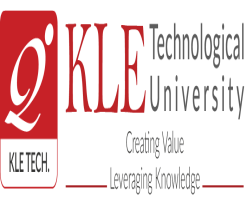
KLE Society's

KLE Technological University



**A MiniProject Report**

**On**

**“Sign Language Recognition using Machine Learning”**

Submitted in partial fulfillment of the requirement for the degree of

Master of Computer Applications

Submitted By

### Saquib A Khazi (01FM19MCA006)

### Priyanka Rokhade (01FM19MCA014)

Under the guidance of

**Prof. Shivanand Seeri**

**Prof. Ashok Chikaraddi**

MASTER OF COMPUTER APPLICATIONS

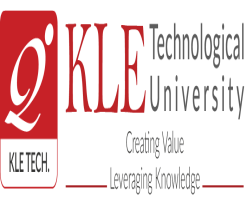
HUBLI–580 031 (India).

Academic year 2021-2022

KLE Society's

KLE Technological University

2021 - 2022



MASTER OF COMPUTER APPLICATIONS

**Certificate**

This is to certify that Minor Project entitled “Sign Language Recognition using Machine Learning” is a bonafied work carried out by the student team Mr. Saquib A Khazi – SRN 01FM19MCA006, Ms. Priyanka Rokhade – SRN 01FM19MCA014, in partial fulfillment of completion of Fourth semester Master of Computer Applications during the year 2021 – 2022. The project report has been approved as it satisfies the academic requirement with respect to the project work prescribed for the above said programme.

|  |  |  |  |
| --- | --- | --- | --- |
| Prof. Shivanand Seeri  Prof. Ashok Chikaraddi | Dr. P. R. Patil M.E, PhD | | Prof. N. H. Ayachit |
| Project Guides,  MCA Dept | Head of Department,  Professor,  MCA Dept. | | Registrar,  KLE Technological University |
|  | ***Viva-Voce Examination*** | |  |
| Name of the Examiners |  | | Signature with Date |
| 1. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | |  | :\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |
| 1. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | |  | :\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |

**ACKNOWLEDGEMENT**

Every successful completion of any undertaking would be complete only after we remember and thank the almighty, the parents, the teachers, and the personalities, who directly or indirectly helped and guided during the execution of that work. The success of this work is equally attributed to all well-wishers who have encouraged and guided throughout the execution.

We express our deepest gratitude to **Prof. Shivananda Seeri, Prof. Ashok Chikaraddi and Prof. P R Patil** for their guidance and assistance throughout the project with great interest.

We avail this opportunity to express deepest gratitude to **Dr. P. R. Patil, HOD** for encouraging us with his valuable guidelines.

We are grateful to **Prof. N. H. Ayachit,** Registrar for his blessings and **KLE Technological University, Hubballi** that has given us a bright future. We would like to thank all the people for their guidance and valuable suggestions throughout the project.

We also thank all teaching and non-teaching staff members of the MCA department for their invaluable cooperation.

Finally, we would like to express our sincere thanks to our **Parents and Friends** for their enormous encouragement and all other who have extended their helping hands towards completion of our project.

Mr. Saquib A Khazi

Ms. Priyanka Rokhade

**ABSTRACT**

*Sign language is a method of communication which uses various hand gestures and movements. Understanding these gestures can be postulated as a pattern recognition problem. Humans use different kinds of gestures and motions to convey different messages to other humans. This project represents a framework for a human computer interface capable of recognizing said gestures from sign language and providing a text output representing the meaning of the gesture. The proposed system will use convolutional neural networks and long short-term memory networks to identify and learn the gestures which will help to minimize the communication barrier between signers and non-signers.*

**SIGN LANGUAGE RECOGNITION**

**USING**

**MACHINE LEARNING**

**INDEX**

**TABLE OF CONTENTS PAGE NO**

1. **INTRODUCTION**…………………………………………………….…..1

1.1 Project Overview

1.2 Motivation

1.3 Objectives of the project

* 1. Problem Definition

1. **PROPOSED SYSTEM**…………………………………………………..2-3

2.1 Description of Proposed System.

2.2 Description of Target Users

2.3 Advantages of Proposed System

2.4 Scope

1. **SOFTWARE REQUIREMENT SPECIFICATION**…………………..4-7

3.1 Overview of SRS

3.2 Requirement Specifications

3.2.1 Functional Requirements

3.2.2 Use case diagrams

3.2.3 Use Case descriptions using scenarios

3.2.4 Nonfunctional Requirements

3.2.4.1 Performance requirements

3.2.4.2 Safety requirements

3.3 Software and Hardware requirement specifications

3.4 GUI of proposed system

1. **SYSTEM DESIGN**……………………………………...……………….8-11

4.1 Architecture of the system

4.2 Level 0 DFD

4.3 Detailed DFD for the proposed system

4.4 Sequence Diagram

1. **IMPLEMENTATION**…………………………………………………..12-17
   1. Proposed Methodology
   2. Description of Modules and code
2. **TESTING** ………………………………………………………….……18
3. **RESULTS & DISCUSSIONS**…………………………..………………19-21
4. **CONCLUSION AND FUTURE SCOPE** ……………………………....22
5. **REFERENCES/BIBLIOGRAPHY**……………………………………..23

**1. Introduction**

**1.1. Overview of the project**

The goal of this project was to build a neural network able to classify which letter of the American Sign Language (ASL) alphabet is being signed, given an image of a signing hand. This project is a first step towards building a possible sign language translator, which can take communications in sign language and translate them into written and oral language. Such a translator would greatly lower the barrier for many deaf and mute individuals to be able to better communicate with others in day-to-day interactions.

**1.2. Motivation**

Sign language is learned by deaf and dumb, and usually it is not known to normal people, so it becomes a challenge for communication between a normal and hearing-impaired person. Its strike to our mind to bridge the between hearing impaired and normal people to make the communication easier. Sign language recognition (SLR) system takes an input expression from the hearing-impaired person gives output to the normal person in the form text or voice.

**1.3. Objectives of the project**

● Communication is always having a great impact in every domain and how it is considered the meaning of the thoughts and expressions that attract the researchers to bridge this gap for every living being.

● The objective of this project is to identify the symbolic expression through images so that the communication gap between a normal and hearing-impaired person can be easily bridged.

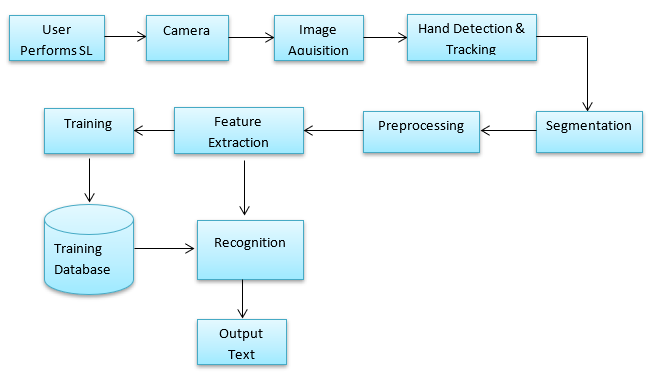
**1.4. Problem definition**

To design and develop an ML project for deaf and dumb people for communicating with the normal people.

We are designing this project to make the communication easier to deaf and dumb people. Given a hand gesture, implementing such an application which detects pre-defined American sign language (ASL) in a real time through hand gestures and providing facility for the user to be able to store the result of the character detected in a txt file, also allowing such users to build their customized gesture so that the problems faced by persons who aren’t able to talk vocally can be accommodated with technological assistance and the barrier of expressing can be overshadowed.

**2. Proposed system**

**2.1Description of proposed system with simple block diagram**



• Data Processing: The load data.py script contains functions to load the Raw Image Data and save the image data as numpy arrays into file storage. The process data.py script will load the image data from data.npy and preprocess the image by resizing/rescaling the image, and applying filters and ZCA whitening to enhance features. During training the processed image data was split into training, validation, and testing data and written to storage. Training also involves a load dataset.py script that loads the relevant data split into a Dataset class. For use of the trained model in classifying gestures, an individual image is loaded and processed from the filesystem.

• Training: The training loop for the model is contained in train model.py. The model is trained with hyperparameters obtained from a config file that lists the learning rate, batch size, image filtering, and number of epochs. The configuration used to train the model is saved along with the model architecture for future evaluation and tweaking for improved results. Within the training loop, the training and validation datasets are loaded as Dataloaders and the model is trained using Adam Optimizer with Cross Entropy Loss. The model is evaluated every epoch on the validation set and the model with best validation accuracy is saved to storage for further evaluation and use. Upon finishing training, the training and validation error and loss is saved to the disk, along with a plot of error and loss over training.

• Classify Gesture: After a model has been trained, it can be used to classify a new ASL gesture that is available as a file on the filesystem. The user inputs the filepath of the gesture image and the test data.py script will pass the filepath to process data.py to load and preprocess the file the same way as the model has been trained.

**2.2. Description of Target users**

In this project the user will have functionalities such as adding new sign language, deleting unspecific sign language. Sign language recognition using machine learning creates new sign using hand gesture and it trains the model for user hand recognition and then it displays the sign using users hand gesture.

**2.3. Advantages/applications of proposed system**

* The purpose of Sign Language Recognition (SLR) systems is to provide an efficient and accurate way to convert sign language into text or voice has aids for the hearing impaired for example, or enabling very young children to interact with computers (recognizing sign language), among others.
* Sign languages are an extremely important communication tool for many deaf and hard-of-hearing people. Sign languages are the native languages of the Deaf community and provide full access to communication.

**2.4. Scope**

One of the solutions to communicate with the deaf-mute people is by using the services of sign language interpreter. But the usage of sign language interpreters could be expensive. Cost-effective solution is required so that the deaf-mute and normal people can communicate normally and easily. Our strategy involves implementing such an application which detects pre-defined American Sign Language (ASL) through hand gestures. For the detection of movement of gesture, we would use basic level of hardware component like camera and interfacing is required. This application will comprise of two core module one is that simply detects the gesture and displays appropriate alphabet. The second is after a certain amount of interval period the scanned frame would be stored into buffer so that a string of character could be generated forming a letter.

**3. Software Requirement Specification**

**3.1Overview of SRS**

The main aim of our project is to improve the communication between normal person and dumb people. The goal of this document is to illustrate and explain the development of our software. Moreover, it will give a detailed description of system constraints and interaction with users. To do so, we developed software that is capable of recognizing sign language gestures, and training our model and sending the translation to a correspondent user.

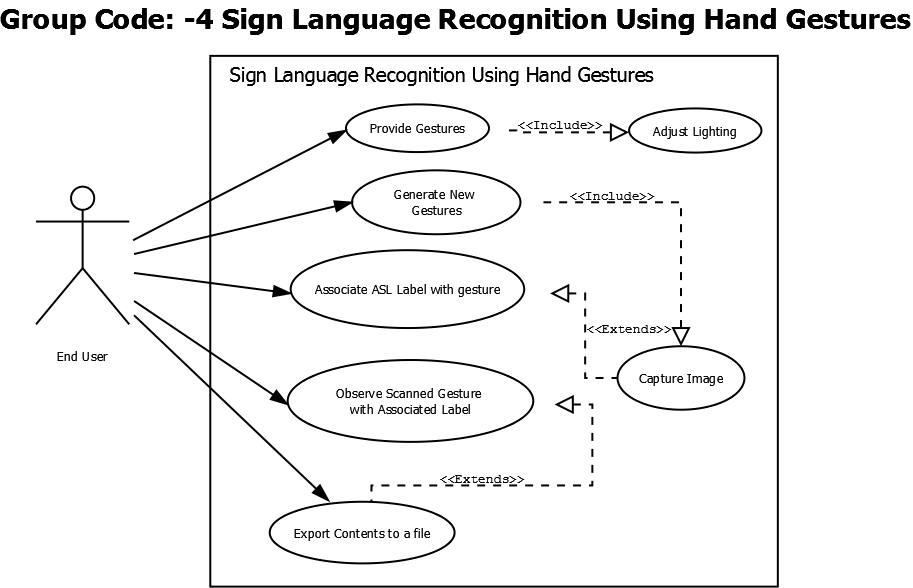
**3.2Requirement Specifications**

* User Interface **–** There is only one type of user that is application user.
* Application User **–** The application user will be able to add new sign language by training the model. On the execution time on GUI the sign language matches the trained data the output will be displayed.

**3.2.1 Functional Requirements**–

* Gesture Recognition: Providing hand gesture to the system camera and software should automatically recognize the gesture through the video input and gives output as a digital alphabet.
* Authentic representation: Here the software authenticate the given input, and should give the correct meaning of the gesture as a output.
* Cross platform support: If we run the Software on any platform it should run effectively and should provide appropriate output.

**3.2.2 Use Case diagram**



**Fig shows Use Case diagram of Sign Language Recognition Using ML**

**3.3.3 Use Case descriptions using scenarios**

Use case diagrams are usually referred to as behavior diagrams used to describe a set of actions that some systems (subjects) should or can perform in collaboration with one or more external users (actors) of the system. Each user should provide some observable and valuable result to the actors or other stakeholders of the system. Use case diagrams are twofold - they are both behavior diagrams, as the describe the behavior of the system, and also structure diagrams - as a special case of class diagrams where classifiers are restricted to be either actors or use cases related to each other with association. Use case diagrams are used to specify:

* External requirements, required usage of a system under design or analysis (subject) - what the system is supposed to do.
* The functionality offered by the subject - what the system can do
* Requirements the specified subject poses on its environment - by defining how environment should interact with the subject so that it will be able perform its services.
* Here the users provide gesture through hand in front of camera, thus it will generate new gesture and stores data in the system.
* The ASL gesture language is stored in system with label and then undergoes CNN algorithm process.
* Captured images are then processed and hence displayed on screen.

**3.3.4 Non-functional Requirements**

**3.3.4.1 Performance requirements**

* To run the application users will require libraries, Spyder application and strong internet connection.

**3.3.4.2 Safety requirements**

* Sign language recognition will not affect any other application installed on the user system. It can not cause any damage to the computer or its internal components.

**3.4 Software and hardware requirement specification**

Software Requirements

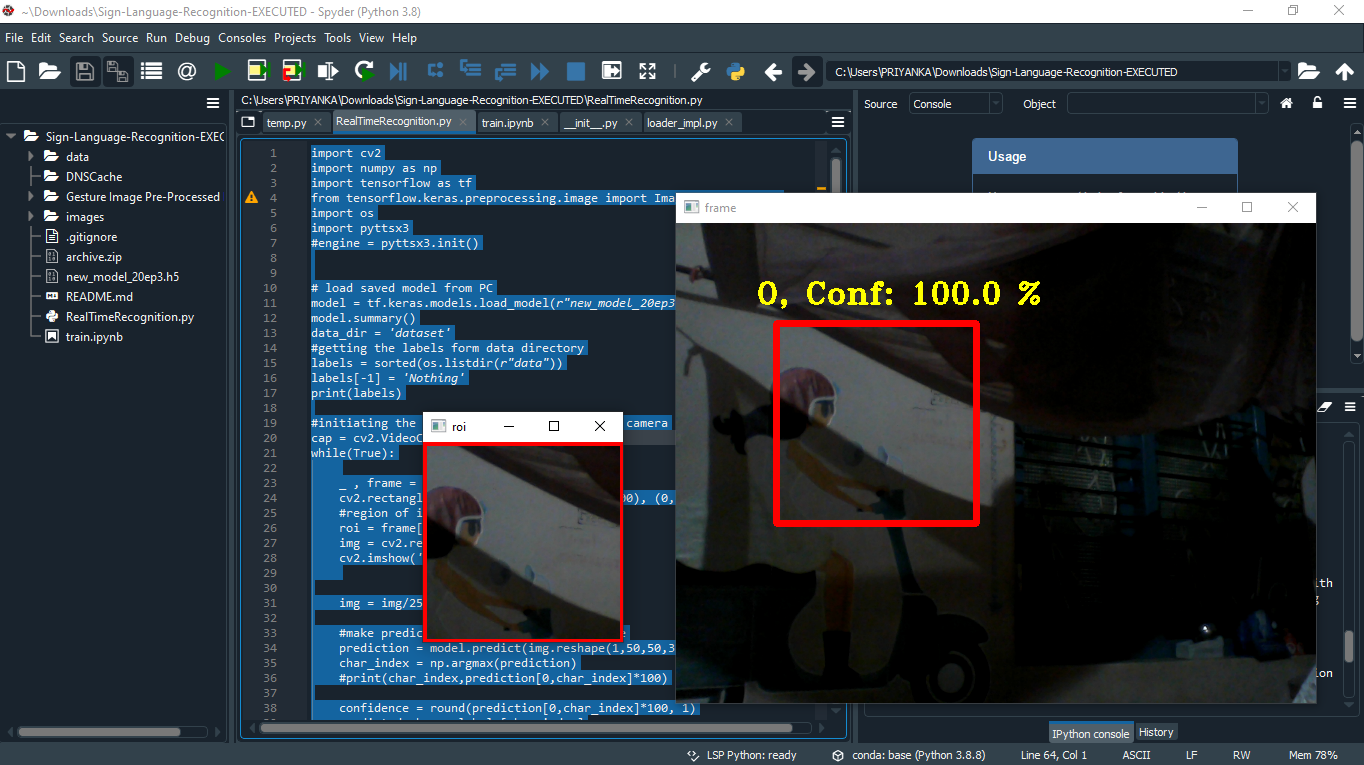
* Programming: Python, OpenCV, TensorFlow, NumPy, Matplotlib, Keras
* Operating System: Windows 10

Hardware Requirements

* Processor: 2.6 GHz or faster
* RAM: 4 GB or higher
* HDD: 2 GB available disk space
* GPU: NVIDIA GTX 1060 or higher or any AMD equivalent card
* Camera: At least 5MP

**3.5 GUI of proposed system**

Our project gives a representation of graphical user interface using GPU and it displays sign language in the screen with accuracy.



**Fig shows GUI of Proposed System**

**4. System design**

**4.1 Architecture of system**

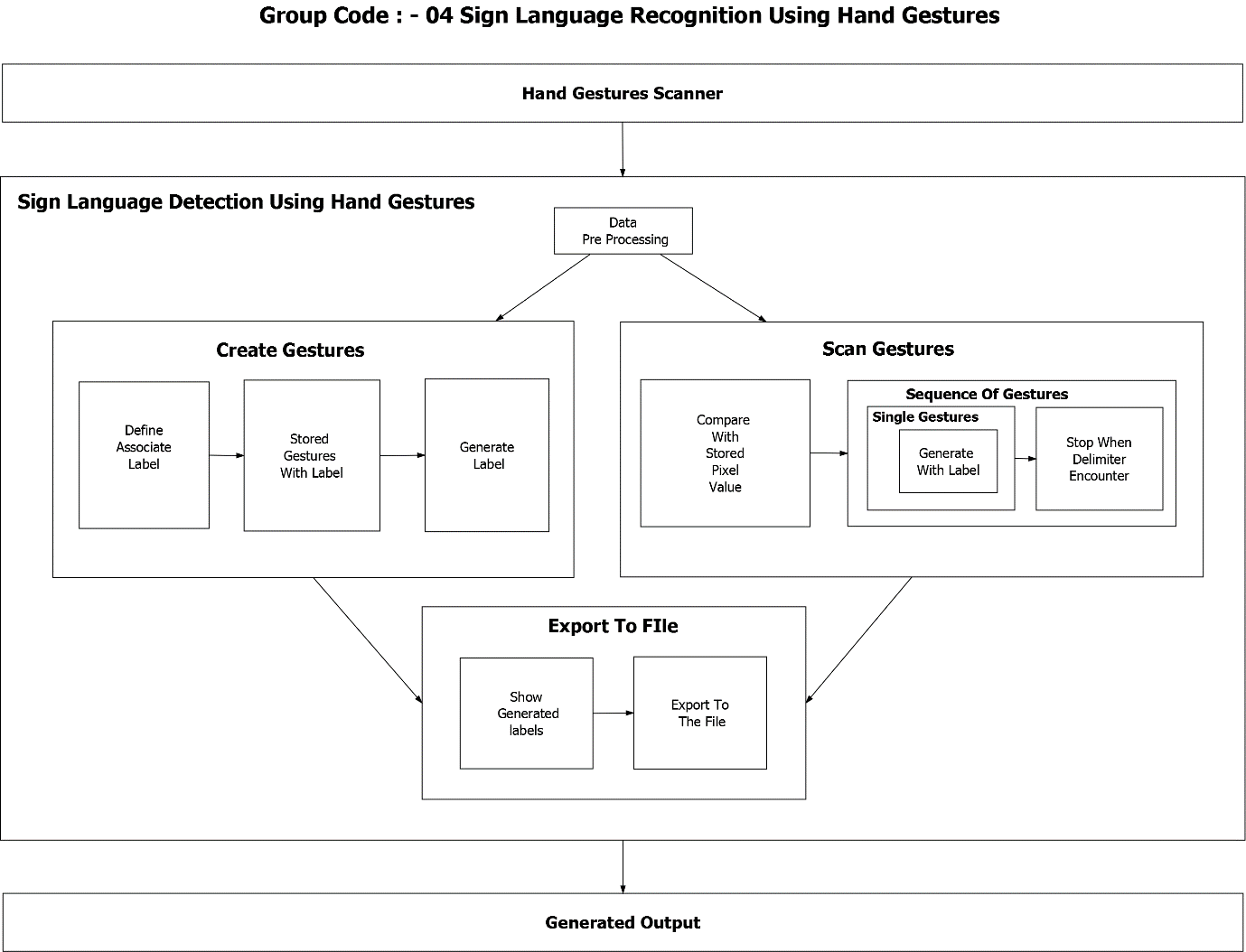
An early stage of the system design process.

• Represents the link between specification and design processes.

• Often carried out in parallel with some specification activities.

• It involves identifying major system components and their communications.

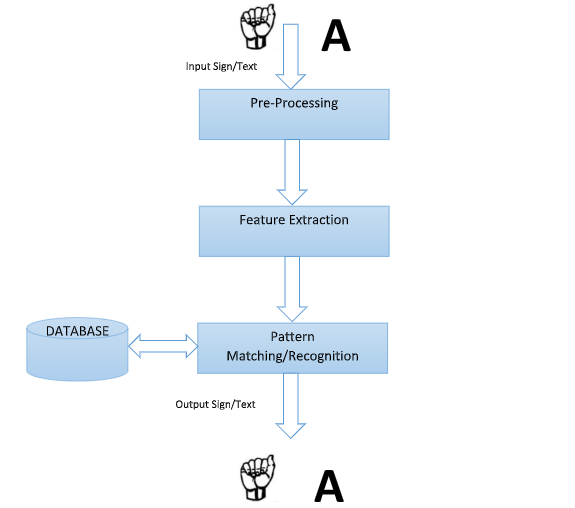
Simple, informal block diagrams showing entities and relationships are the most frequently used method for documenting software architectures.

****

**Fig 4.1 represents the architecture design.**

**4.2 Level 0 DFD**

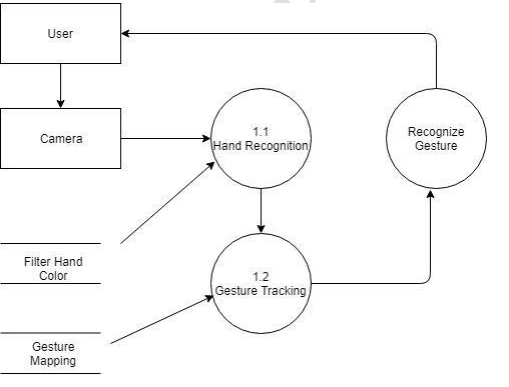
In this level 0 data flow diagram, the whole system is represented with the help of input, processing and output. The input to the gesture recognition system is the live feed from the camera which contains the gestures performed by the user. The camera provides the frames which can be mapped to their corresponding meanings.



**Fig 4.2 shows level 0 DFD**

**4.3 Detailed DFD**

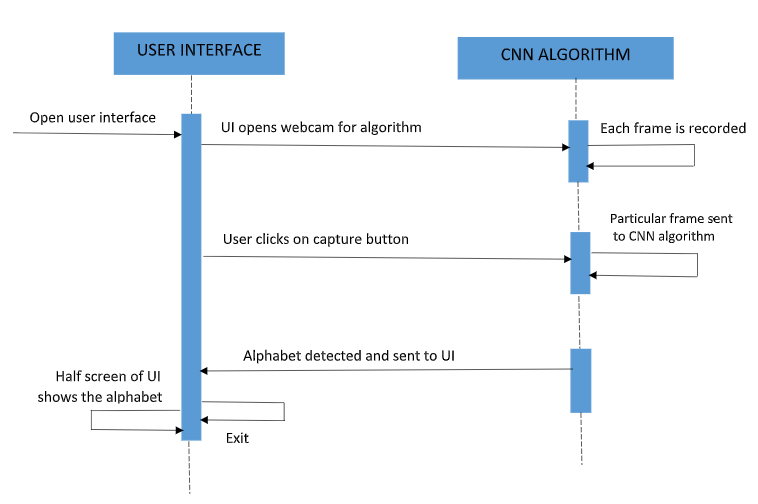
In this data flow diagram, the gesture recognition module is explained in further detail. The camera provides live feed of the user actions. Operations are performed to enhance the hand movements. The hand movement is recognized and sent to the gesture tracking module. The gesture tracking module checks for the gesture in the pre trained network of gestures and their respective meanings. The gesture control module maps the gesture to its meaning. Thus in these processes the gesture is recognized and meaning is displayed.



**Fig 4.3 shows detailed DFD**

**4.4 Sequence Diagram**

A sequence diagram is a type of interaction diagram because it describes how—and in what order—a group of objects works together. These diagrams are used by software developers and business professionals to understand requirements for a new system or to document an existing process.



**Fig 4.6 represents the Sequence diagram.**

**5. Implementation**

**5.1 Proposed methodology**

The proposed algorithm consisted of four major steps which are namely Image Acquisition, Feature Extraction, Orientation Detection and Gesture Recognition.

All of the following steps are explained in details in the later part of the paper with all the information on how the module is working and what behavior the module is supposedly expected to portray. While deciding on the following algorithm it was observed that pre-processing steps that are to be applied on the images for removal of noise in the background was not at all required and the approach was concluded to be simple and easy to implement. The steps of the methodology are further explained in details:

Image Acquisition

Scale space Extreme Detection

Image Acquisition

Scale space Extreme Detection

Key point localization

Gesture Recognition

Feature Extraction

Orientation Detection

Orientation Assignment

Key point Descriptor

1. Flow chart of Proposed Sign Language Recognition System

Image Acquisition

The first step of Image Acquisition as the name suggests is of acquiring the image during runtime through integrated webcam and while acquiring. The images will be stored in the directory as soon as they are captured and the recently captured image will be acquired and will be compared with the images stored for specific letter in the database using the algorithm and the comparison will give the gesture that was done and the translated text for the following gesture. The images will be captured through basic code of opening a webcam and then capturing the image through frames per second which will be stored in another directory where all the inputs images are stored in another directory and the recent captured image is picked up and the comparison with given set of images are made.

Gesture of letters shown here is used for testing the recognition algorithm.

A

B

C

D

F

F

G

Key point Localization. This stage attempts to eliminate more points from the list of key points by finding those that have low contrast or are poorly localized on an edge. This is achieved by calculating the Laplacian. The location of extremum, z, is given by:

Gestures for different Signs=21 2

Feature Extraction

For any object there are many features, interesting points on the object that can be extracted to provide a "feature" description of the object image features provide a set of features of an object that are not affected by many of the complications experienced in other methods, such as object scaling and rotation. For image feature generation, takes an image and transforms it into a "large collection of local feature vectors". Each of these feature vectors is invariant to any scaling, rotation or translation of the image.

Scale-Space Extrema Detection. This stage of the filtering attempts to identify those locations and scales that are identifiable from different views of the same object. This can be efficiently achieved using a "scale space" function. It is based on the Gaussian function. The scale space is defined by the equation 1.

L(x, y,) = G(x, y,) \* I(x, y) (1)

Where \* is the convolution operator, G(x, y,) is a variable-scale Gaussian and I(x, y) is the input image.

If the function value at z is below a threshold value then this point is excluded. This removes extrema with low contrast. To eliminate extrema based on poor localization it is noted that in these cases there is a large principle curvature across the edge but a small curvature in the perpendicular direction in the difference of Gaussian function.

1. Orientation Assignment. This step aims to assign a consistent orientation to the key points based on local image properties. The key point descriptor, can then be represented relative to this orientation, achieving invariance to rotation. The approach taken to find an orientation is:

* Use the key points scale to select the Gaussian smoothed image L
* Compute gradient magnitude, m = ( + 1, ( 1, ))2 + ( , + 1 (, 1))2
* Compute orientation, = tan( , + 1. 1 )/( + 1,1, ))
* Form an orientation histogram from gradient orientations of sample points and then locate the highest peak in the histogram. Use this peak and any other local peak within 80% of the height of this peak to create a key point with that orientation some points will be assigned multiple orientations Fit a parabola to the 3 histogram values closest to each peak to interpolate the peaks position.

1. Key point Descriptor. The local gradient data, used above, is also used to create key point descriptors. The gradient information is rotated to line up with the orientation of the key point and then weighted by a Gaussian with variance of 1.5 \* key point scale. This data is then used to create a set of histograms over a window centered on the key point.

Key point descriptors typically uses a set of 16 histograms, aligned in a 4×4 grid, each with 8 orientation bins, one for each of the main compass directions and one for each of the mid-points of these directions. This results in a feature vector containing 128 elements.

In the database we have already provided one image each for a sign to make the comparisons. After the input image is provided to the application through defined functions the application will first calculate the key points of the input image after the key points of the input image is calculated then the comparison will start. The application picks up all the images specified in database one by one find the key points of each image one by one and finds the number of matched key points , the comparisons with highest matched key points in an image will take the lead and will be produces as an output. The following process is explained with an example along with figures.

Supposedly if we provide the input image as the Sign/Gesture for character C

3. Key points calculated for input image.

4. No key points matching between database and input image.

5. Only two key points matching between input and database image.

6. Only one key point matching between the images.

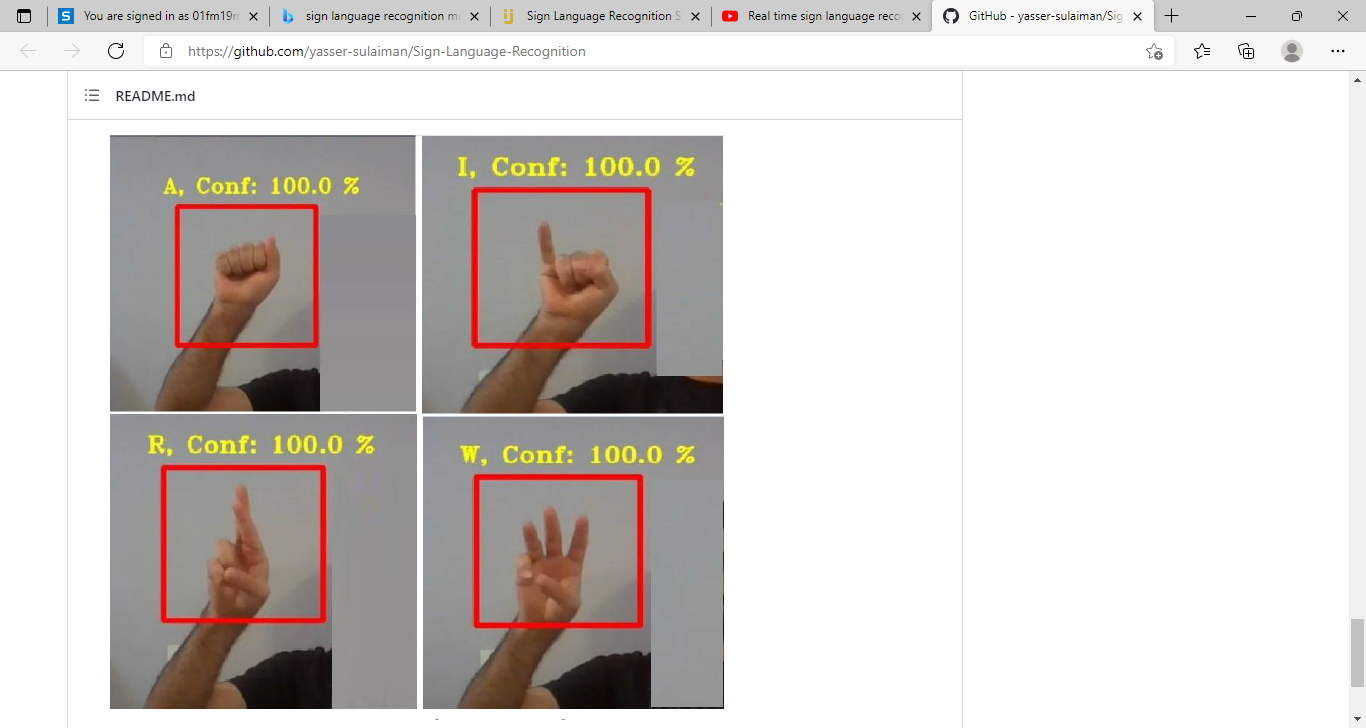
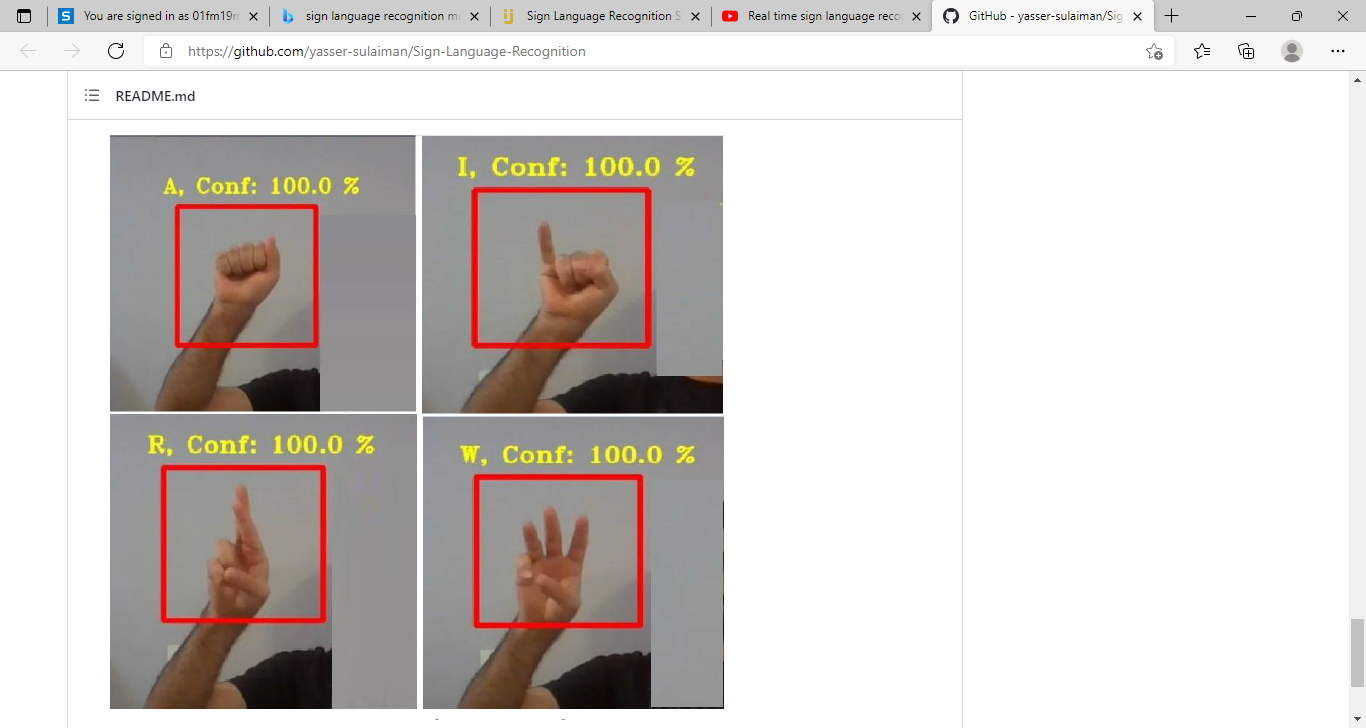
7: The highest key points matching image i.e. 8 key points is found correct and displayed

Gesture Recognition

Finally when the whole process is complete the application will then convert the gesture into its recognized character or alphabet which might be helpful to be understood in layman’s language. The following process includes passing out the single dimensional array of 26 character corresponding to alphabets has been passed where the image number stored in database is provided in the array.

8. The output in the interface for character

Screen shots of project

**5.2 Modules and code**

**Realtimerecognition.py**

import cv2

import numpy as np

import tensorflow as tf

from tensorflow.keras.preprocessing.image import ImageDataGenerator

import os

import pyttsx3

# load saved model from PC

model = tf.keras.models.load\_model(r"new\_model\_20ep3.h5")

model.summary()

data\_dir = 'dataset'

#getting the labels form data directory

labels = sorted(os.listdir(r"data"))

labels[-1] = 'Nothing'

print(labels)

#initiating the video source, 0 for internal camera

cap = cv2.VideoCapture(0)

while(True):

\_ , frame = cap.read()

cv2.rectangle(frame, (100, 100), (300, 300), (0, 0, 255), 5)

#region of intrest

roi = frame[100:300, 100:300]

img = cv2.resize(roi, (50, 50))

cv2.imshow('roi', roi)

img = img/255

#make predication about the current frame

prediction = model.predict(img.reshape(1,50,50,3))

char\_index = np.argmax(prediction)

#print(char\_index,prediction[0,char\_index]\*100)

confidence = round(prediction[0,char\_index]\*100, 1)

predicted\_char = labels[char\_index]

# Initialize the engine

engine = pyttsx3.init()

engine.say(predicted\_char)

engine.runAndWait()

font = cv2.FONT\_HERSHEY\_TRIPLEX

fontScale = 1

color = (0,255,255)

thickness = 2

#writing the predicted char and its confidence percentage to the frame

msg = predicted\_char +', Conf: ' +str(confidence)+' %'

cv2.putText(frame, msg, (80, 80), font, fontScale, color, thickness)

cv2.imshow('frame',frame)

#close the camera when press 'q'

if cv2.waitKey(10) & 0xFF == ord('q'):

break

#release the camera and close all windows

cap.release()

cv2.destroyAllWindows()

**6. Testing**

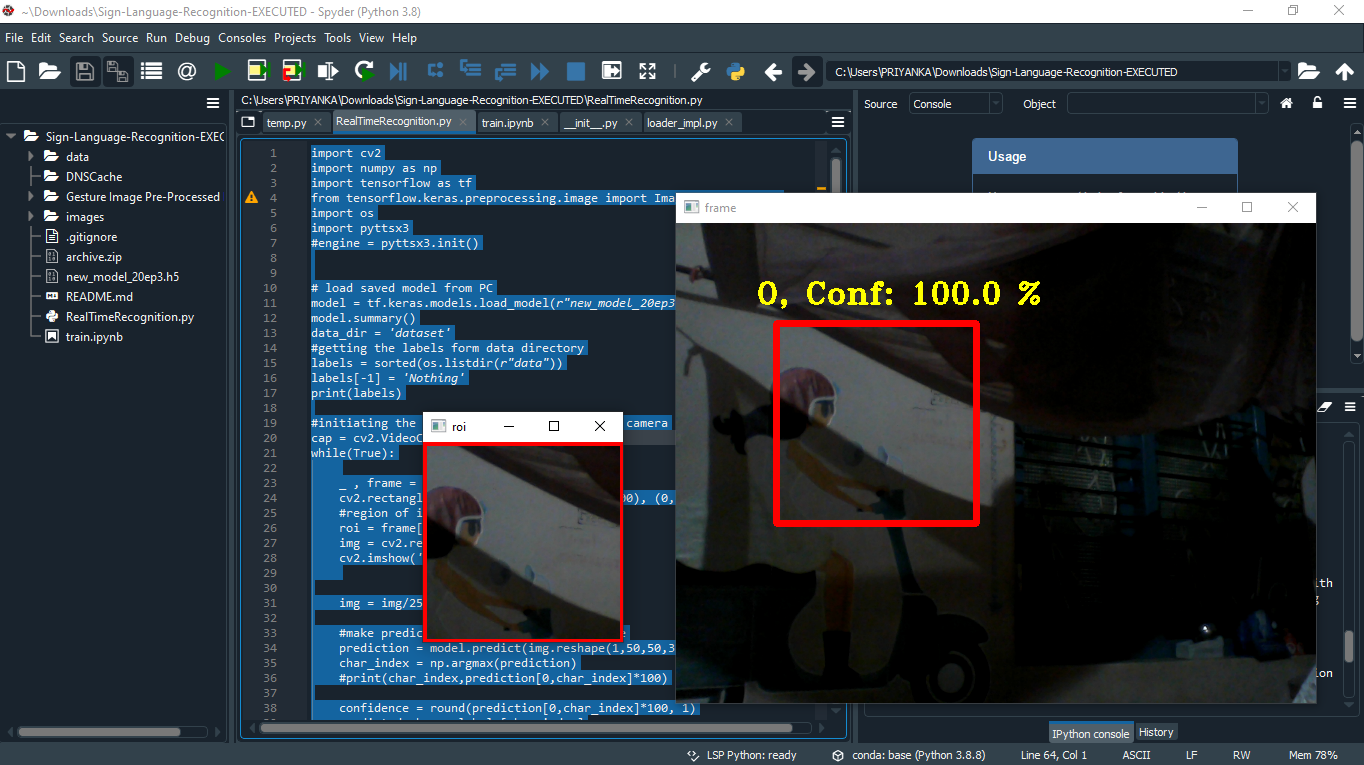
A **TEST CASE** is a set of actions executed to verify a particular feature or functionality of your software application. A Test Case contains test steps, test data, precondition, and post condition developed for specific test scenario to verify any requirement. The test case includes specific variables or conditions, using which a testing engineer can compare expected and actual results to determine whether a software product is functioning as per the requirements of the user.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Sr.no** | **Test Plan** | **Input test image** | **Output test image** | **Actual output** | **Output obtained** |
| 01 | Threshold |  |  | 5 | 5 |
| 02 | Edge Detection |  |  | 5 | 5 |

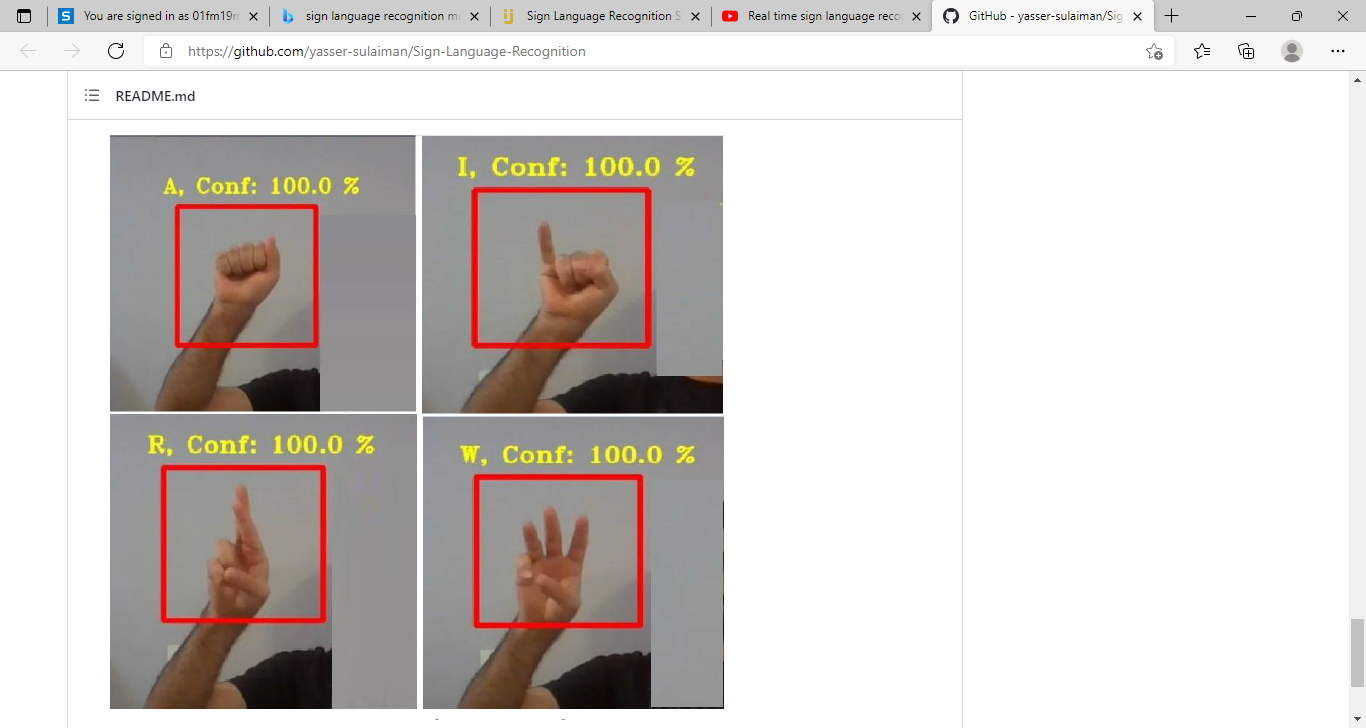
**7. Result and discussions**

When we run the program, firstly we get the two GUI interface and camera gets started. Here we provide our hand gesture to the system camera in a particular red highlighted square box as a symbol to get recognized by the deaf and dumb people. Even it gives voice output of a particular alphabet.

We started out by building our own Convolutional Neural Network and testing it on a static gesture database. The results were astonishing and we got an accuracy of about 100%. The following are the screen shots of our project.

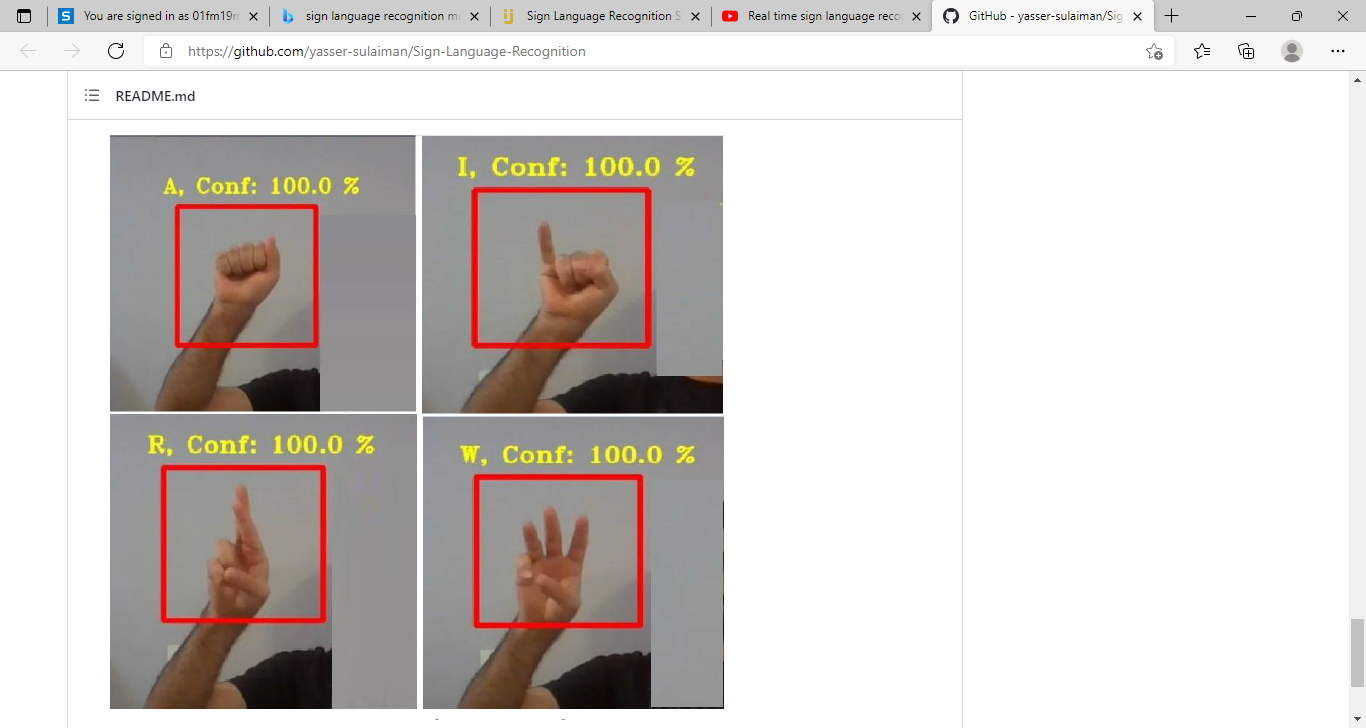


**Fig shows GUI**



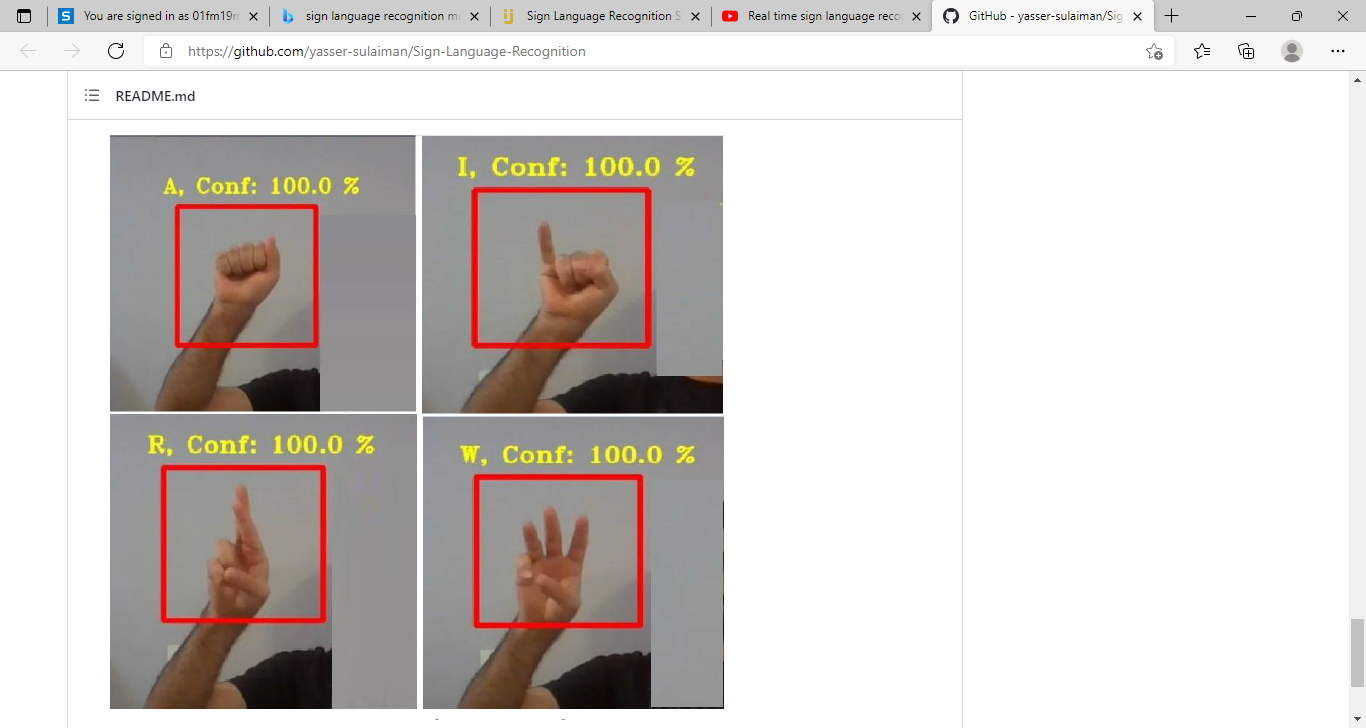
**Fig shows Sign A**

As we provided the input as symbol A it recognizes the symbol and gives the output as A with accuracy of 100%.



**Fig shows Sign I**

As we provided the input as symbol I it recognizes the symbol and gives the output as I with accuracy of 100%.



**Fig shows Sign R**

As we provided the input as symbol R it recognizes the symbol and gives the output as R with accuracy of 100%.

**7. Conclusion and future scope**

**Conclusion**

In this project, we built a system which will correctly recognize ASL Alpha bets and Numbers, which mainly depend on hand and fingers. The model used in this project uses CNN for detection of 26 English ASL alphabets using different image enhancement techniques. This model also uses Convolution layer to highlight the dominant features and max pooling layers to reduce computing cost. The current version of model has one convolution, one max pooling and three fully connected layers. These layers may change while increasing the accuracy of model. The current version of model has accuracy of 100%. Furthermore, added features provide the means for an effortless two way communication between the corresponding entities.

**Future Scope**

This project holds immense potential in terms of real world applications and can be used as a platform for development of solutions to a number of problems. The future scope of this project includes

● Increasing accuracy of current system: This is done in two ways. Firstly, by increasing the size of training dataset to include more variations of characters. Accuracy can also be improved by training the data to a deeper level. However, this will require a very high configuration machine.

● Script recognition engine: The base framework and training approach used in this project can be used to create an application that learns to recognize handwritten scripts given a sizeable dataset to learn from.

**8. References and Bibliography**

**[1]**Pigou L., Dieleman S., Kindermans PJ., Schrauwen B. (2015) Sign Language Recognition Using Convolutional Neural Networks. In: Agapito L., Bronstein M., Rother C. (eds) Computer Vision - ECCV 2014 Workshops. ECCV 2014. Lecture Notes in Computer Science, vol 8925. Springer, Cham.

**[2]** PADMAVATHI. S, SAIPREETHY.M.S, V. Indian sign language character recognition using neural networks. IJCA Special Issue on Recent Trends in Pattern Recognition and Image Analysis, RTPRIA (2013).

**[3]**SAKSHI GOYAL, ISHITA SHARMA, S. S. Sign language recognition system for deaf and dumb people. International Journal of Engineering Research Technology 2, 4 (April 2013)

**[4]** JOYEETA SINGH, K. D. Indian sign language recognition using eigen value weighted euclidean distance based classification technique. International Journal of Advanced Computer Science and Applications 4, 2 (2013). NEHA V. TAVARI, P. A. V. D. Indian sign language recognition based on histograms of oriented gradient. International Journal of Computer Science and Information Technologies 5, 3 (2014), 3657–3660.

**[5]** [arkottke/winguiauto: Windows GUI automation utilities. (github.com)](https://github.com/arkottke/winguiauto)

**[6]**[Sign-Language-Recognition-Using-Hand-Gestures-Keras-PyQT5-OpenCV/ASL Recognition Using Hand Gestures.pdf at master · shadabsk/Sign-Language-Recognition-Using-Hand-Gestures-Keras-PyQT5-OpenCV (github.com)](https://github.com/shadabsk/Sign-Language-Recognition-Using-Hand-Gestures-Keras-PyQT5-OpenCV/blob/master/Documentation/ASL%20Recognition%20Using%20Hand%20Gestures.pdf)

**[7]**[Real time sign language recognition (python, tensorflow, keras and openCV) - YouTube](https://www.youtube.com/watch?v=6VBk9Me0aqM).