easily understand their language. I have also described the main objectives,activities, architecture diagram, UML diagram and plan of project etc in this report.

Table Of Contents

Table Of Figures

Introduction

In this project, I would like to introduce great people with extraordinary talents and determination of goal. They are capable of getting things done. These people have courage to complete their goals. Despite of having communication gap between normal people, they leave remarkable marks on our hearts. They face an abundant number of challenges in life. For example, Marlee Matlin was a famous American actress who was deaf. She lost her hearing sense at a very young age. Despite these hurdles, she did well in her career. With her passion and self motivation, she got so many awards. So, Marlee Matlin is the best example for deaf and mute people. Let’s go deep into this problem.

In the United States, 3.6% of the population are deaf and mute[1].Deaf and mute people can not hear and speak properly. From the research, deaf and mute people can communicate easily with each other because they know sign language. In their schooling, they have mandatory subjects of sign language, gesture recognition, picture analysis etc. In this digital era, we have technology for deaf and mute people such as wearable devices, speech to text software, text to speech software, augmentative and alternative communication devices etc. Deaf and mute people get help from such devices. From this problem, I got to know that they communicate with each other easily but, when they communicate with normal people,they face communication gaps between them. I know you have questions such as, What is the main reason for this communication gap? How can we resolve this issue? So, let’s explore this problem in depth and from this problem I am planning to make one Web Application which can resolve this problem between deaf and mute people and normal people.In our educational system, we do not have any mandatory subject as sign language .As a result, we are unaware about sign language anatomy. So, when deaf and mute people are trying to communicate with us , at that time we are not able to understand their language immediately.

To solve this problem, I am planning to make one web application for smooth communication. All over the world, we have so many sign language dataset such as Indian sign language, Chinese sign language, French sign language etc but , I am making a feasible web application which works in every language. In this project I am using Sign Language as a dataset for my machine learning model. I am making my own dataset for the accuracy purpose. Sign Language (SL) substantially facilitates communication in the deaf community. However, there are only ~250,000-500,000 speakers which significantly limits the number of people that they can easily communicate with [2]. My web application will use one webcam to capture the hand sign image of deaf and mute people through OpenCV. Moreover, it will do some feature extraction from the image. Then, the sign will be recognized by the Convolutional Neural Network(CNN). CNN will be trained by the sign language dataset. At the conclusion, the output will be visibly presented on the screen, making it accessible for individuals to identify their language. Furthermore, I've integrated a learning resource where users can explore the sign language dataset and understand the functionality of my web application. Additionally, I've implemented multiple language translation options, allowing users to communicate in their preferred language. Moreover, I've included a voice translation feature to enhance convenience and efficiency. This project aims to be user-friendly and a time-saving solution for communication needs.

Objectives

The main objective of my project is to find the groundbreaking line to address the communication gap faced by individuals who are deaf and mute. The main central line of this project is to find any common line between individuals who are facing difficulties of communication in day to day life. Deaf and mute people often face difficulties in communication with normal people due to lack of sign language knowledge. This project basically reduces the communication gap and gives some innovative web applications for deaf and mute people.

Secondly, I need to consider the dataset of sign language. For this project, I am planning to use Sign Language as a dataset. Moreover, I need to do some research on what type of machine learning model I should go for. Right now my thinking goes onto the Convolutional Neural Networks. Convolutional neural network (CNN) is a multilayer neural network, and it is also the most classical and common deep learning framework[3]. The convolution neural network (CNN) developed in recent years has been widely used in the field of image processing because it is good at dealing with image classification and recognition problems and has brought great improvement in the accuracy of many machine learning tasks[3].

I need to use OpenCV for image capturing. It will capture the sign image of deaf and mute people. FurtherMore, we need to do some feature extraction because, in daily life we have some image distraction because of the so many lights on it. This feature extraction will extract the image in some black and white form in some pixels. Moreover, I need to convert coloured image into the black and white grayscale image. I am planning to use Gussian blur and threshold blur technique into my OpenCV capturing to enhance the ability of predication. After that, we need to run a machine learning model on this image and it will give text output on the screen .

I am planning to integrate this machine learning model with the front end part. I have done some research on whether I should go for Flask or Django. Most probably, I am planning to use Flask to integrate with frontend. Integrating Flask with front-end frameworks bridges Python's robust backend capabilities with dynamic UIs[6]. I am also using HTML/CSS and bootstrap for UI. Firstly, I am making my font end into the figma then, I will be building the frontend part through HTML/CSS, Bootstrap.

Moreover, I have added the functionalities for multiple language translation and voice translation for the web application where users can choose any language they want to communicate with. So, basically I am making a feasible web application for everyone. Firstly, I have to do some research on translation APIs. In the digital market, there are an abundant number of APIs available. But, most of the APIs are not free of cost. They are charging money for each APIs calls. Furthermore, I will be doing research on free APIs that would be good and great for my web application. When I establish language translation APIs and Voice translation APIs my project will be more highlighted and more user friendly. I have also added the pause, reset and learning resource for the user.so, the user can turn on and off video anytime. Users can also learn how to use the web application through the learning resource site.

Last objective of my project is risk and mitigation. Optimisation of challenges could be hard for trained machine learning models. However, I need to overcome this scenario and look forward to the evolving machine learning model. In my prediction, there will be major three challenges that could be came across in my project. First one is environmental hurdles, second one is sign boundary detection and last one is overfitting issue in the convolutional neural network machine learning model. Moreover, I can get some issues in the user interface where I need to find some optimisation for the project. Also real time applications will face the challenges in web application responsiveness.

Finally, my project comes up with an innovative idea with the great user experience, some innovative features and efficient solution of communication gap between individuals .

Work Scope

From this Sign Language detection and interpretation project we specifically focus on overcoming the communication challenges faced by deaf and mute communities. This project involves Sign Language Dataset, Sign Language gesture detection model, computer vision, deep learning etc.

1. **Machine Learning Sign Language detection model**
   - Design CNN model which can detect the A-Z hand sign of deaf and mute people.
   - The Sign Language dataset will be used for this project which will be captured by myself in future for training and testing so that my model will get high accuracy.

2. **Real time web application**
   - Develop responsive, adaptive and user friendly web applications with integration of high accuracy models with frontend.
   - Integrate OpenCV software library to capture the images of user's hand sign and transfer these images to the CNN model for detection.
   - Integrate backend with frontend with the use of Flask, HTML/CSS and Bootstrap.
   - Integrate Learning resources page for user's leaning purpose for signs.
   - Developed pause, reset button so, user can turn on and off video anytime
   - Integrated translation APIs for voice and language translation.

3. **Research in future**

   - Improve accuracy of CNN model by utilising different resources and techniques.
   - I will overcome challenges which I will get during development. Also, extend my further research for gesture detection.

4. **Feasibility of model**

   - I will test this model in different environmental conditions as well as different background and noise.
   - If I will face any challenges then, I will try to improve my project to be able to give correct results in any environment.

5. **Documentation**

   - Write one documentation to explain my development strategies, design and architecture of the model as well as detailing all functionality.
   - Mention some risk and mitigation which I will discover throughout the development phase.

By defining these attributes, we can say that my project will be completed and able to work in any environment. In the end, It will break the gap of communication between deaf and mute individuals.

Target Audience and Features

The target audience in my project are disabled people who are mute and deaf. Other people are normal people who want to start conversations with deaf and mute people.

1. **Deaf And Mute people**

   Deaf and mute people are the main pillar of this web application. Deaf and mute people are responsible for carrying out all the operations. Deaf and mute people will use my web application for their easy communication with normal people. Firstly, Deaf and mute people have knowledge of sign language. They will transfer their knowledge into the camera and the camera will capture the predicted sign and convert into the text. Moreover, In my web application users can choose any appropriate language they want to communicate with. Deaf and mute individuals are not only the primary beneficiaries but also the driving force behind the functionality of this web application. They are entrusted with the responsibility of executing all operations within the system. By utilising this application, they can seamlessly communicate with individuals who do not understand sign language, thus breaking down barriers to communication and fostering inclusivity.

2. Normal Users

Normal people come into the picture. Normal people is the crucial part of this web application. Normal people are the main reason I am making this web application. We all know that, Deaf and mute people has knowledge of sign language but we do not have knowledge of sign language. When they communicate with us we face hurdles to understand their language. To resolve this issue, I am making one web application which will solve the issue of communication and provide less communication gap between deaf and mute people with normal people. In my web application. Users have to choose their appropriate language to communicate with. Deaf and mute people will start the webcam and do some sign in the front of the camera. Webcam will capture the sign and find appropriate signs with the help of CNN model. After carefully testing, the predicted model will display the text output into the screen. At that time, users have the option to choose their language to translate text. Moreover, I have also included the feature of voice translation to make a user friendly web application.

Tools and Development Environment

In this page, I will explain what types of programming language, software and hardware are used in my web application.

1. **Programming Language**

    - Used Python Programming for the backend part: **Python 3.12.3**
    - Used Flask to integrate backend with frontend and provide remarkable User Interface: **Flask 2.3.3**
    - Used HTML/CSS and Bootstrap markup language to provide great UI/UX: **HTML/CSS, Bootstrap**
    - Used figma to build UI for web application: **Figma**

2. **Software Requirements**

    - Used OpenCV python library to capture live sign images: **OpenCV**
    - Used CNN machine learning model to predict sign images with the used of testing and training dataset: **CNN model**
    - Used Keras and tensorflow to implement CNN model, Particularly used tensorflow for the feature map: **Keras, Tensorflow**
    - Used Numpy python library for the image processing: **Numpy**

3. **Hardware Requirements**

    - Users should has webcam on hardware device such as mobile, computer and laptop: **Webcam**
    - My web application is purely based on the software development side so, there are no hardware requirements as such.

Requirements Description

1. **Real Time Image Capture**

   Description: This application captures real-time images of user's' hand signs through the webcam.

   Precondition: Users have to open the camera.

   Postcondition: Hand sign image will be captured one by one

2. **MSL Detection Model**

   Description: Implement a Convolutional Neural Network (CNN) model trained on the multiple sign language dataset to accurately detect hand signs.

   Precondition: Collect images after pre-processing technique.

   Postcondition: predict hand sign according to captured images.

3. **Image Processing**

   Description: Use image preprocessing techniques to subtract background, normalization, and edge detection.

   Precondition: Transfer captured images by OpenCV

   Postcondition: processed color image into black-white as well as subtract distraction from image.

4. **Web Application Interface**

   Description: Design a user-friendly web application so that users can easily interact with an application.

   Precondition: Users have to open the homepage and take images as input.

   Postcondition: Display camera screen and output text.

5. **Continuous Prediction**

   Description: Implement a continuous prediction approach to address sign boundary detection challenge.

   Precondition: Users have to on camera throughout the time

   Postcondition: CNN model will continuously predict the images and convert into the sentence.

Architecture Diagram

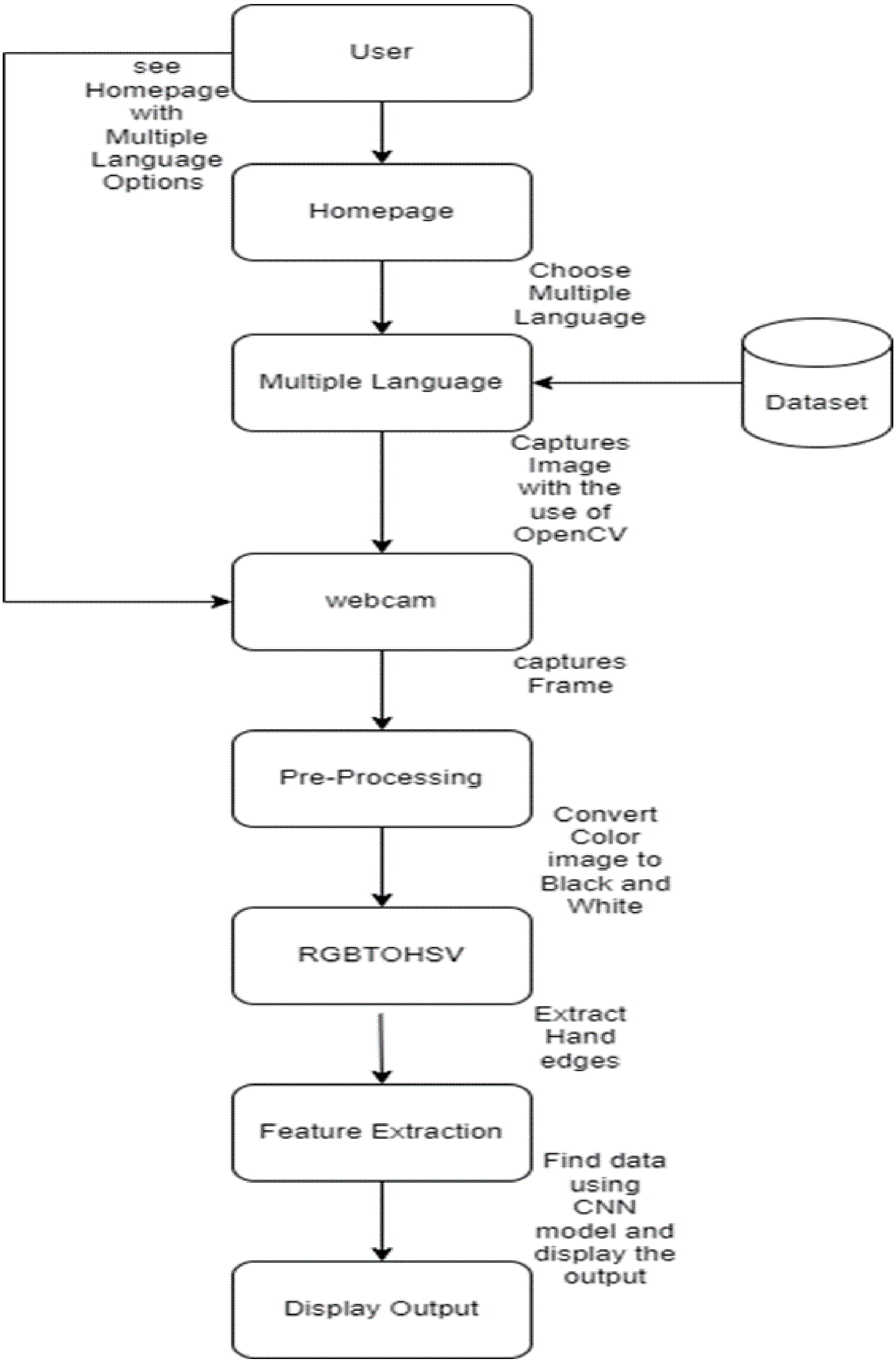

User
see Homepage with Multiple Language Options
Homepage
Choose Multiple Language
Multiple Language
Dataset
Captures Image with the use of OpenCV
webcam
captures Frame
Pre-Processing
Convert Color image to Black and White
RGBTOHSV
Extract Hand edges
Feature Extraction
Find data using CNN model and display the output
Display Output

Figure 1 Architecture Diagram

In the above Architecture diagram, User can see the homepage with the multiple language options. User can choose their appropriate language. Then, the webcam will capture the image of the sign. This image will be captured by OpenCv. It will be then pre processed. In this preprocessing stage, pixel of 8 * 8 matrices will be taken from the whole video. Moreover, in the next RGBTOHSV stage, coloured image will be transformed into the black and white form. Next stage is hand detection and edge detection, it will capture the proper edges of the hand gesture. Then the next step is feature extraction, it will extract the distraction of the image. For example, there may be a chance where multiple lights can be flooded during the image capturing. So, the main function of this step is to remove all the distraction from the frame. Then, CNN comes into the picture. CNN model has been trained by multiple datasets. It will use it to get the output from the dataset. Finally,at the end it will give output on the screen.

UML Diagram

1. Use Case Diagram

The below diagram shows the Use Case Diagram of my project. First of all, images will be captured through a webcam and processed through OpenCV. Furthermore, the image will be pre-processing and classifying. Sign image will be recognized by the CNN model and display the output on the screen.

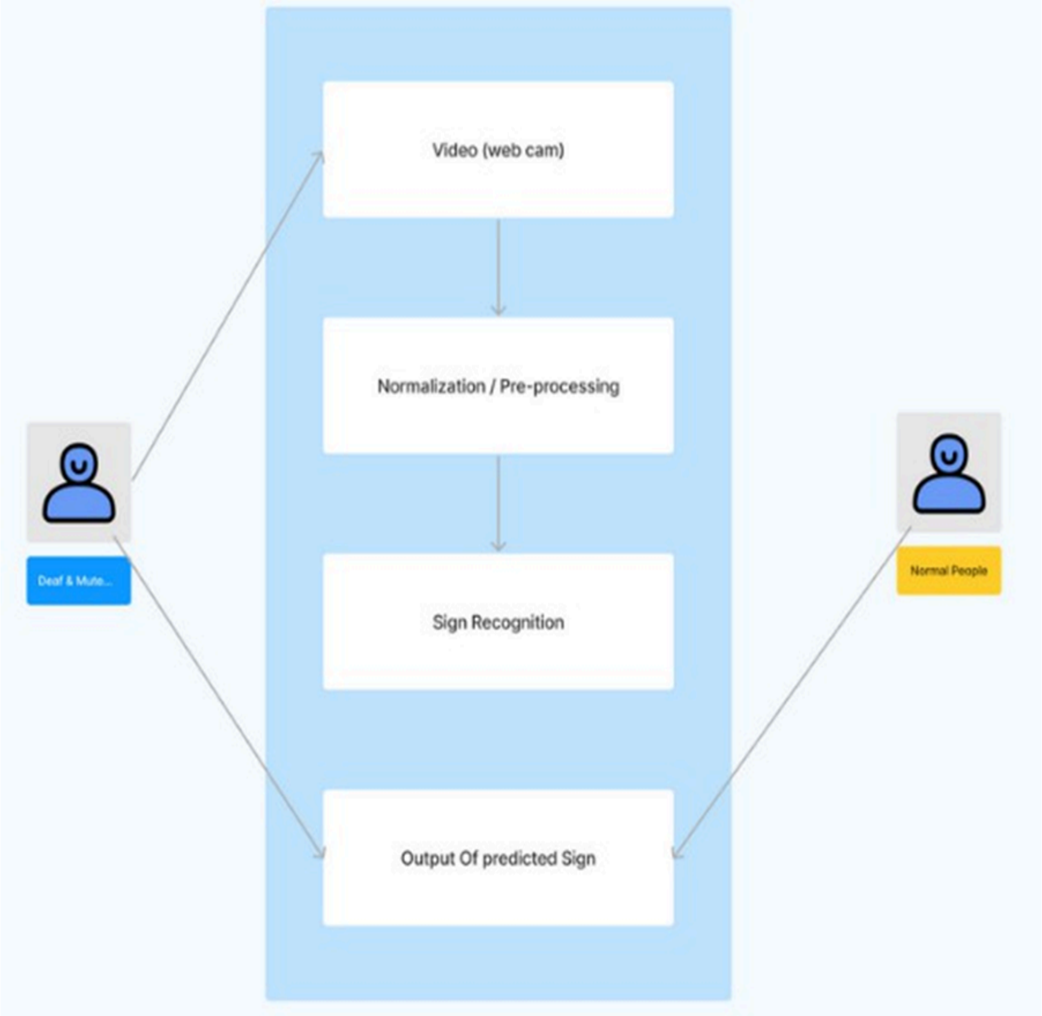


Figure 2 Use Case Diagram

2. Sequential Diagram

Sequential diagram is basically the flow of the project. In this diagram, First OpenCV is captured sign images of the users. Then, these sign image will be pre-processed and send to the CNN model. CNN model is fully trained by the multiple dataset. At the end, it displays the full output on the screen.

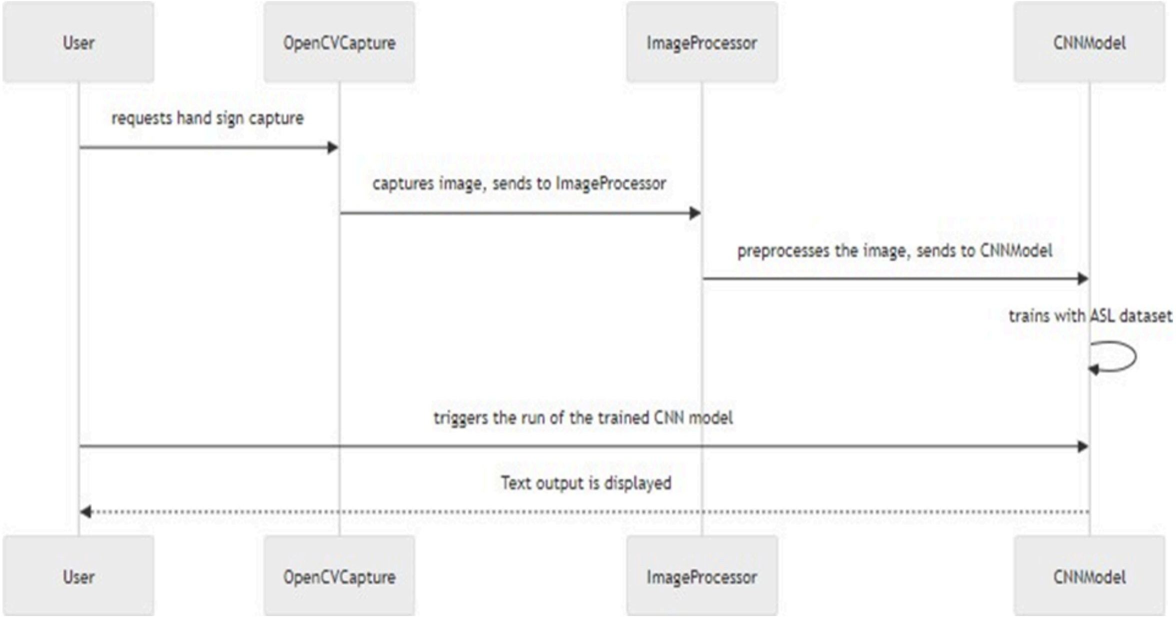

User
OpenCVCapture
ImageProcessor
CNNModel
requests hand sign capture
captures image, sends to ImageProcessor
preprocesses the image, sends to CNNModel
trains with ASL dataset
triggers the run of the trained CNN model
Text output is displayed
User
OpenCVCapture
ImageProcessor
CNNModel

Figure 3 Sequential Diagram

3. Activity Diagram

Activity diagram is a basically representation of flow charts of the projects.

This diagram is a representation of all activities which will be done in project.

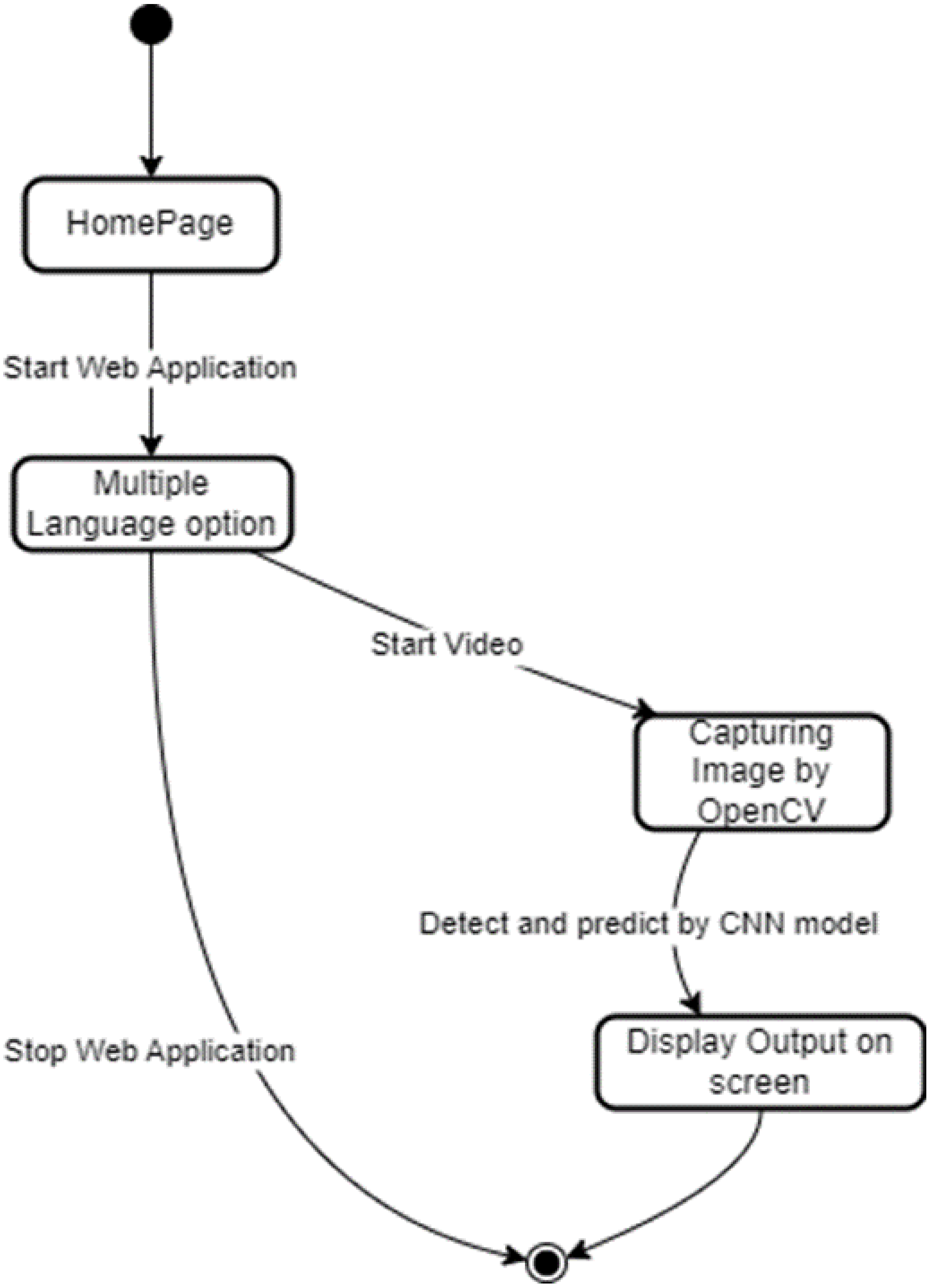


Figure 4 Activity Diagram

Activities

1. Dataset

All over the world, we have different types of people who are speaking different languages. So as per the language , we have an abundant number of sign languages which are used all over the world. We have Chinese sign language, Indian sign language, and French sign language. In this project, I am planning to use the Sign Language dataset for the reference. It has different types of sign image which will be trained by the convolutional neural network. I have made my own dataset for the accuracy purpose. So, It will be able to predict my hand sign edges fastly. As you can seen in the picture, there is an alphabetical image of a hand gesture.

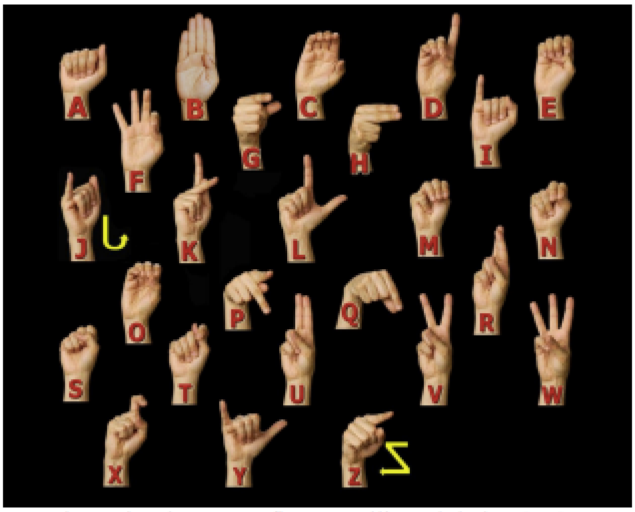


Figure 5 Sign Language Dataset [From Link]

2. **Machine Learning Model Training**

In machine learning, there are different type of machine learning models such as Artificial Neural Network, Convolutional Neural Network and k nearest neighbours. In this project, I am planning to use the Convolutional Neural Network as a machine learning model because this model is known for image processing. First of all, I need to train this model using the Sign language data set(SL). I need to use Keras, Tensorflow to implement the CNN model. Furthermore, this model will calculate the prediction score.

3. **Integrated model with OpenCV**

OpenCV is an open source software library which is used to capture the image and computer vision. It contains low level image processing functions and high level algorithms for face detection, feature matching and tracking[6]. In this project, I will use the OpenCV software library. For that I need to integrate this library into my backend support with the help of python.

4. **Testing and Validation**

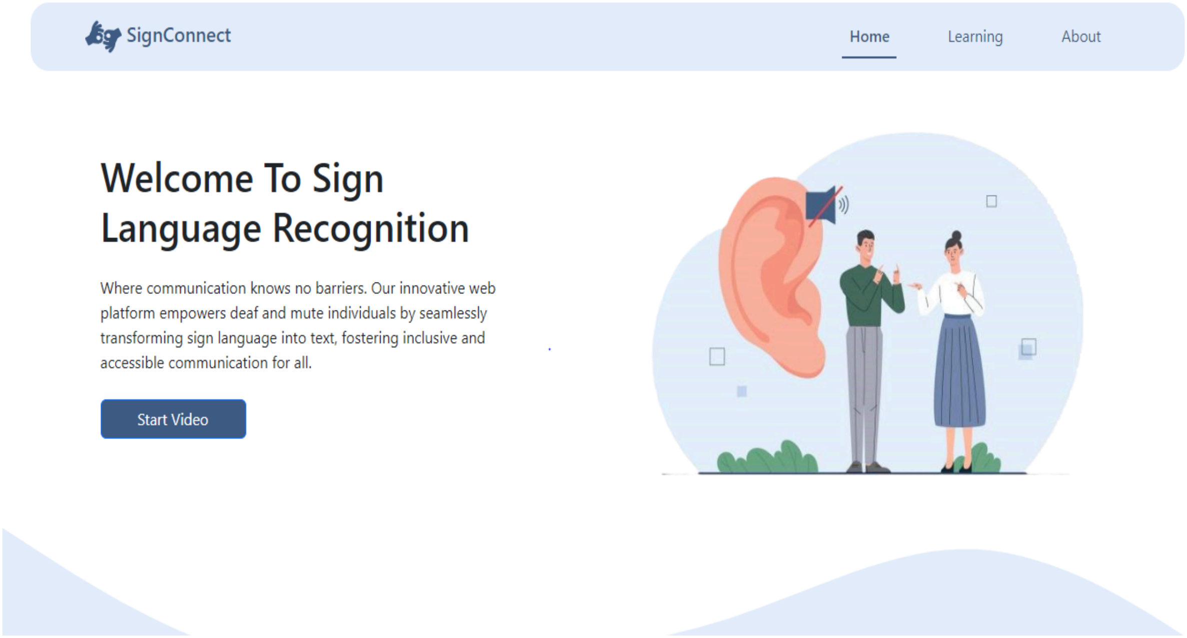
Finally, I need to check my web application in real life. I need to consider different types of aspects which can be obstacles for the project. When I will run my web application and if there is any bug then, I have to optimise it. I need to test input data, CNN model, trained dataset, feature extraction, sign recognition etc. I need to basically do trial and error for this web application.

Implementation

1. Frontend Part

1.1 Home Page

Here I have attached the frontend home page where people can see the navigation bar with home, learning and about sections. Users can also see the introduction of the web application in the bottom of that user can see the start video button. When the user clicks on the start video button, it will redirect to the prediction page where video will be turned on and predicted output will be displayed. For this frontend part I have firstly created the Figma. Figma will help to give quicksight view of user interfaces. Then, I have started building this frontend part with the use of HTML/CSS, Javascript, Ajax and bootstrap etc. I have taken the illusion image from google.



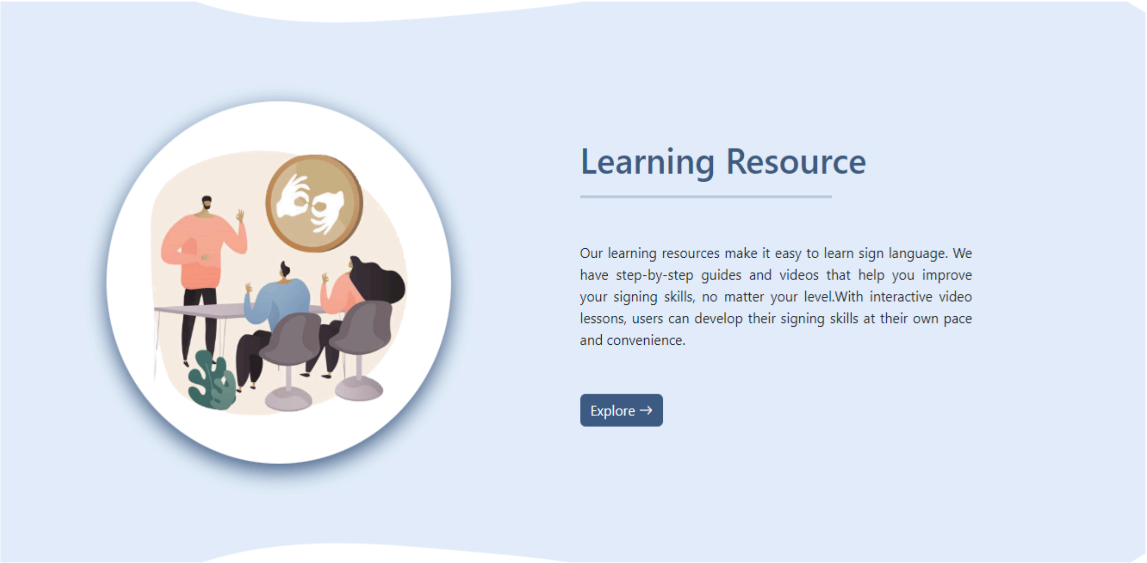
Welcome To Sign Language Recognition

Where communication knows no barriers. Our innovative web platform empowers deaf and mute individuals by seamlessly transforming sign language into text, fostering inclusive and accessible communication for all.

Figure 6 Home Page

1.2 Learning page

I have added the learning resources page where people can see each and every sign with the character tags. If some people does not have knowledge of hand sign language so, they can refer the learning page.



Learning Resource

Our learning resources make it easy to learn sign language. We have step-by-step guides and videos that help you improve your signing skills, no matter your level.With interactive video lessons, users can develop their signing skills at their own pace and convenience.

Figure 7 Learning Resources

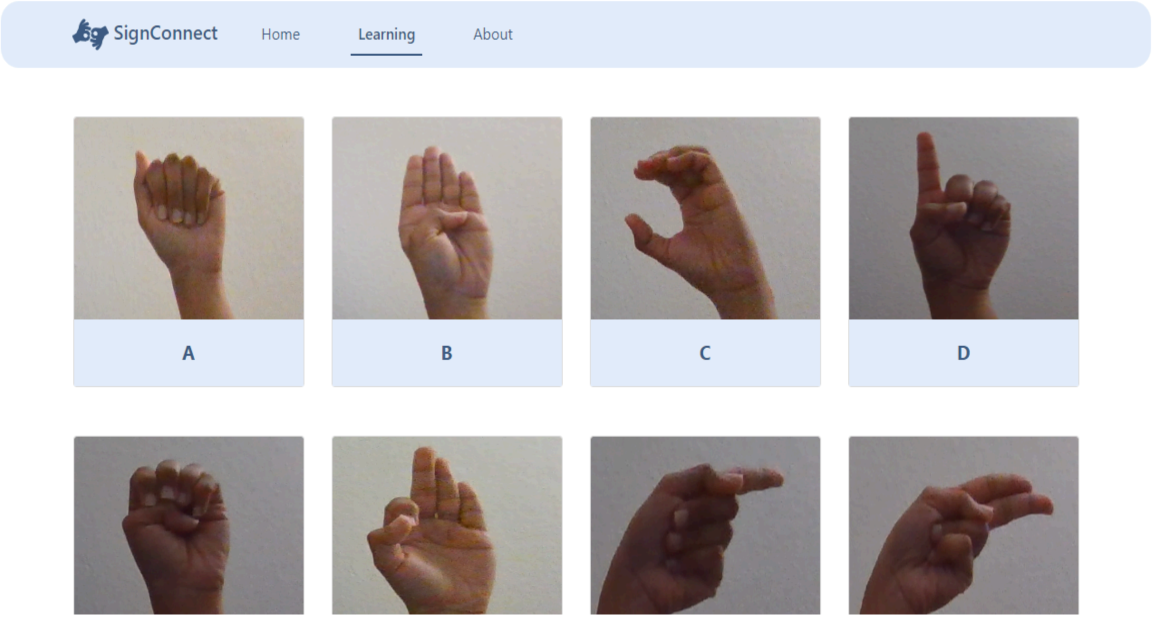
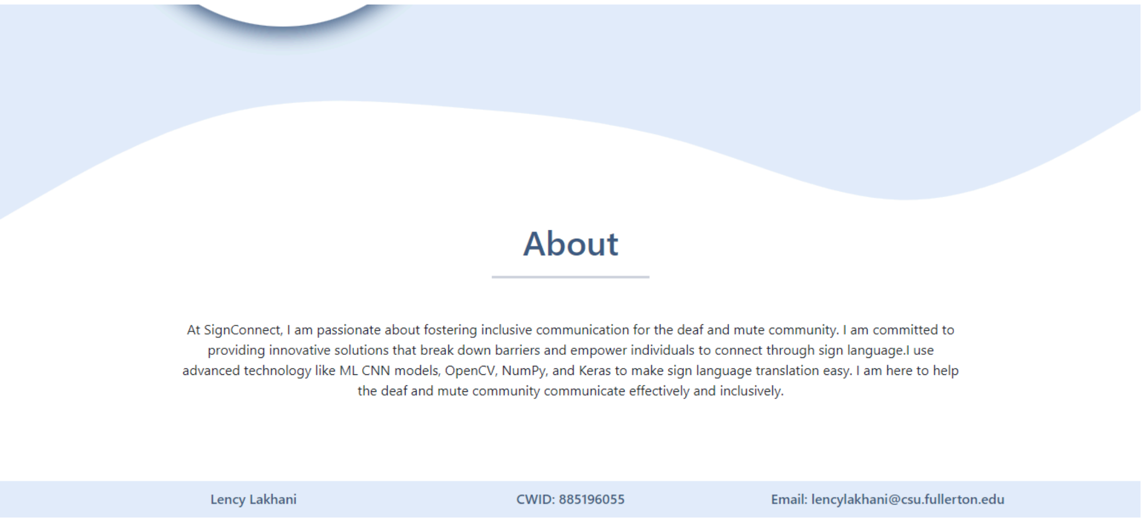


Figure 8 Learning Dataset

1.3 About Page

I have also included the about page of the web application where people can get an idea about how my web application works and what types of technology I have used it. I have also included my email, CWID and name for the inquiry about the web application. This page whole above images are the main landing page of my web application.



About

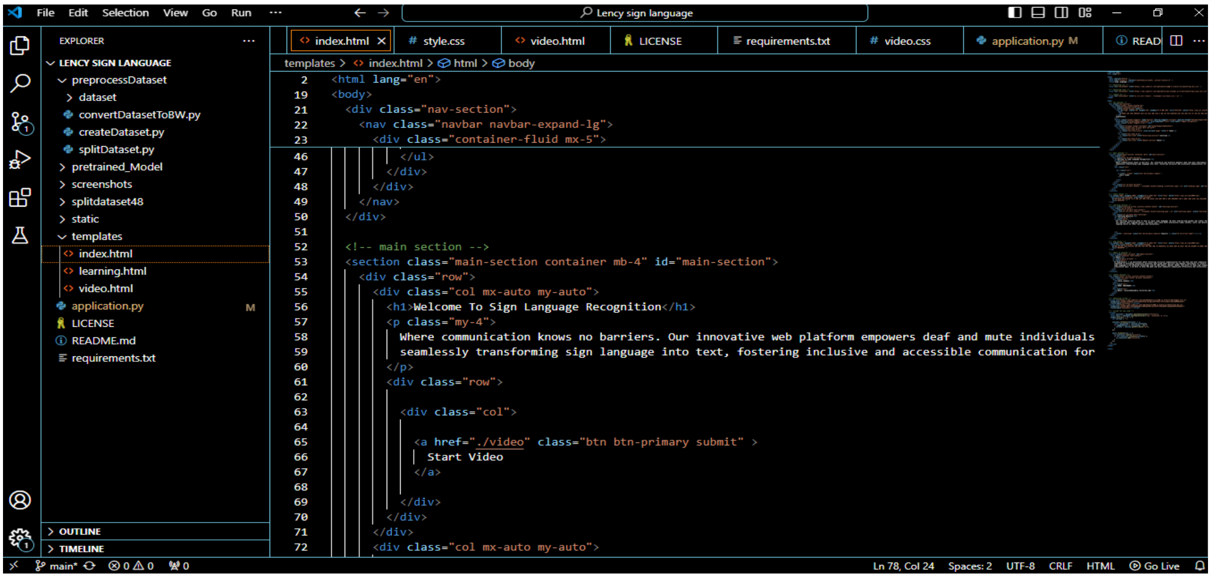
At SignConnect, I am passionate about fostering inclusive communication for the deaf and mute community. I am committed to providing innovative solutions that break down barriers and empower individuals to connect through sign language.I use advanced technology like ML CNN models, OpenCV, NumPy, and Keras to make sign language translation easy. I am here to help the deaf and mute community communicate effectively and inclusively.

Lency Lakhani

CWID: 885196055

Email: lencylakhani@csu.fullerton.edu

Figure 9 About Page



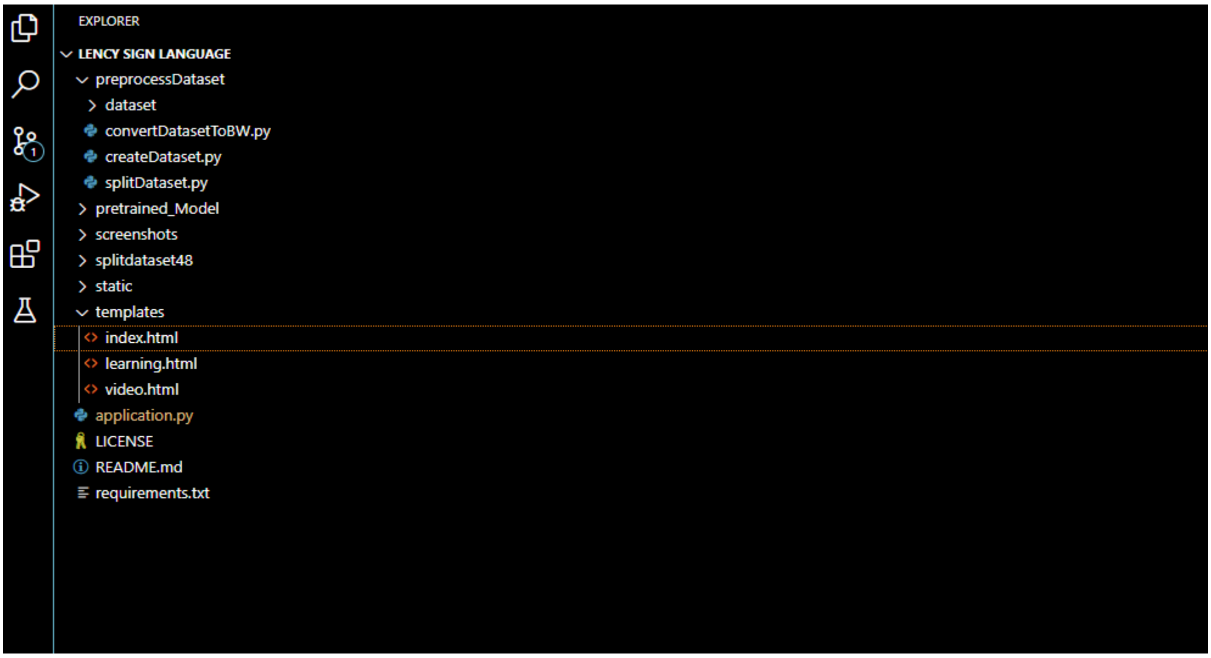
```
<html lang="en">
<body>
  <div class="nav-section">
    <nav class="navbar navbar-expand-lg">
      <div class="container-fluid mx-5">
          </ul>
        </div>
      </div>
    </nav>
  </div>

  <!-- main section -->
  <section class="main-section container mb-4" id="main-section">
    <div class="row">
      <div class="col mx-auto my-auto">
        <h1>Welcome To Sign Language Recognition</h1>
        <p class="my-4">
          Where communication knows no barriers. Our innovative web platform empowers deaf and mute individuals
          seamlessly transforming sign language into text, fostering inclusive and accessible communication for
        </p>
        <div class="row">

          <div class="col">

            <a href="./video" class="btn btn-primary submit" >
              Start Video
            </a>

          </div>
        </div>
      </div>
      <div class="col mx-auto my-auto">
```

Figure 10 Frontend Code

2. Backend Part

2.1 Source File Structure

I have added the image of the file structure that I use to build my web applications. I have given the proper folder name to each and every folder. So, users can get an idea which folder belongs to this functionality. First one is the preprocess dataset folder, which is not processed. It has to be processed in the future. To process the dataset I have created one folder which is convertDatasetToBW.py which helps me to convert colored image into the black and white gray scale image. CreateDatset.py file will be used to capture my hand sign images for future prediction. splitDataset.py will be used for spilt the dataset for the testing and training. I have also added the static and templates folder for the frontend part. Lately, application.py is the main file to run the web application.

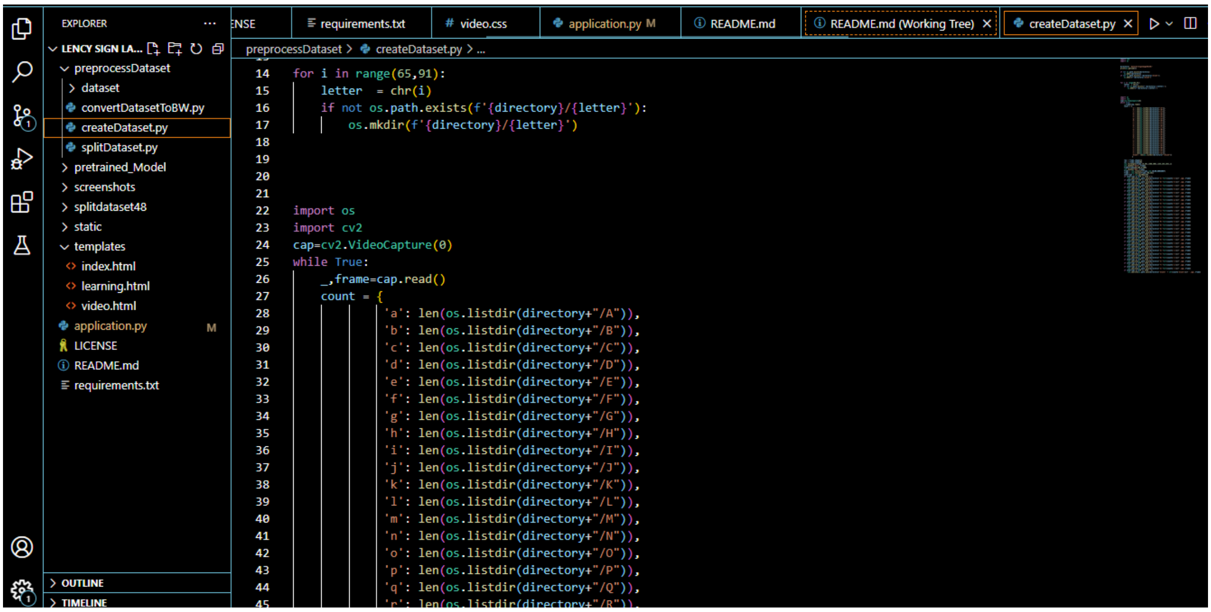


```
v LENCY SIGN LANGUAGE
  v preprocessDataset
    > dataset
    convertDatasetToBW.py
    createDataset.py
    splitDataset.py
  > pretrained_Model
  > screenshots
  > splitdataset48
  > static
  v templates
    index.html
    learning.html
    video.html
  application.py
  LICENSE
  README.md
  requirements.txt
```

Figure 11 File Structure

2.2 Create Dataset

I have created my own dataset for the accuracy purpose. I have used OpenCV python Library to capture real time images. Why I have used my own dataset? I have used my own dataset for the accuracy purpose because, In the machine learning prediction model, It will distract the edges of the hand and give predicted output form the testing dataset. If I am using my own dataset then, It will be predicted according to my hand edges and sizes.So, whenever I predict the result it will be give accurate predicted result in the given time frame. Here, I have taken 500 images of the each character from A-Z. so, approximately, I have taken 13,000 images for whole dataset. Then , I have used this dataset into the CNN model to train the model.

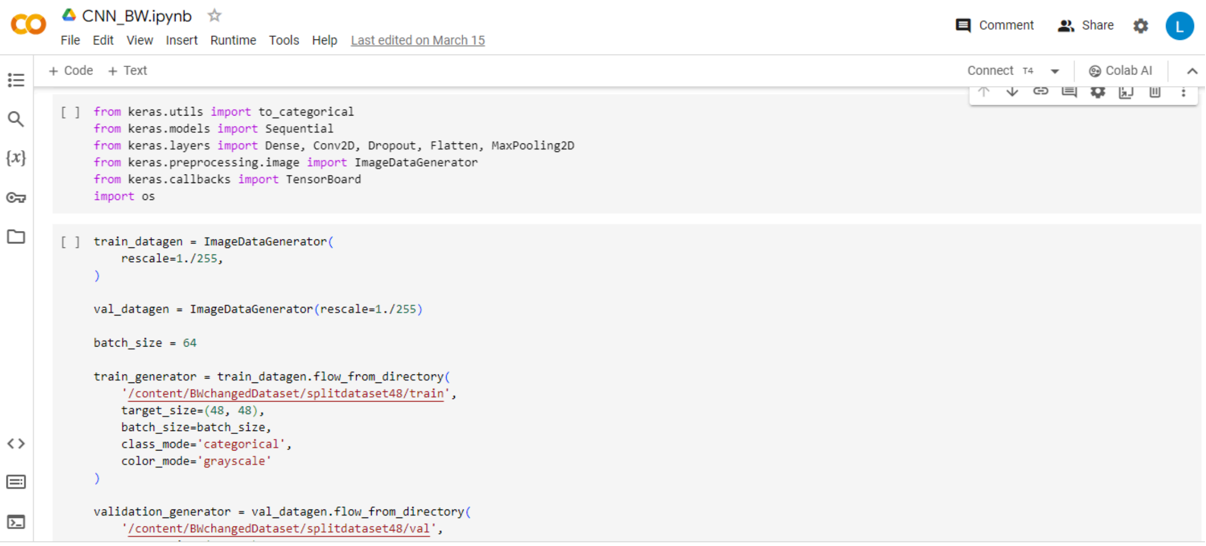


```
for i in range(65,91):
    letter  = chr(i)
    if not os.path.exists(f'{directory}/{letter}'):
        os.mkdir(f'{directory}/{letter}')



import os
import cv2
cap=cv2.VideoCapture(0)
while True:
    _,frame=cap.read()
    count = {
             'a': len(os.listdir(directory+"/A")),
             'b': len(os.listdir(directory+"/B")),
             'c': len(os.listdir(directory+"/C")),
             'd': len(os.listdir(directory+"/D")),
             'e': len(os.listdir(directory+"/E")),
             'f': len(os.listdir(directory+"/F")),
             'g': len(os.listdir(directory+"/G")),
             'h': len(os.listdir(directory+"/H")),
             'i': len(os.listdir(directory+"/I")),
             'j': len(os.listdir(directory+"/J")),
             'k': len(os.listdir(directory+"/K")),
             'l': len(os.listdir(directory+"/L")),
             'm': len(os.listdir(directory+"/M")),
             'n': len(os.listdir(directory+"/N")),
             'o': len(os.listdir(directory+"/O")),
             'p': len(os.listdir(directory+"/P")),
             'q': len(os.listdir(directory+"/Q")),
             'r': len(os.listdir(directory+"/R")),
```

Figure 12 Createdataset Code

2.3 Trained CNN Model

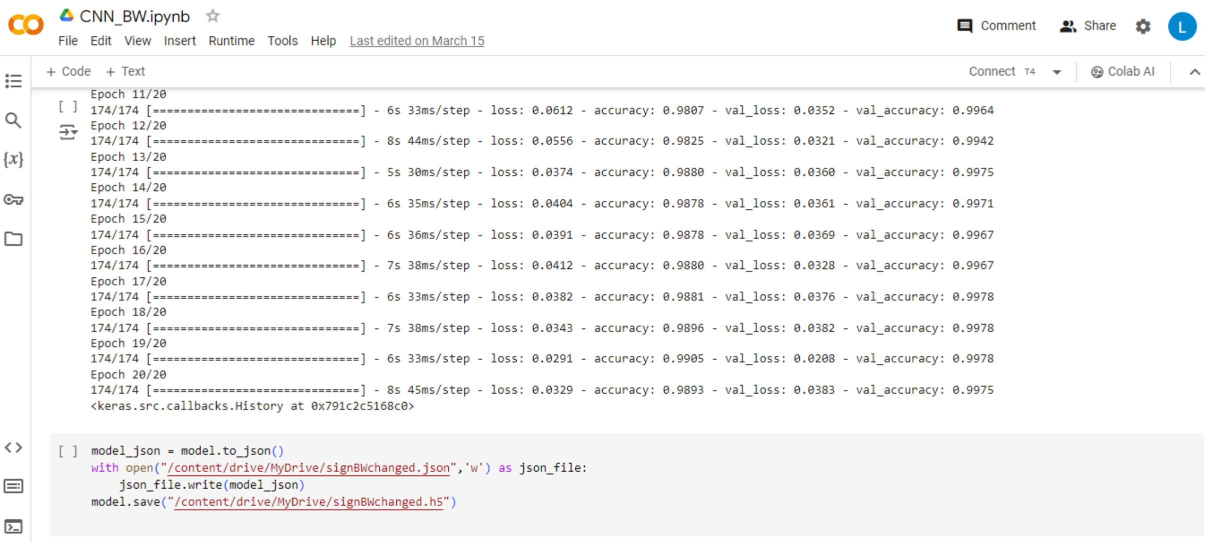
I have attached the images of CNN Model that I have already trained using the sign language dataset. I have used keras and tensorflow library to implement the CNN model. Tensorflow library will hepl to provide the feature map in training. From the CNN model, I have got 98% training accuracy and 99% validation accuracy.


CNN_BW.ipynb
File Edit View Insert Runtime Tools Help Last edited on March 15
Comment
Share
L
+ Code + Text
Connect T4
Colab AI

```
from keras.utils import to_categorical
from keras.models import Sequential
from keras.layers import Dense, Conv2D, Dropout, Flatten, MaxPooling2D
from keras.preprocessing.image import ImageDataGenerator
from keras.callbacks import TensorBoard
import os
```

```
train_datagen = ImageDataGenerator(
    rescale=1./255,
)

val_datagen = ImageDataGenerator(rescale=1./255)

batch_size = 64

train_generator = train_datagen.flow_from_directory(
    '/content/BWchangedDataset/splitdataset48/train',
    target_size=(48, 48),
    batch_size=batch_size,
    class_mode='categorical',
    color_mode='grayscale'
)

validation_generator = val_datagen.flow_from_directory(
    '/content/BWchangedDataset/splitdataset48/val',
```

Figure 13 CNN Trained Model


CNN_BW.ipynb
File Edit View Insert Runtime Tools Help Last edited on March 15
Comment
Share
L
+ Code + Text
Connect T4
Colab AI

```
Epoch 11/20
174/174 [==============================] - 6s 33ms/step - loss: 0.0612 - accuracy: 0.9807 - val_loss: 0.0352 - val_accuracy: 0.9964
Epoch 12/20
174/174 [==============================] - 8s 44ms/step - loss: 0.0556 - accuracy: 0.9825 - val_loss: 0.0321 - val_accuracy: 0.9942
Epoch 13/20
174/174 [==============================] - 5s 30ms/step - loss: 0.0374 - accuracy: 0.9880 - val_loss: 0.0360 - val_accuracy: 0.9975
Epoch 14/20
174/174 [==============================] - 6s 35ms/step - loss: 0.0404 - accuracy: 0.9878 - val_loss: 0.0361 - val_accuracy: 0.9971
Epoch 15/20
174/174 [==============================] - 6s 36ms/step - loss: 0.0391 - accuracy: 0.9878 - val_loss: 0.0369 - val_accuracy: 0.9967
Epoch 16/20
174/174 [==============================] - 7s 38ms/step - loss: 0.0412 - accuracy: 0.9880 - val_loss: 0.0328 - val_accuracy: 0.9967
Epoch 17/20
174/174 [==============================] - 6s 33ms/step - loss: 0.0382 - accuracy: 0.9881 - val_loss: 0.0376 - val_accuracy: 0.9978
Epoch 18/20
174/174 [==============================] - 7s 38ms/step - loss: 0.0343 - accuracy: 0.9896 - val_loss: 0.0382 - val_accuracy: 0.9978
Epoch 19/20
174/174 [==============================] - 6s 33ms/step - loss: 0.0291 - accuracy: 0.9905 - val_loss: 0.0208 - val_accuracy: 0.9978
Epoch 20/20
174/174 [==============================] - 8s 45ms/step - loss: 0.0329 - accuracy: 0.9893 - val_loss: 0.0383 - val_accuracy: 0.9975
<keras.src.callbacks.History at 0x791c2c5168c0>
```

```
model_json = model.to_json()
with open("/content/drive/MyDrive/signBWchanged.json",'w') as json_file:
    json_file.write(model_json)
model.save("/content/drive/MyDrive/signBWchanged.h5")
```

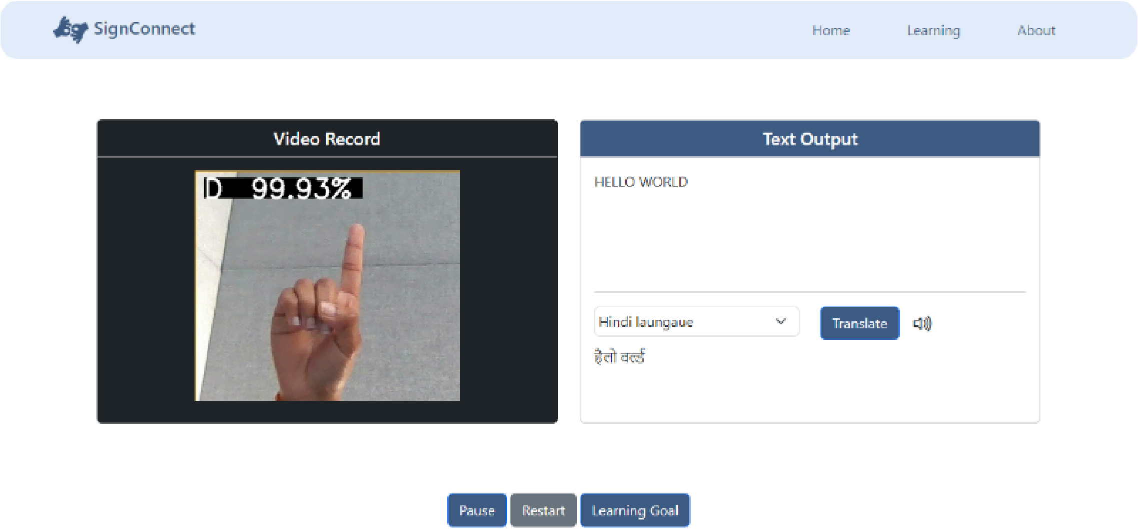
Figure 14 CNN model with training and testing accuracy

2.4 Integrate backend with frontend using Flask

I have integrated the backend with the frontend with the use of flask.I have used Javascript, Ajax and Jquery for support.

2.5 Prediction

Here, I have attached the images of the predicted output of my web application. This image is of my second video.html frontend page. Where I have integrated the backend CNN model with frontend. Here, Deaf and mute people do some sign in the front of the webcam and my CNN model will produce the output on the text output panel. As you can see here, I have done some signs for **"Hello World"** and I have translated that hello world into the **Hindi Language** so, in translation, it is reflected in hindi. Moreover, I have attached a voice translation option. Where user can click on microphone to activate the voice over functionality.

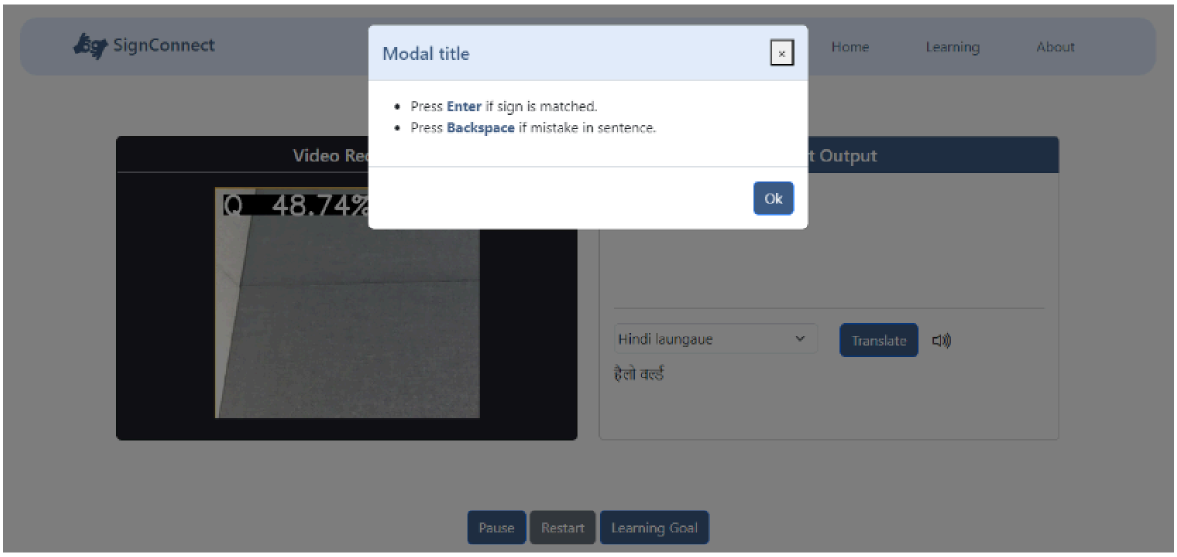


HELLO WORLD

हैलो वर्ल्ड

Figure 15 prediction output

In the above images, There are some three button at the bottom. I will give you the brief introduction of that button. Firstly, Pause button will turn on and turn off the video at any given time. Secondly, Reset button will reset everything from the text output panel. So, User have to start again from the scratch. Lastly, Learning resource will teach how to play with predicted sign and display onto the screen.



Modal title

- Press **Enter** if sign is matched.
- Press **Backspace** if mistake in sentence.

Hindi laungaue

हैलो वर्ल्ड

Figure 16 Learning Instructions

In the above image, when the user clicks on the learning goal button , the modal will be popped up with the instruction how predicted output will be displayed onto the screen. To predict the output onto the screen, users just have to click **“Enter key”** to append text output one by one. If there may be a mistake in the sentence, the user can press **“Backspace Key”** to delete the character for the end of the sentences.

Risk and Mitigation

I discovered three major risks that can be problematic for my project. I will also give the optimised solution for these risks. The following risks are defined below.

1. **Environmental concerns**

   There may be a chance to capture blur images due to multiple lights reflected on the screen. From this distractiontion , I need to find some innovative solution. To optimise this problem, I am planning to add some pre-processing techniques which can overcome these hurdles.

   Firstly, I will apply some background subtraction technique in which it first isolates the hand and subtracts the effect of lights. Moreover, I will use some normalisation technique in which, image will be normalised from the colour and intensity of the image. I will also implement the hand detection and edge detection technique in which it counters the edges of the hand and distract the background. By using this approach, I will get some positive feedback towards the web application reliability and validity.

2. **Sign Boundary Detection**

   This problem arises when some person is doing one hand sign gesture and suddenly he changes his first sign to another sign. It will create some confusion for the CNN model to which sign I have to recognize first. To resolve this problem, we need to use a continuous approach of prediction and detection.

To resolve this issue, we need to do real time data analysis. In this process, it will capture the continuous frames of an image in real time. Instead of relying on 10-12 frames of images, we need to take a real time frame for the image which can continuously predict the data. By adapting this technology, we can give a responsive and adaptive model with accuracy.

3. **Overfitting**

Adaptiveness of the new sign will be the main concern for this model. When the model is dealing with new and unseen signs, it will not be able to recognize what is this and from where this overfitting issue comes. To resolve this issue, I am planning to use some data augmentation technique in which , CNN model has been trained by some various hand gestures with multiple lights, some blurry effects etc so, it can predict well. By using this technique, responsiveness of the web application will be increased.

Project Plan

1. Research on machine learning algorithms
   - ***Time Duration:*** 2-3 weeks
   - ***Hours:*** 20
   - ***Project Progress:*** 11.49%

2. Design an architecture of a machine learning model and research on dataset
   - ***Time Duration:*** 4 weeks
   - ***Hours:*** 38
   - ***Project Progress:*** 21.83%

3. Design a user interface
   - ***Time Duration:*** 2-3 weeks
   - ***Hours:*** 20
   - ***Project Progress:*** 11.49%

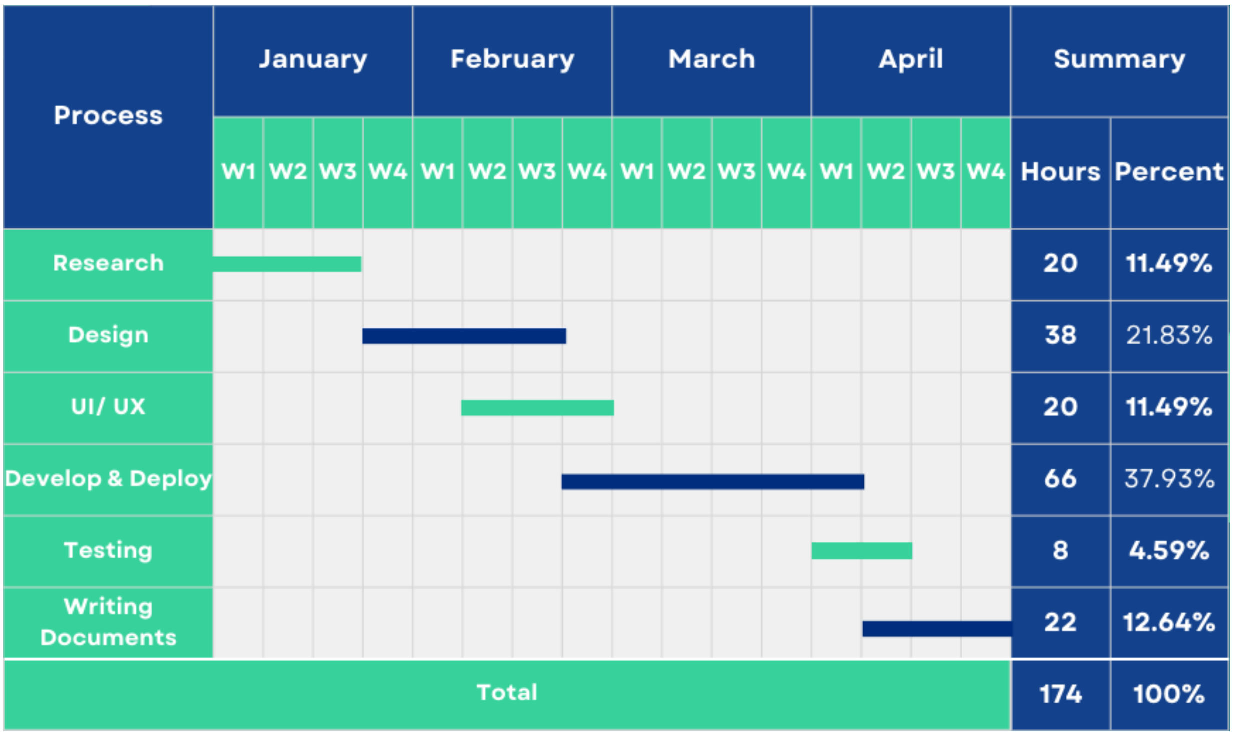
4. Develop and deploy real-time application
   - ***Time Duration:*** 6 weeks
   - ***Hours:*** 66
   - ***Project Progress:*** 37.93%

5. Testing and fixing application errors
    - ***Time Duration:*** 2 weeks
    - ***Hours:*** 8
    - ***Project Progress:*** 4.59%

6. Writing documents
    - ***Time Duration:*** 2-3 weeks
    - ***Hours:*** 22
    - ***Project Progress:*** 12.64%

Total of 174 hours will be given to this project to accomplish in the month end of april.

Project Schedule



Process	January				February				March				April				Summary	
	W1	W2	W3	W4	W1	W2	W3	W4	W1	W2	W3	W4	W1	W2	W3	W4	Hours	Percent
Research	■	■	■														20	11.49%
Design				■	■	■	■										38	21.83%
UI/ UX						■	■	■									20	11.49%
Develop & Deploy								■	■	■	■	■	■				66	37.93%
Testing													■	■			8	4.59%
Writing Documents														■	■	■	22	12.64%
Total																	174	100%

Figure 17 Project Schedule

As per the above schedule, I will give more time to develop and deploy my real-time application because for development, I have to develop a machine learning model as well as integrate that model with my application, where I can access images of user's hands through a webcam. Alternatively, I plan to allocate minimum time to the testing phase since I consistently assess the accuracy of the machine learning model throughout the development phase. So, we don't need to spend extra time on testing at a later stage.

Installation Instruction

1. **Clone the Repository**

   git clone https://github.com/LencyLakhani/SignToText_Using_CNN.git

2. **Install all the dependencies**

   Flask==3.0.2
   keras==2.13.1
   numpy==1.24.3
   opencv_python==4.9.0.80

   Also, You can see the dependencies form: pip install -r requirements.txt

3. **Run the Flask application**

   python application.py

4. **Access the application through the local web browser**

   `http://localhost:5000`

Recommendation For enhancement

I got a recommendation from one of my friend to include Risk and mitigation, project schedule and planning into the project report.

Future Work

Addressing Environmental Concerns: While the implemented pre-processing techniques provide a foundation for mitigating environmental challenges, future endeavors could explore advanced computer vision algorithms and machine learning models to dynamically adapt to varying lighting conditions and environmental factors. Integrating sensor technologies or depth-sensing cameras may offer more robust input data for improved gesture recognition.

Advancing Real-time Sign Boundary Detection: The adoption of a continuous approach to prediction and detection via real-time data analysis shows promise in resolving sign boundary detection issues. Further research could focus on optimizing the efficiency and accuracy of real-time processing algorithms. Leveraging parallel processing or hardware acceleration techniques could achieve low-latency gesture recognition. Exploring novel approaches like dynamic time warping or sequence-to-sequence models may enhance the model's capability to handle complex and dynamic sign sequences.

Mitigating Overfitting with Data Augmentation: While data augmentation techniques can help alleviate overfitting and enhance model generalization, continuous efforts are necessary to expand and diversify the training dataset. Future work may involve collecting additional annotated data from diverse sources and populations, encompassing variations in hand shapes, skin tones, and gestures. Exploring semi-supervised or unsupervised learning approaches could enable the model to leverage unlabeled data for further improvement, enhancing its adaptability to unseen sign gestures.

In addition to these technical advancements, future work should also focus on user feedback and usability studies to ensure that the developed solution meets the practical needs and expectations of the deaf and mute community. Collaborating with stakeholders and domain experts will be essential for refining the system and maximizing its impact on improving communication accessibility. By incorporating both technological innovations and user-centric design principles, the project can continue to evolve towards a more inclusive and empowering solution for sign language recognition and communication.

By delving into these areas in future research and development, the project can evolve towards a more resilient, adaptable, and inclusive solution for sign language recognition and communication.

Conclusion

In conclusion, this project represents a significant step forward in leveraging technology to bridge communication gaps for the deaf and mute community. By combining computer vision techniques with deep learning methodologies, we have developed a robust solution capable of accurately converting sign language gestures into text.

The utilization of Convolutional Neural Networks (CNNs) has enabled us to train a model that can effectively recognize and interpret hand gestures, paving the way for more accessible communication channels. Moreover, our focus on user-centric design has led to the creation of an intuitive and interactive user interface, ensuring ease of use for individuals interacting with the system.

Moving forward, this project holds immense potential for further enhancements and real-world applications. With continued refinement and optimization, our solution can play a pivotal role in empowering individuals with hearing and speech impairments, fostering inclusivity, and enabling seamless communication in diverse settings. Ultimately, this endeavor underscores the transformative power of technology in addressing societal challenges and promoting equality and accessibility for all.

References

1. *National Deaf Center* (2023, November 3). https://nationaldeafcenter.org/faq/how-many-deaf-people-live-in-the-united-states/

2. Brandon Garcia, Sigberto Alarcon Viesca. (2016). “*Real-time American Sign Language Recognition with Convolutional Neural Networks*”. Stanford University CS231n: Deep Learning for Computer Vision. https://cs231n.stanford.edu/reports/2016/pdfs/214_Report.pdf

3. Xin, M., Wang, Y. (2019) Research on image classification model based on deep convolution neural network. *J Image Video Proc*. https://doi.org/10.1186/s13640-019-0417-8

4. Thad Starner and Alex Pentland. (1996). *Real-Time American Sign Language Recognition from Video Using Hidden Markov Models*. https://cdn.aaai.org/Symposia/Fall/1996/FS-96-05/FS96-05-017.pdf

5. Mahamkali, N., & Ayyasamy, V. (2015). OpenCV for Computer Vision Applications. *ResearchGate*. https://www.researchgate.net/publication/301590571_OpenCV_for_Computer_Vision_Applications
6. https://marketsplash.com/how-to-integrate-flask-with-front-end-frameworks/