Table of Contents

ABSTRACT

Speech impairment is a disability which affects an individual's ability to communicate using speech and hearing. People who are affected by this use other media of communication such as sign language. In the recent past, research in the field of automatic sign language recognition using machine learning methods have demonstrated remarkable success and made momentous progression. A system was developed that will serve as a learning tool for starters in sign language that involves hand detection. Sign language is intentionally designed to allow deaf and dumb communities to convey messages and to connect with society. Unfortunately, learning and practicing sign language is not common among society; hence, this study developed a sign language recognition prototype using the Machine Learning. The images were fed into the model called the Convolutional Neural Network (CNN) for classification of images. Keras was used for training of images. Provided with a proper lighting condition and a uniform background. Many existing studies have proposed methods for incomplete sign language recognition, whereas this study aimed for full American Sign Language (ASL) recognition, which consists of 26 letters. Most of the ASL letters are static (no movement), but certain ASL letters are dynamic (they require certain movements). The approach is used for fast computation and is done in real time. As a result, the sign language recognition system has great potential for reducing the gap between deaf and dumb communities and others. The proposed prototype could also serve as an interpreter for the deaf and dumb in everyday life in service sectors, such as at the bank or post office. Based on the inferences from these approaches, this article discussed how machine learning methods could benefit the field of automatic sign language recognition and the potential gaps that machine learning approaches need to address for the real-time sign language recognition.

1. INTRODUCTION

1.1. PROJECT OVERVIEW

The innovation made by the task changes a Hard of hearing Quiet individual's hand developments into the proper American Communication through signing (ASL) letter sets for a meeting individual to impart. Individuals who need to speak with hard of hearing quiet individuals who need to speak with others, as well as hard of hearing quiet individuals who need to speak with others, are our venture's essential clients. The objective of this drive is to further develop correspondence in crisis circumstances. The CNN (Convolutional Brain Organization) model from the Tensor-Stream bundle is utilized to develop the venture on the Python stage.

1.2. PURPOSE

Everyone can't stand to have a human interpreter of communication through signing, they may not be accessible constantly and they are very costly. Individuals who take part in discussion with hard of hearing quiet individual will think that it is hard and dreary. Hard of hearing quiet individual might lose a ton of chances since they can't talk or offer their viewpoints verbally in circumstances like a meeting. This undertaking means to defeat the said difficulties.

2. LITERATURE SURVEY

2.1. EXISTING PROBLEMS

- This dataset was benchmarked on six different classification models of machine learning by changing the parameters. Classification models are evaluated based on the HOG features extracted from the skin filtered image. An overall accuracy of 91.72% was achieved comprising of all machine learning models.
- The system is trained with double handed sign language by using a minimum eigenvalue algorithm. Here Logitech web camera is used for image acquisition and processing is performed in MATLAB.
- Development of an artificial intelligence voice-based smart device that include the Flex sensors, LCD module, microcontroller, SD card memory, hearing phones, etc. Improves the quality of life without the assistance of some artificial means.
- The human hand comprises of numerous associated parts and joints, making it a complex object for input. The majority of the sign make utilization of both the hands together. Using webcam, capture the image of the hand to be tested. And convert the captured RGB image into HSV image and then into binary image. The edges are detected using canny edge detection.
- Over the years, communication has played a vital role in exchange of information and feelings in one's life. Sign language is the only medium through which specially abled people can connect to rest of the world through different hand gestures. With the advances in machine learning techniques, Hand gesture recognition (HGR) became a very important research topic.
- Processed image, next step is feature extraction & followed by classifier, recognized gestures are displayed as Hindi & English text & played as Hindi & English audio.

2.2 REFERENCE

[1] Leela Surya Teja Mangamuri; Lakshay Jain; Abhishek Sharmay, “Two Hand Indian Sign Language dataset for benchmarking classification models of Machine Learning”, 2019 International Conference on Issues and Challenges in Intelligent Computing Techniques (ICICT).

[2] Kusumika Krori Dutta; Satheesh Kumar Raju K.; Anil Kumar G.S.; Sunny Arokia Swamy B., “Double handed Indian Sign Language to speech and text”, 2015 Third International Conference on Image Information Processing (ICIIP).

[3] Dhaya Sindhu Battina, “Innovative study of an AI voice based smart device to assist deaf people”, International Journal of Creative Research Thoughts (IJCRT),Volume 9, Issue 10 October 2021

[4] Shraddha R. Ghorpade, Prof. Surendra K. Waghmare,"Communication system for deaf and dumb people",International Journal of Science and Research (IJSR)2013, ISSN (Online): 2319-7064.

[5] Kusumika Krori Dutta,Sunny Arokia Swamy Bellary,"Machine Learning Techniques for Indian Sign Language Recognition",2017 International Conference on Current Trends in Computer, Electrical, Electronics and Communication (CTCEEC).

[6] Umang Patel, Aarti G. Ambekar,"Moment Based Sign Language Recognition for Indian Languages",2017 International Conference on Computing, Communication, Control and Automation (ICCUBEA).

2.3. PROBLEM STATEMENT DEFINITION

The investigation of human-PC communication has shown a lot of revenue close by motion acknowledgment. In numerous areas of human-PC communication, including augmented reality, gaming, vehicle framework control, and automated control, fast and exact hand motion acknowledgment is essential. As additional sensors are added, there are various ways of arranging hand movements. Since signal ID is an issue of picture characterization and 2D CNNs are viable in picture order, we have decided to utilize a Convolutional Brain Organization for this undertaking. A framework that changes over the gesture based communication into the particular ASL (American Communication through signing) letter set to pass a message on to ordinary individuals is created in this task.

3. IDEATION & PROPOSED SOLUTION

3.1. EMPATHY MAP CANVAS

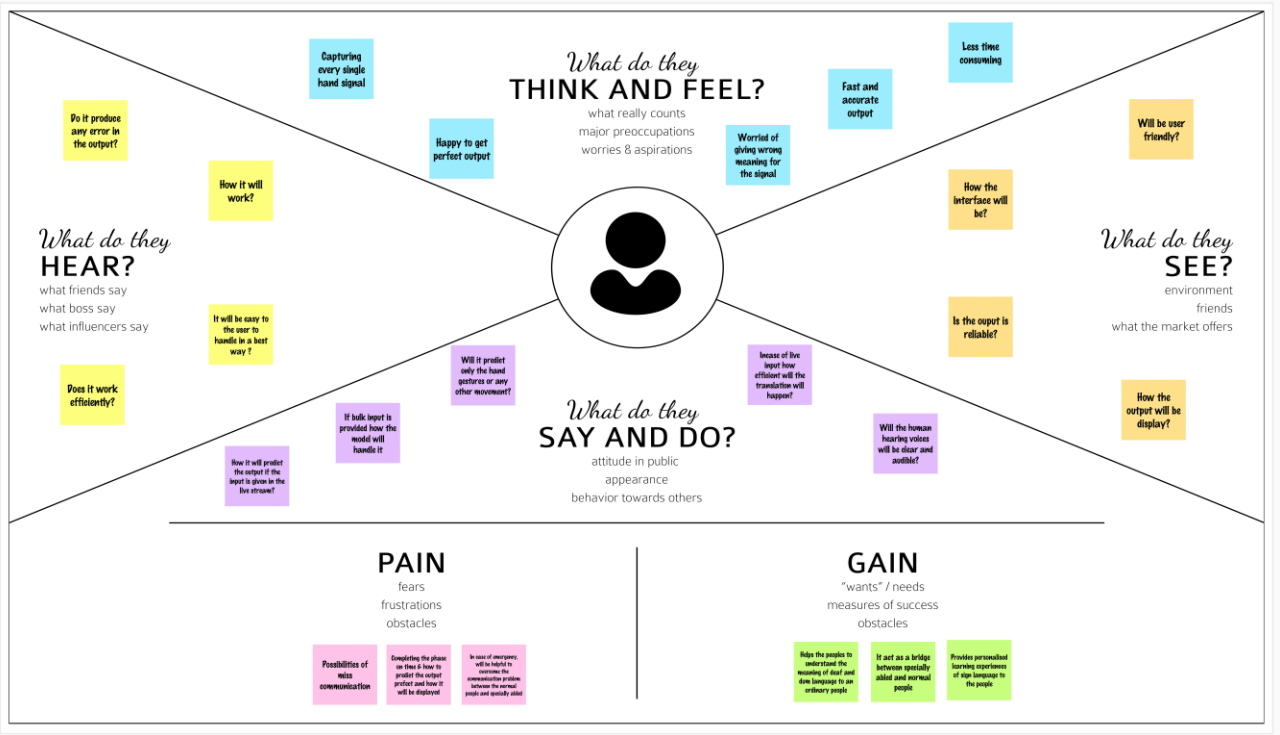


Fig 3.1.1 Empathy Map

3.2. IDEATION & BRAINSTORMING

Step-1: Team Gathering, Collaboration and Select the Problem Statement

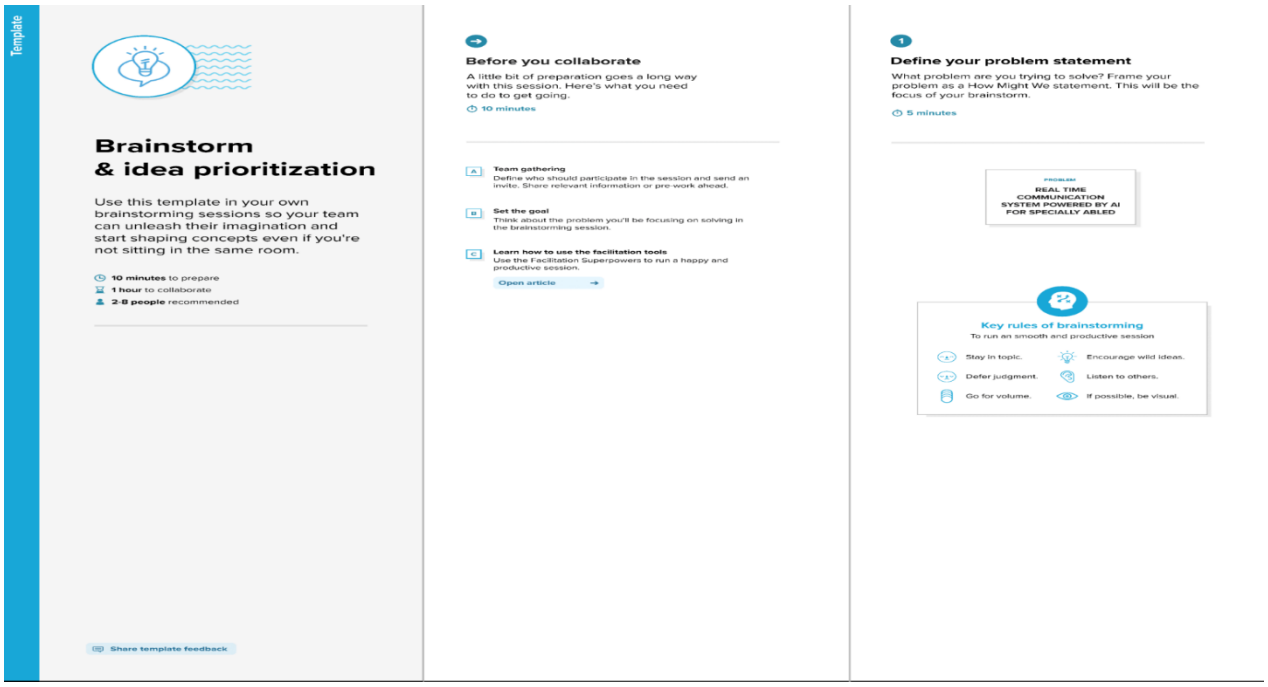


Fig 3.2.1 Problem Statement Collection

Step-2: Brainstorm, Idea Listing and Grouping

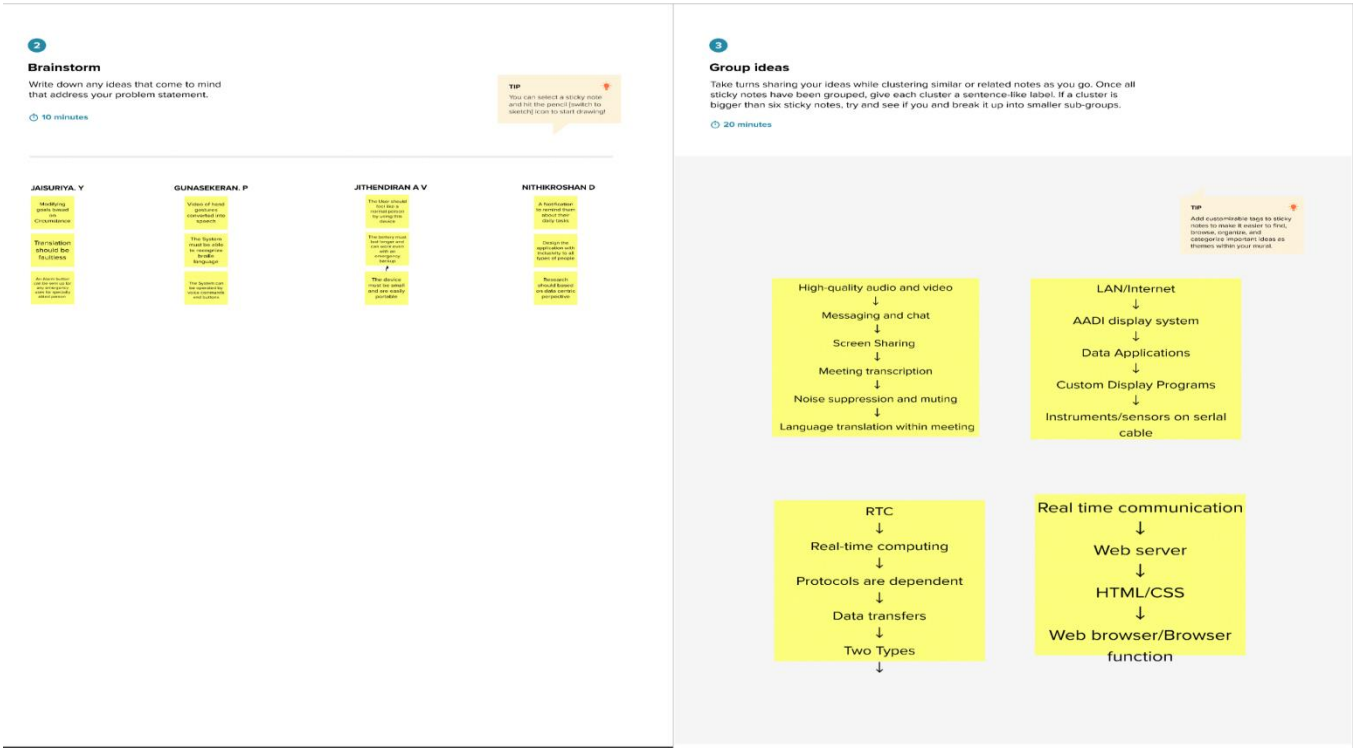


Fig 3.2.2 Listing of Idea

Step-3: Idea Prioritization

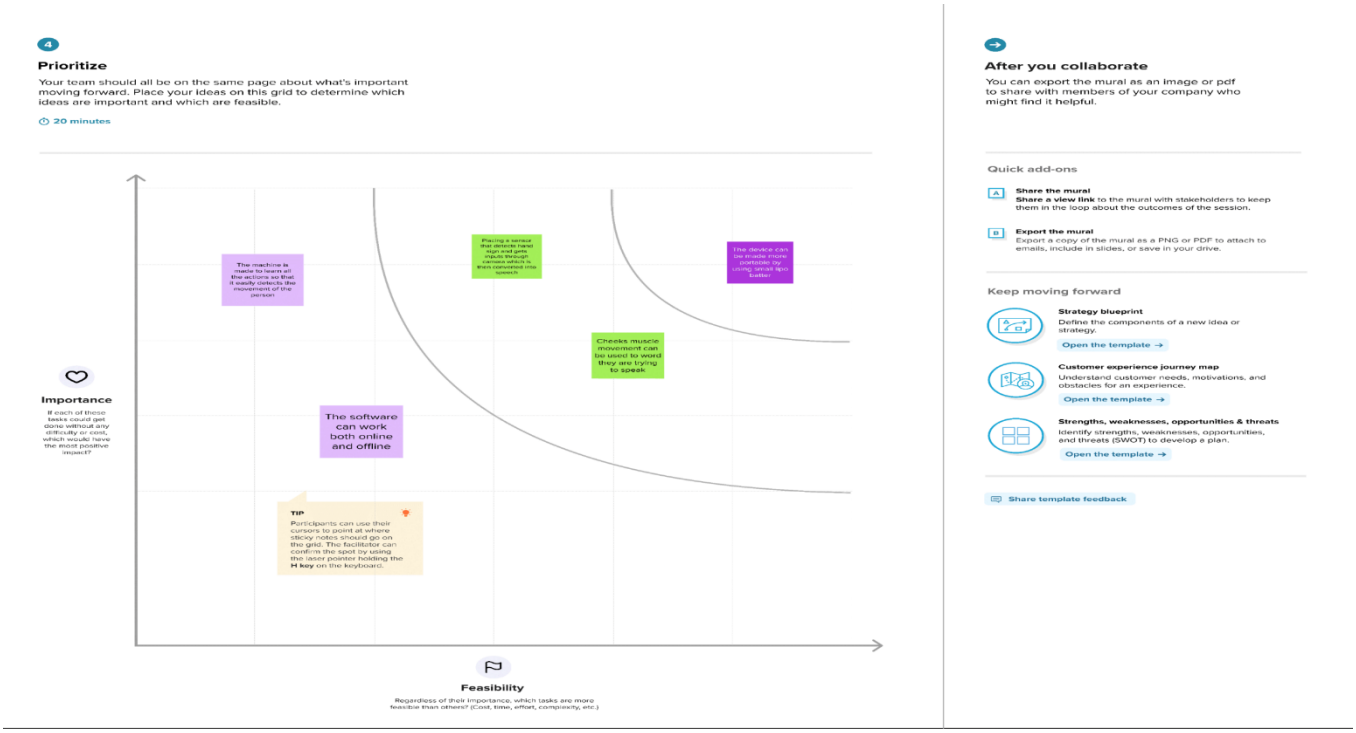


Fig 3.2.3 Idea Prioritization

3.3. PROPOSED SOLUTION

S.No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	Communication between deaf-mute and anormal person has always been a challenging task. It is very difficult for mute people to convey their message to normal people in emergency times as wellas in normal times.
2.	Idea / Solution description	**1.** Voice Conversion System with Hand Gesture Recognition and translation will be very useful to have a proper conversation between a normal person and an impaired person in any language. **2.** To design and implement a system using artificial intelligence, image processing and data mining concepts to take input as hand gestures.
3.	Novelty / Uniqueness	We are using a convolution neural network to create a model that is trained on different hand gestures and an app is built for the use of this mode. **1.** Artificial Intelligence developed the app called GnoSys that uses neuralnetworks and computer vision. **2.** It recognizes the video of sign language speaker and then smart algorithms translate it into speech.
4.	Social Impact / CustomerSatisfaction	**1.** The main purpose of this application is to make deaf-mute people feel independent and more confident. **2.** About two thirds of people with a mobility and dexterity disability are most likely to experience a great deal of difficulty with everyday activities.
5.	Business Model (Revenue Model)	1. AI can generate revenue through direct customers and collaborate with health care sector and generate revenue from their customers. 2. B2B setting uses to employ deaf and mute employees can use to convey messages according to the company.
6.	Scalability of the Solution	1. Enhance people with disabilities to step into a world where there are facing difficulties in communication. 2. AI technology helps disabled people to open up new opportunities for accessibility inclusion in society and independent living. 3. It might open the door to more cutting edge and creative approaches to the most difficult problems facing disadvantaged People.

3.4. PROBLEM SOLUTION FIT

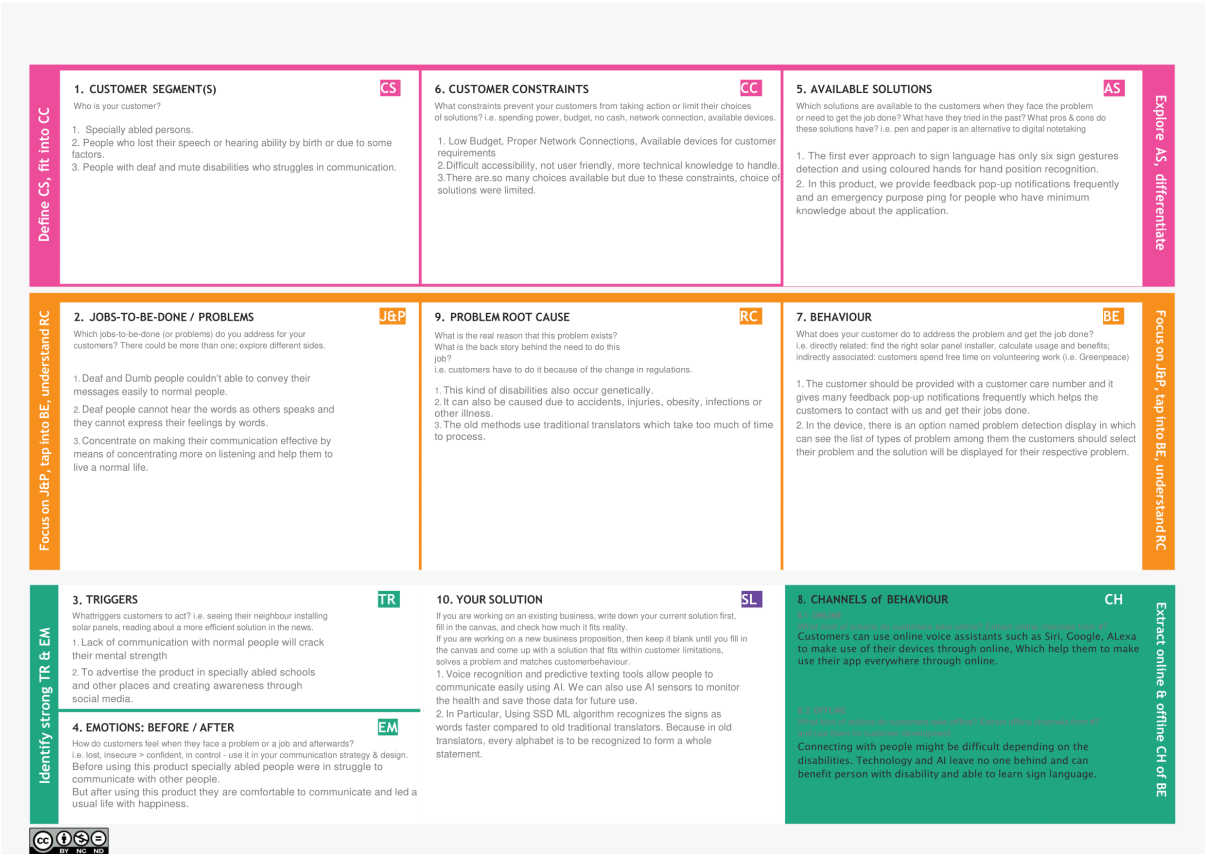


Fig 3.4.1 Problem Solution Fit

4. REQUIREMENT ANALYSIS

4.1 FUNCTIONAL REQUIREMENT

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	User Registration	Registration through Form Registration through Gmail Registration through LinkedIN
FR-2	User Confirmation	Confirmation via Email Confirmation via OTP
FR-3	Uploading image	Upload image through camera Upload image through gallery
FR-4	Text to speech	Select speech icon to convert the respective text for sign language
FR-5	Whiteboard	Use whiteboard to share the message by drawing
FR-6	Emergency templates	Select emergency templates icon to pass the message quickly

4.2. NON-FUNCTIONAL REQUIREMENT

FR No.	Non-Functional Requirement	Description
NFR-1	Usability	Client can undoubtedly upload the image and this application is planned in a manner here, client can without much of a stretch discover some predefined layouts
NFR-2	Security	Client should sign in into an app only then proceed for further process. So unapproved access will be kept away from at max.
NFR-3	Reliability	This application has robust adaptation to non-critical failure and regardless of whether an error happens likewise it recuperates rapidly.
NFR-4	Performance	This application will rapidly transfer and process the images since it predicts the gestures through signing utilizing CNN model and it gives high accuracy.
NFR-5	Availability	The predefined formats will be accessible to all clients and furthermore have whiteboard choice. This application is planned such that it is straightforward and accessible to all clients.
NFR-6	Scalability	Engineers can add new formats and it will build adaptability and this application has premium elements where client approach google maps and google duo.

5. PROJECT DESIGN

5.1. DATA FLOW DIAGRAMS

- Level 0 – DFD

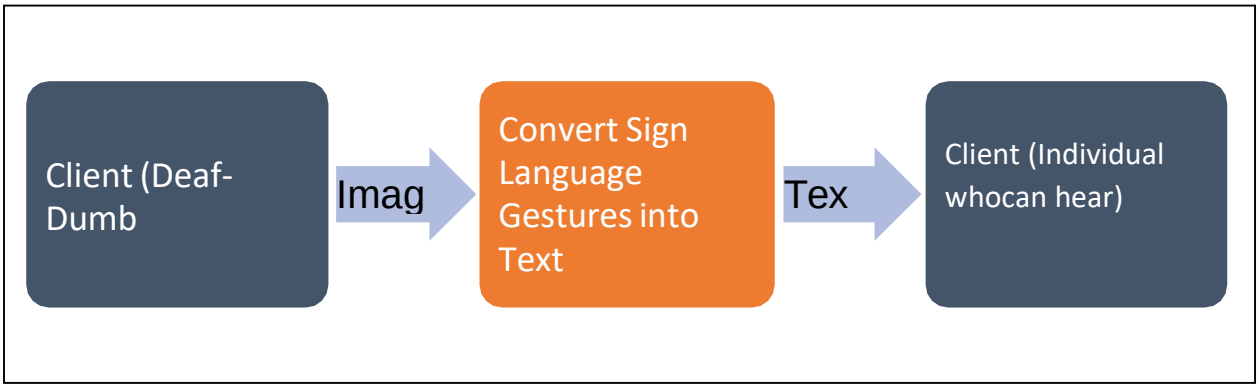


Fig 5.1.1 Data Flow Diagram – Level 0

- Level 1 – DFD

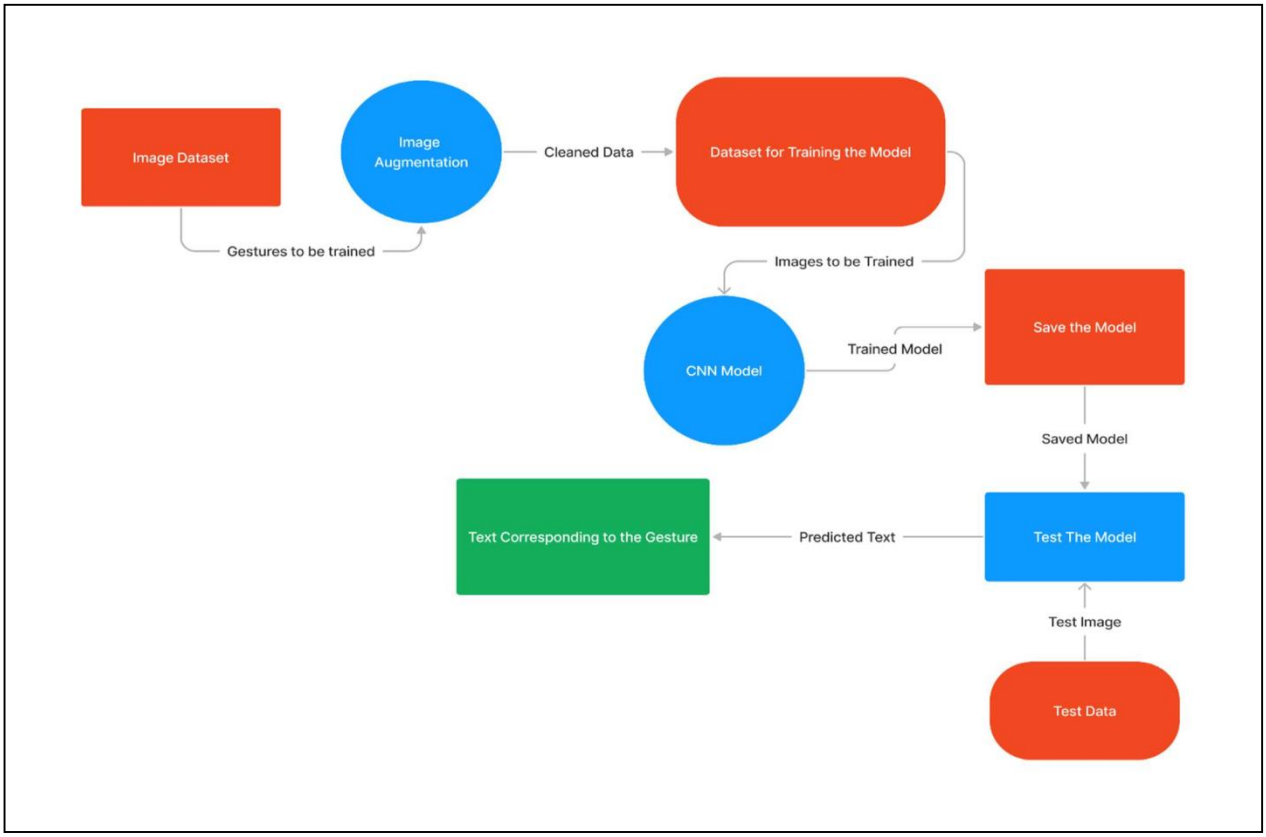


Fig 5.1.2 Data Flow Diagram – Level 1

❖ Simplified Flow Diagram

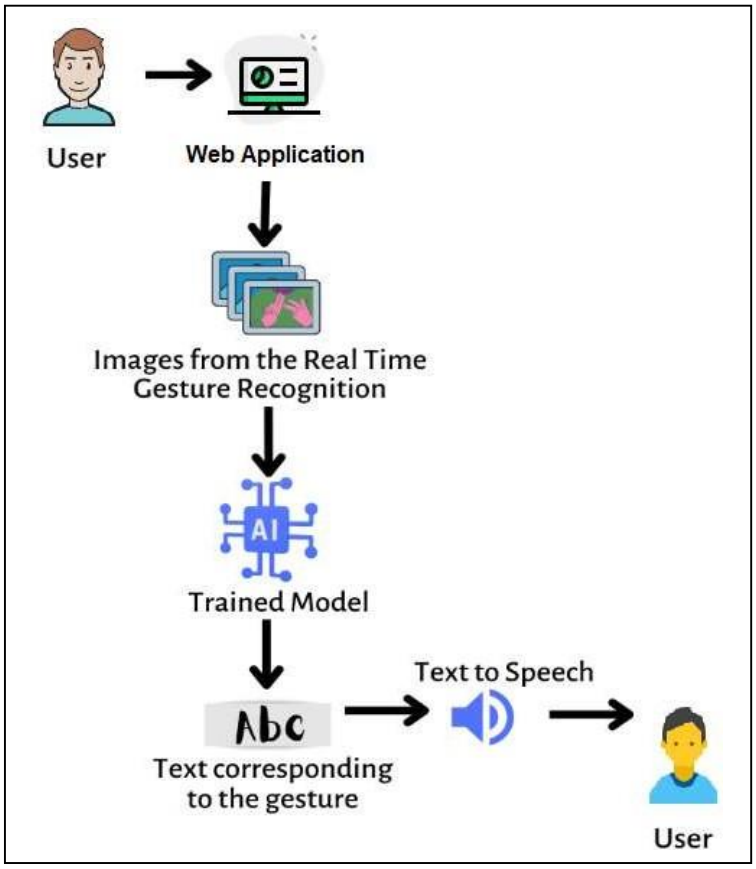


Fig 5.1.3 Simplified flow Diagram

5.2. SOLUTION & TECHNICAL ARCHITECTURE

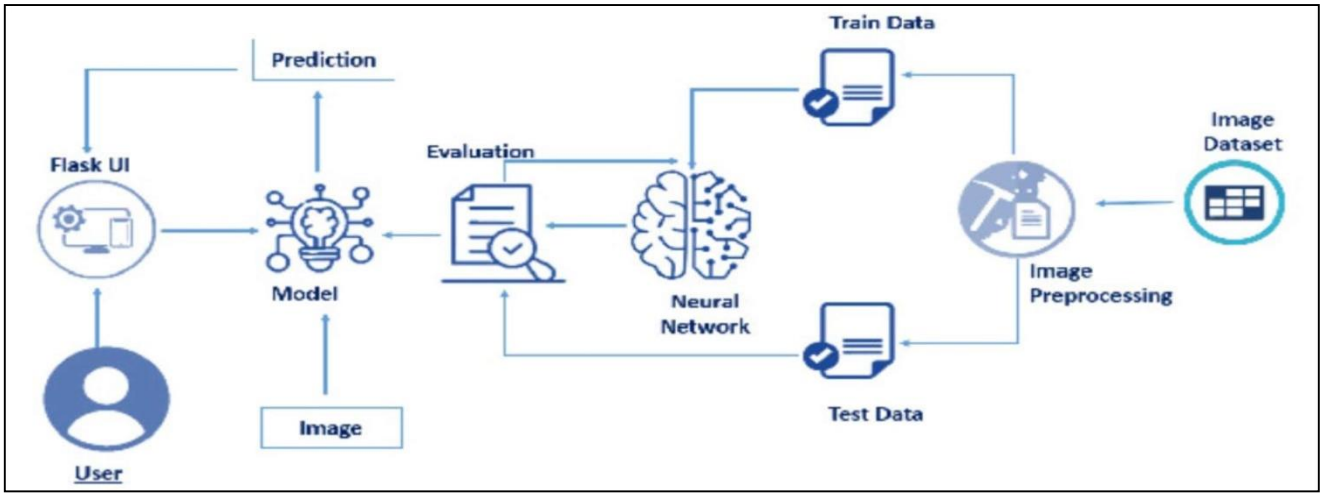


Fig 5.2.1 Technical Architecture

5.3. USER STORIES

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptancecriteria
Customer (People who cannot hear)	Convert sign language into text	USN - 1	As a user, I can open camera in the app and record my signs to be converted into text	I can communicate with normal people effectively
		USN - 2	As a user, I can upload my previous sign gestures to communicate faster	I can have a list of frequently used signs to make for fast reference
	Dashboard	USN – 3	Buttons to record the signs, to convert in real-time and other buttons should be available in the right places	All features must be easily accessible
		USN – 4	Emergency calls must be available so that I can press a button in times of emergency to get the attention of others	I can feel safe because of the Emergency Button which can get me help.
Customer (People who can hear and talk)	Convert sign language into text	USN - 5	As a user, I can open back camera in the app and record the specially abled people's signs to be converted into text	I can understand the mode of communication of specially-abled people effectively
	Show the message to convey in the form of text	USN - 6	As a user, I can open a Text-pad that is available in the app, so that the deaf people can see the message I need to convey	I can convey my message to them effectively
Administrator	Integrate application with trained model	USN - 7	As an admin, I should be able to integrate the AI model into the application and maintain the application	I can give best experience to app users

6. PROJECT PLANNING & SCHEDULING

6.1 SPRINT PLANNING & ESTIMATION

Sprint	Functional Requirement(Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Data Collection	USN-1	Collect Dataset .	9	High	Jaisuriya. Y Jithendiran. A.V
Sprint-1		USN-2	Image preprocessing	8	Medium	Gunasekeran. P Nithikroshan.D
Sprint-2	Model Building	USN-3	Import the required libraries, add the necessary layers and compile the model	10	High	Jaisuriya. Y Jithendiran. A.V
Sprint-2		USN-4	Training the image classification model using CNN	7	Medium	Jaisuriya. Y Jithendiran. A.V
Sprint-3	Training and Testing	USN-5	Training the model and testing the model's performance	9	High	Gunasekeran. P Nithikroshan.D
Sprint-4	Implementation ofthe application	USN-6	Converting the input sign languageimages into English alphabets	8	Medium	Jaisuriya. Y Jithendiran. A.V

6.2 SPRINT DELIVERY SCHEDULE

Sprint	Total Story Points	Duration	Sprint Start Date	Sprint End Date (Planned)	Story Points Completed (as on Planned End Date)	Sprint Release Date (Actual)
Sprint-1	10	6 Days	24 Oct 2022	29 Oct 2022	8	29 Oct 2022
Sprint-2	10	6 Days	31 Oct 2022	04 Nov 2022	5	04 Nov 2022
Sprint-3	10	6 Days	07 Nov 2022	11 Nov 2022	7	11 Nov 2022
Sprint-4	10	6 Days	14 Nov 2022	18 Nov 2022	5	18 Nov 2022

- Burn down Chart

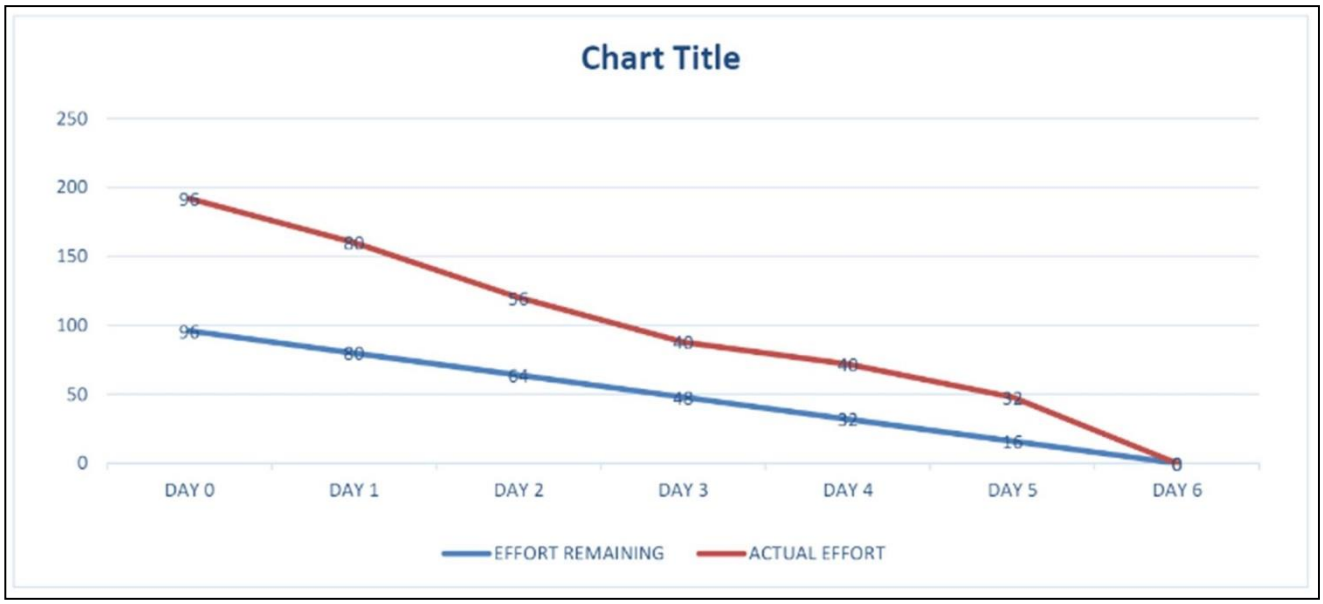


Fig 6.2.1 Burn down chart

- Velocity Chart

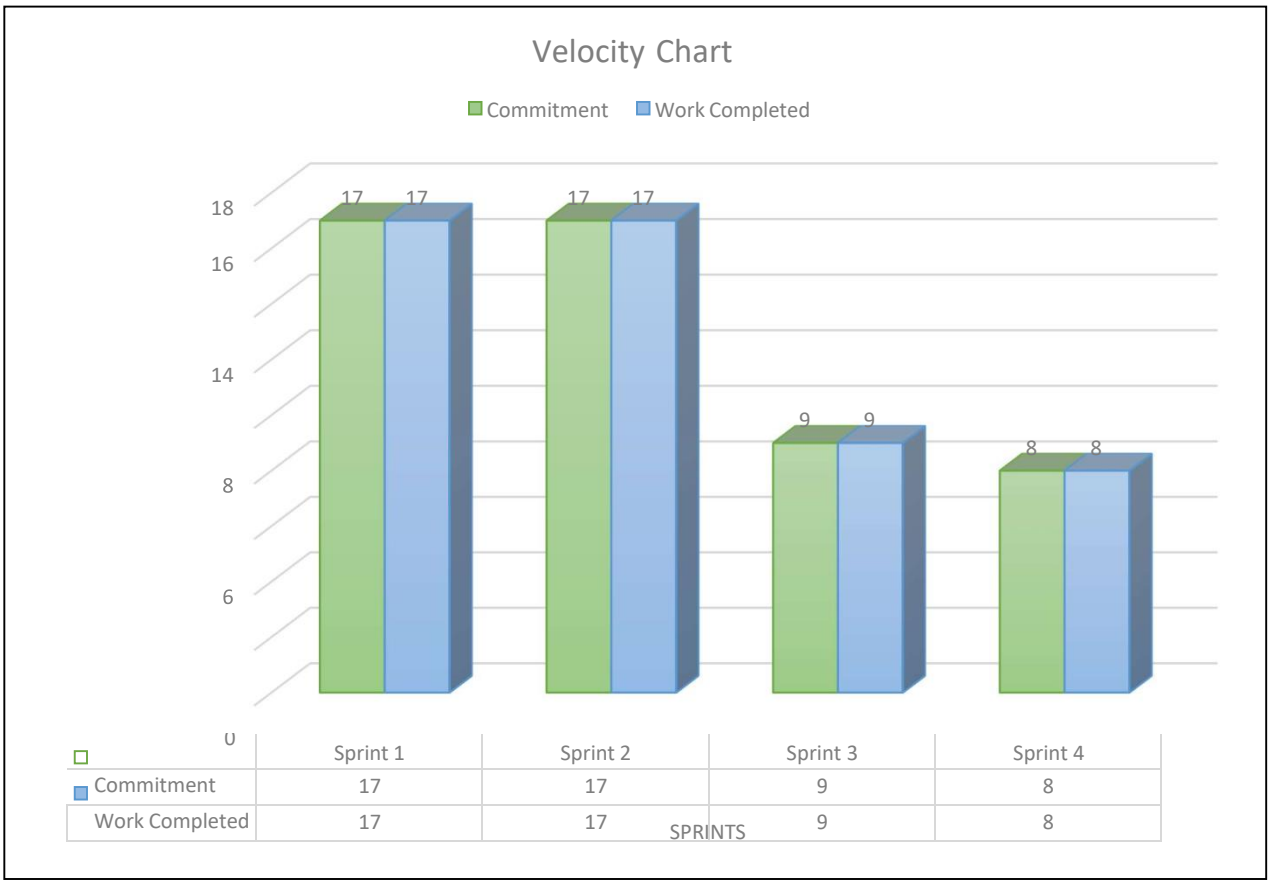


Fig 6.2.2 Velocity Chart

6.3. REPORTS FROM JIRA

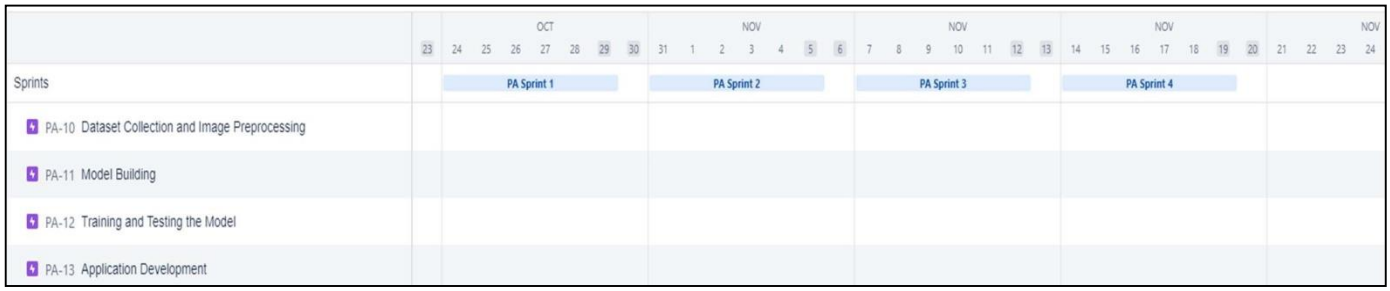


Fig 6.3.1 Jira Report

7. CODING & SOLUTIONING

7.1. IMAGE PREPROCESSING

- Image pre-processing includes zooming, shearing, flipping to increase the robustness of the model after it is built. Keras package is used for pre-processing images.
- Importing Image Data Generator Library to create an instance for which include shearing, rescale, zooming, etc., to make the model robust with different types of images.

```
In [1]: from keras.preprocessing.image import ImageDataGenerator
        train_datagen = ImageDataGenerator(rescale = 1./255,shear_range=0.2,zoom_range=0.2,horizontal_flip=True)
        test_datagen = ImageDataGenerator(rescale=1./255)
```

```
ureWarning: Passing (type, 1) or '1type' as a synonym of type is deprecated; in a future version of numpy, it will be underst
ood as (type, (1,)) / '(1,)type'.
  _np_quint8 = np.dtype([("quint8", np.uint8, 1)])
c:\users\adars\appdata\local\programs\python\python37\lib\site-packages\tensorboard\compat\tensorflow_stub\dtypes.py:543: Fut
ureWarning: Passing (type, 1) or '1type' as a synonym of type is deprecated; in a future version of numpy, it will be underst
ood as (type, (1,)) / '(1,)type'.
  _np_qint16 = np.dtype([("qint16", np.int16, 1)])
c:\users\adars\appdata\local\programs\python\python37\lib\site-packages\tensorboard\compat\tensorflow_stub\dtypes.py:544: Fut
ureWarning: Passing (type, 1) or '1type' as a synonym of type is deprecated; in a future version of numpy, it will be underst
ood as (type, (1,)) / '(1,)type'.
  _np_quint16 = np.dtype([("quint16", np.uint16, 1)])
c:\users\adars\appdata\local\programs\python\python37\lib\site-packages\tensorboard\compat\tensorflow_stub\dtypes.py:545: Fut
ureWarning: Passing (type, 1) or '1type' as a synonym of type is deprecated; in a future version of numpy, it will be underst
ood as (type, (1,)) / '(1,)type'.
  _np_qint32 = np.dtype([("qint32", np.int32, 1)])
c:\users\adars\appdata\local\programs\python\python37\lib\site-packages\tensorboard\compat\tensorflow_stub\dtypes.py:550: Fut
ureWarning: Passing (type, 1) or '1type' as a synonym of type is deprecated; in a future version of numpy, it will be underst
ood as (type, (1,)) / '(1,)type'.
  np_resource = np.dtype([("resource", np.ubyte, 1)])
```

Fig 7.1.1 Image Processing

- Applying Image Data Generator Functionality to Train and Test Set

```
In [2]: x_train = train_datagen.flow_from_directory('dataset/training_set',target_size=(64,64),batch_size=300,class_mode='categorical',c

        x_test = test_datagen.flow_from_directory('dataset/test_set',target_size=(64,64),batch_size=300,class_mode='categorical',color_m

        Found 15750 images belonging to 9 classes.
        Found 2250 images belonging to 9 classes.
```

Fig 7.1.2 Train and Test Data Set

7.2. MODEL BUILDING

- Importing The Required Model Building Libraries

```
In [3]: from keras.models import Sequential
        from keras.layers import Dense
        from keras.layers import Convolution2D
        from keras.layers import MaxPooling2D
        from keras.layers import Dropout
        from keras.layers import Flatten
```

Fig 7.2.1 Importing Libraries

- Initializing the Mode

```
In [4]: model = Sequential()

        WARNING:tensorflow:From c:\users\adars\appdata\local\programs\python\python37\lib\site-packages\keras\backend\tensorflow_backen
        d.py:74: The name tf.get_default_graph is deprecated. Please use tf.compat.v1.get_default_graph instead.
```

Fig 7.2.2 Initializing the Mode

- Adding the Convolution Layer

```
In [5]: model.add(Convolution2D(32,(3,3),input_shape=(64,64,1),activation='relu'))

        WARNING:tensorflow:From c:\users\adars\appdata\local\programs\python\python37\lib\site-packages\keras\backend\tensorflow_backen
        d.py:517: The name tf.placeholder is deprecated. Please use tf.compat.v1.placeholder instead.

        WARNING:tensorflow:From c:\users\adars\appdata\local\programs\python\python37\lib\site-packages\keras\backend\tensorflow_backen
        d.py:4138: The name tf.random_uniform is deprecated. Please use tf.random.uniform instead.
```

Fig 7.2.3 Adding the Convolution Layer

- Adding the Pooling Layer

```
In [6]: model.add(MaxPooling2D(pool_size=(2,2)))

        WARNING:tensorflow:From c:\users\adars\appdata\local\programs\python\python37\lib\site-packages\keras\backend\tensorflow_backen
        d.py:3976: The name tf.nn.max_pool is deprecated. Please use tf.nn.max_pool2d instead.
```

Fig 7.2.4 Aadding The Pooling Layer

- Adding the Flatten Layer

```
In [7]: model.add(Flatten())
```

Fig 7.2.5 Aadding Flatten Layer

- Adding the Dense Layers

```
In [8]: model.add(Dense(units=512,activation='relu'))
        model.add(Dense(units=9,activation='softmax'))
```

Fig 7.2.6 Aadding Dense Layer

- Compiling the Model

```
In [9]: model.compile(loss='categorical_crossentropy',optimizer='adam',metrics=['accuracy'])

        WARNING:tensorflow:From c:\users\adars\appdata\local\programs\python\python37\lib\site-packages\keras\optimizers.py:790: The na
        me tf.train.Optimizer is deprecated. Please use tf.compat.v1.train.Optimizer instead.

        WARNING:tensorflow:From c:\users\adars\appdata\local\programs\python\python37\lib\site-packages\keras\backend\tensorflow_backen
        d.py:3295: The name tf.log is deprecated. Please use tf.math.log instead.
```

Fig 7.2.7 Compiling the Model

- Fit and Saving the Model

```
In [10]: model.fit_generator(x_train,steps_per_epoch=24,epochs=10,validation_data=x_test,validation_steps=40)

         model.save('signlanguage.h5')

         WARNING:tensorflow:From c:\users\adars\appdata\local\programs\python\python37\lib\site-packages\tensorflow\python\ops\math_gra
         d.py:1250: add_dispatch_support.<locals>.wrapper (from tensorflow.python.ops.array_ops) is deprecated and will be removed in a
         future version.
         Instructions for updating:
         Use tf.where in 2.0, which has the same broadcast rule as np.where
         WARNING:tensorflow:From c:\users\adars\appdata\local\programs\python\python37\lib\site-packages\keras\backend\tensorflow_backen
         d.py:986: The name tf.assign_add is deprecated. Please use tf.compat.v1.assign_add instead.

         Epoch 1/10
         24/24 [==============================] - 17s 720ms/step - loss: 1.3771 - acc: 0.6265 - val_loss: 0.5072 - val_acc: 0.8809
         Epoch 2/10
         24/24 [==============================] - 15s 635ms/step - loss: 0.3532 - acc: 0.9039 - val_loss: 0.2881 - val_acc: 0.9302
         Epoch 3/10
         24/24 [==============================] - 15s 633ms/step - loss: 0.1826 - acc: 0.9522 - val_loss: 0.1999 - val_acc: 0.9471
         Epoch 4/10
         24/24 [==============================] - 15s 645ms/step - loss: 0.1130 - acc: 0.9707 - val_loss: 0.1902 - val_acc: 0.9676
         Epoch 5/10
         24/24 [==============================] - 15s 630ms/step - loss: 0.0773 - acc: 0.9802 - val_loss: 0.1742 - val_acc: 0.9653
         Epoch 6/10
         24/24 [==============================] - 15s 641ms/step - loss: 0.0534 - acc: 0.9872 - val_loss: 0.1953 - val_acc: 0.9733
         Epoch 7/10
         24/24 [==============================] - 15s 633ms/step - loss: 0.0366 - acc: 0.9917 - val_loss: 0.1920 - val_acc: 0.9698
         Epoch 8/10
         24/24 [==============================] - 15s 637ms/step - loss: 0.0314 - acc: 0.9936 - val_loss: 0.2024 - val_acc: 0.9716
         Epoch 9/10
         24/24 [==============================] - 15s 630ms/step - loss: 0.0271 - acc: 0.9939 - val_loss: 0.2371 - val_acc: 0.9738
         Epoch 10/10
         24/24 [==============================] - 15s 634ms/step - loss: 0.0192 - acc: 0.9965 - val_loss: 0.1944 - val_acc: 0.9747
```

Fig 7.2.8 Fit and Saving the Model

7.3. TESTING THE MODEL

- Importing The Packages and Loading the Saved Model

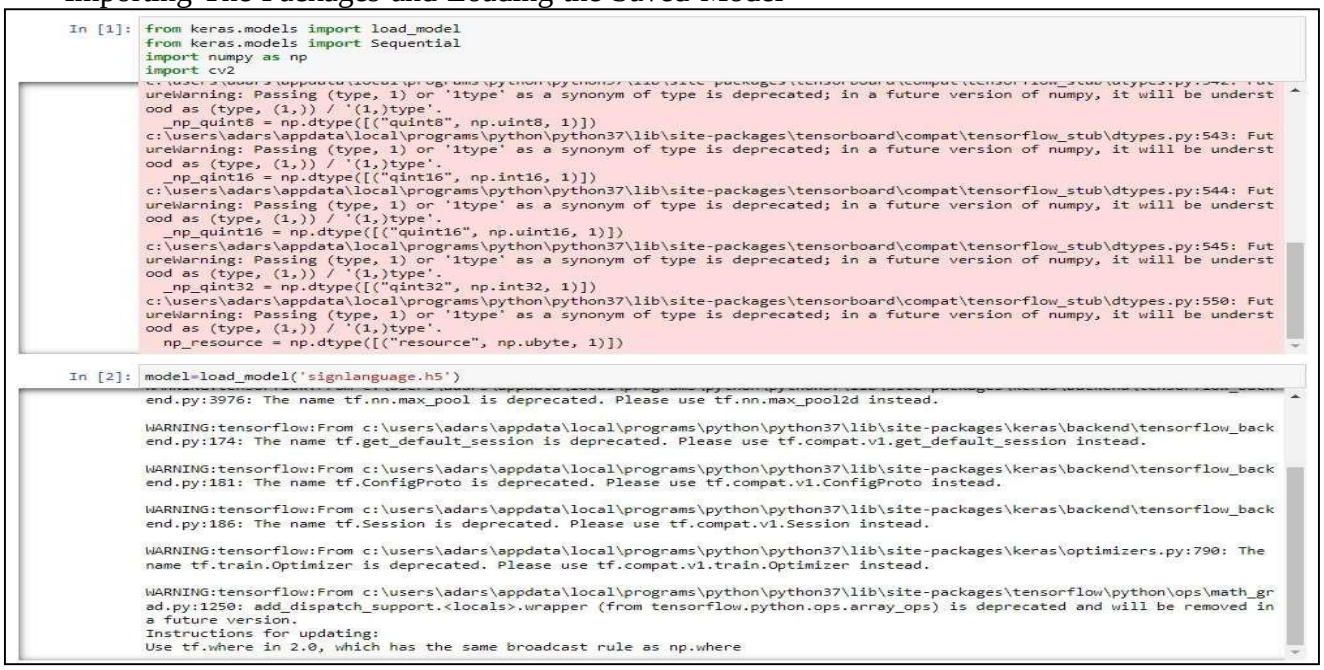


Fig 7.3.1 Load the Model

- Loading the Test Image, Pre-Processing it And Prediction

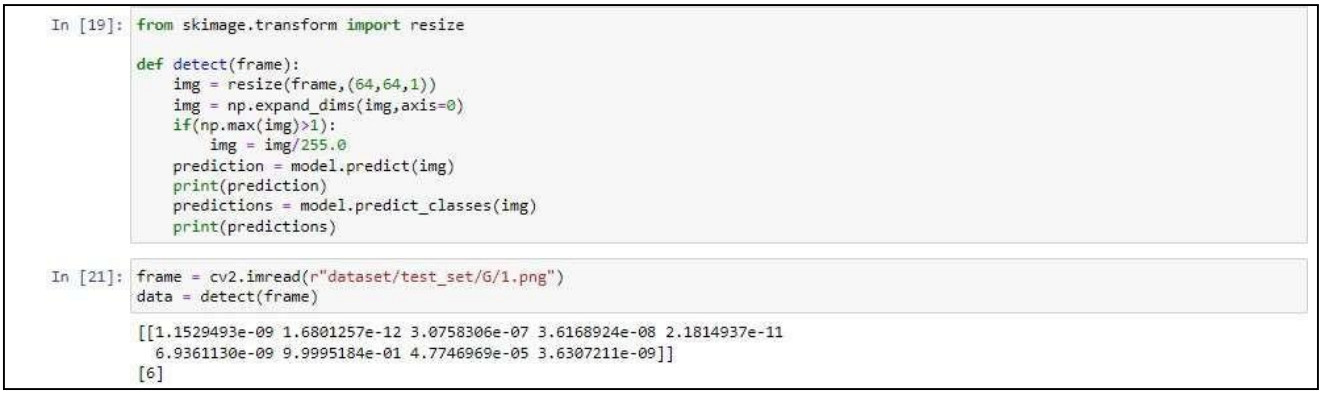


Fig 7.3.2 Test Image, Pre-Processing it And Prediction

- The output [6] in the above image represents the index value in the array[‘A’,’B’,’C’,’D’,’E’,’F’,’G’,’H’,’I’].
- Thus, the predicted alphabet is G.

7.4. FLASK APPLICATION

- Loading the required packages (Fig 7.4.1)
- Initializing graph, loading the model, initializing the flask app and loading the video.

```
import numpy as np
import cv2
import os
from tensorflow.keras.models import load_model
from tensorflow.keras.preprocessing import image
from tensorflow.keras.backend import set_session
from flask import Flask, render_template,Response
import tensorflow as tf
from gtts import gTTS
global graph
global writer
from skimage.transform import resize
```

Fig 7.4.1 Loading the required packages

- Graph element is required to work with TensorFlow. So, graph element is created explicitly.

```
graph = tf.get_default_graph()
model = load_model('signlanguagel.h5')
vals = ['A','B','C','D','E','F','G','H','I']
app = Flask(__name__)
print("[INFO] accessing video stream...")
camera = cv2.VideoCapture(1)
camera.set(cv2.CAP_PROP_FRAME_WIDTH, 1280)
camera.set(cv2.CAP_PROP_FRAME_HEIGHT, 720)
pred=""
```

Fig 7.4.2 Initializing graph

- Configuring the home page

```
@app.route('/')
def index():
    return render_template('index.html')
```

Fig 7.4.3 Configure the home page

- Pre-processing the frames captured from the camera

```
def detect(frame):
    global pred
    global graph
    img = resize(frame,(64,64,1))
    x = image.img_to_array(img)
    x = np.expand_dims(x,axis=0)
    with graph.as_default():
        predictions = model.predict_classes(x)
    print(predictions)
    pred=vals[predictions[0]]
    print(pred)
```

Fig 7.4.4 Pre-processing the frames

- Video Feed call from the HTML PAGE

```
@app.route('/video_feed')
def video_feed():
    return Response(gen(),mimetype='multipart/x-mixed-replace; boundary=frame')

if __name__=='__main__':
    app.run()
```

Fig 7.4.5 Video Feed call

7.5. HTML PAGE

- HTML page to display the processed video on the screen, so that the person can show signs which canbe detected.

```
<!doctype html>
<html lang="en">
<head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0, shrink-to-fit=no">
    <title>VHearuU</title>
    <link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/css/bootstrap.min.css">
    <link rel="stylesheet" href="https://use.fontawesome.com/releases/v5.12.0/css/all.css">
    <link rel="stylesheet" href="static/css/Banner-Heading-Image.css">
    <link rel="stylesheet" href="static/css/Navbar-Centered-Brand.css">
    <link rel="stylesheet" href="static/css/styles.css">
</head>
<body>

<nav class="navbar navbar-light navbar-expand-md py-3" style="background: #429691;">
        <div class="container">
            <div></div><a class="navbar-brand d-flex align-items-center" href="#"><h4 style="color: #ffffff; font-style: oblique; text-align: center;font-family: Arial Black"><strong> Real-Time Communication
                    System Powered By AI&amp;nbsp;For Specially Abled - Team ID: PNT2022TMID09233</strong></h4></a>
            <div></div>
        </div>
    </nav>

                <img src="{{ url_for('video_feed') }}" style="width: 100%;height: 100%;color: rgb(255,255,255);text-align: center;font-size: 20px;"
                    alt="Camera Access Not Provided!">


<section>
        <div class="container">
            <div class="accordion text-white" role="tablist" id="accordion-1">

                <div class="accordion-item" style="font-style: oblique; background: #429691;">
                    <h2 class="accordion-header" role="tab"><button class="accordion-button collapsed"
                            data-bs-toggle="collapse" data-bs-target="#accordion-1 .item-2" aria-expanded="false"
                            aria-controls="accordion-1 .item-2"
                            style="font-style: oblique; background: #cc7931;color: #ffffff;">Project: Real-Time Communication System Powered by AI for Specially Abled</button></h2>
                    <div class="accordion-collapse collapse item-2" role="tabpanel" data-bs-parent="#accordion-1">
                        <div class="accordion-body">
                            <p class="mb-0">Team ID: PNT2022TMID09233<br><br>VHearU Developed By,<br><br>1. <strong>Abubucker Siddique</strong> 310619104001<br>2.
                                <strong>Adarsh Srinivas</strong> 310619104003<br>3. <strong>Ashwath</strong> 310619104012<br>4. <strong>Jayasurya</strong> 310619104044
                            </p>
                        </div>
                    </div>
                </div>
            </div>
        </div>
    </section>
    <div class="modal fade" role="dialog" tabindex="-1" id="modal-1">
        <div class="modal-dialog" role="document">
            <div class="modal-content">
                <div class="modal-header">
                    <h4 class="modal-title">American Sign Language - Alphabets</h4><button type="button"
                        class="btn-close" data-bs-dismiss="modal" aria-label="Close"></button>
                </div>
                <div class="modal-body"><img src="{{ url_for('static', filename='img/ASL_Alphabets.png') }}" width="100%"></div>
                <div class="modal-footer"><button class="btn btn-secondary" type="button"
                        data-bs-dismiss="modal">Close</button></div>
            </div>
        </div>
    </div>
    <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/js/bootstrap.bundle.min.js"></script>
</body>
</html>
```

Fig 7.5.1 HTML Page

8. TESTING

8.1. TEST CASES

Test Case ID	Test Scenario	Steps to Execute	Expected Result	Actual Result
1	Verify if user is able to provide camera access.	1. Enter URL and click go. 2. Give Camera Access.	Camera is On.	Working as expected.
2	Verify if user is able to get the desirable prediction for the gesture.	1. Enter URL and click go. 2. Give Camera Access. 3. Make Gesture in front of camera.	Alphabet is predicted for the gesture.	Working as expected.

8.2. USER ACCEPTANCE TESTING

- **Defect Analysis**

Resolution	Severity 1	Severity 2	Severity 3	Severity 4	Subtotal
By Design	0	12	1	1	14
External	5	0	0	0	5
Fixed	11	3	2	2	18
Skipped	0	0	2	0	2
Won't Fix	4	0	0	0	4
Totals	20	15	5	3	43

- **Test Case Analysis**

Section	Total Cases	Not Tested	Fail	Pass
Client Application	5	1	0	5
Security	2	0	0	2
Exception Reporting	2	0	0	2
Final Report Output	9	0	0	9

- The project developed was tested by an end user and the application converts the gestures to itsrespective alphabet accurately.

9. RESULT

9.1 SAMPLE OUTPUT

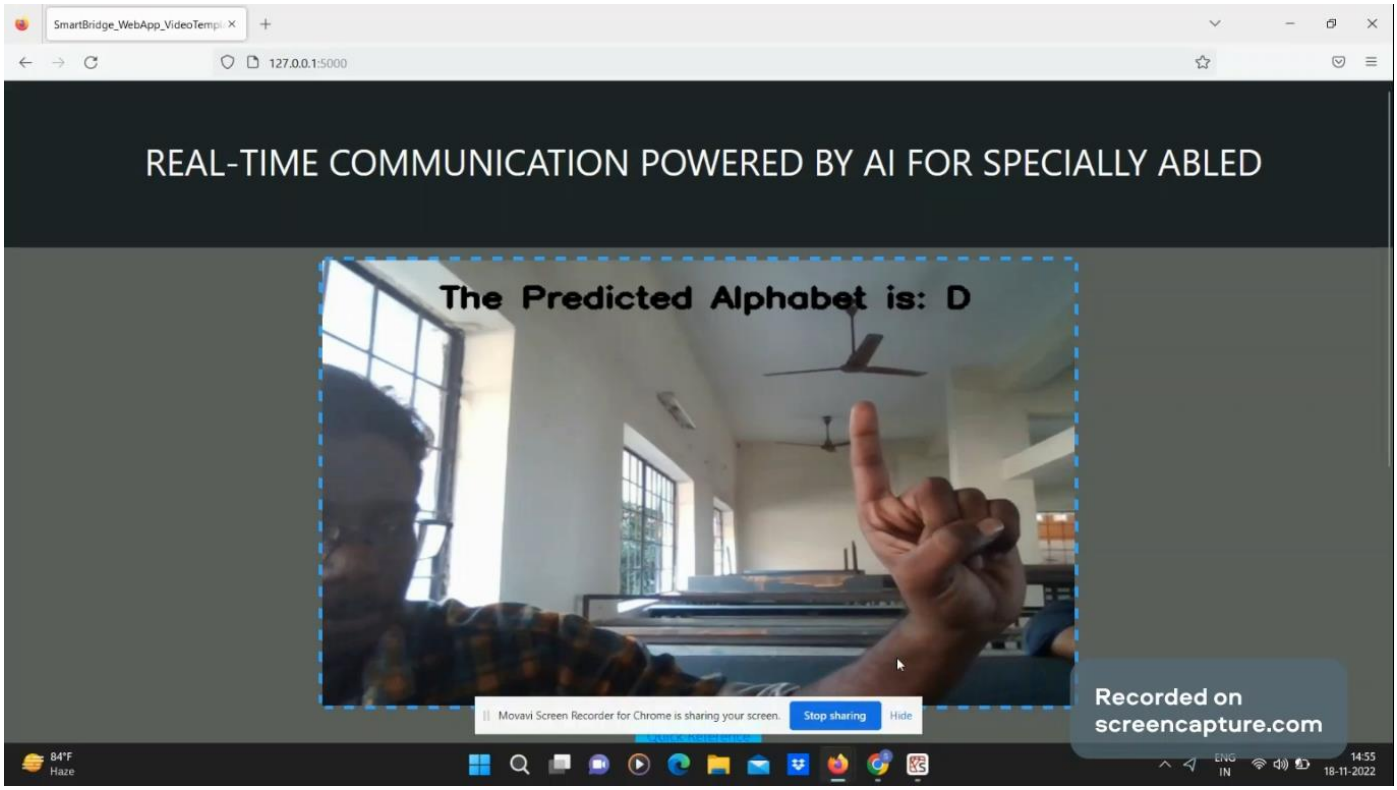


Fig 9.1 Predicated Letter “D”

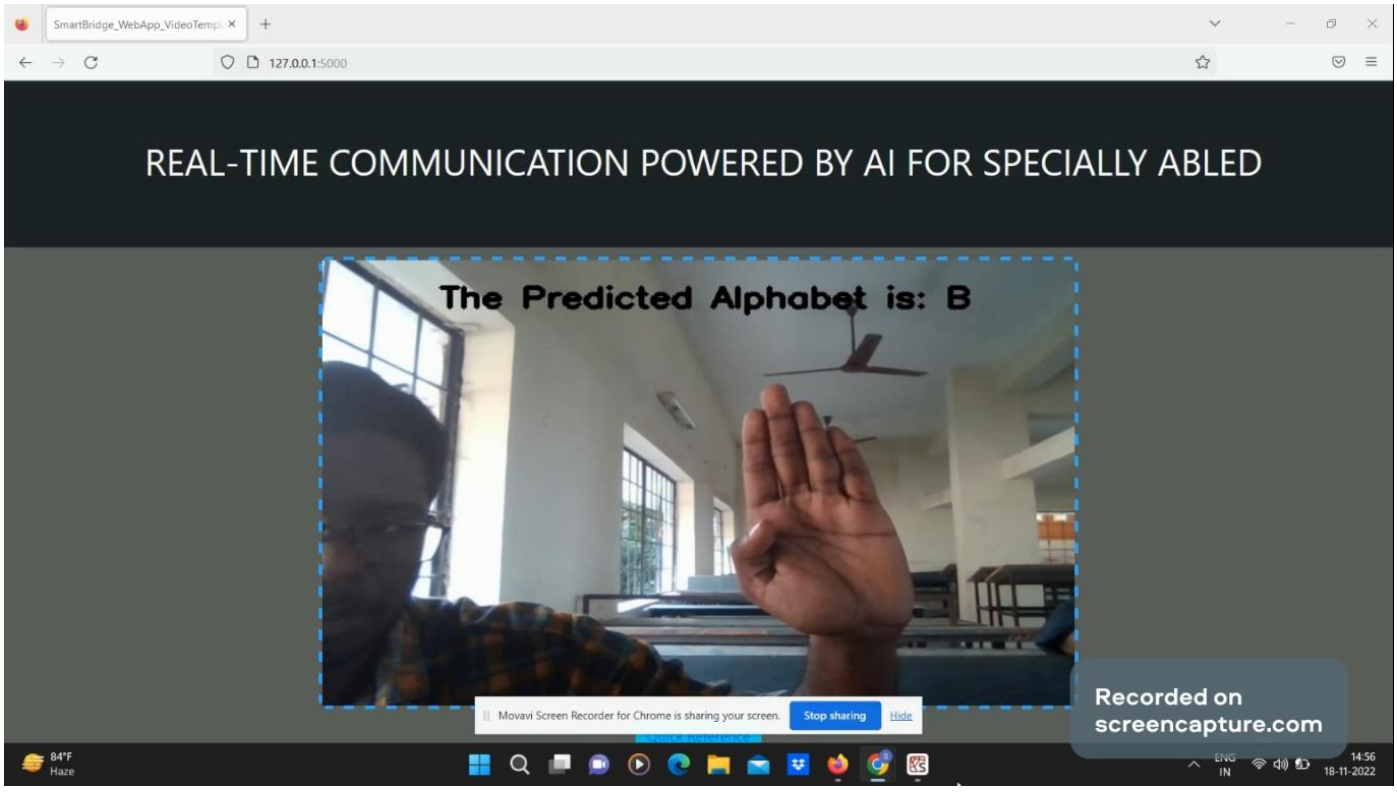


Fig 9.2 Predicated Letter “B”

9.2. PERFORMANCE METRICS

- Model Summary

```
In [40]: model.summary()

Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d (Conv2D)              (None, 62, 62, 32)        320
_________________________________________________________________
max_pooling2d (MaxPooling2D) (None, 31, 31, 32)        0
_________________________________________________________________
flatten (Flatten)            (None, 30752)             0
_________________________________________________________________
dense (Dense)                (None, 512)               15745536
_________________________________________________________________
dense_1 (Dense)              (None, 9)                 4617
=================================================================
Total params: 15,750,473
Trainable params: 15,750,473
Non-trainable params: 0
_________________________________________________________________
```

- Confusion Matrix and Classification Report

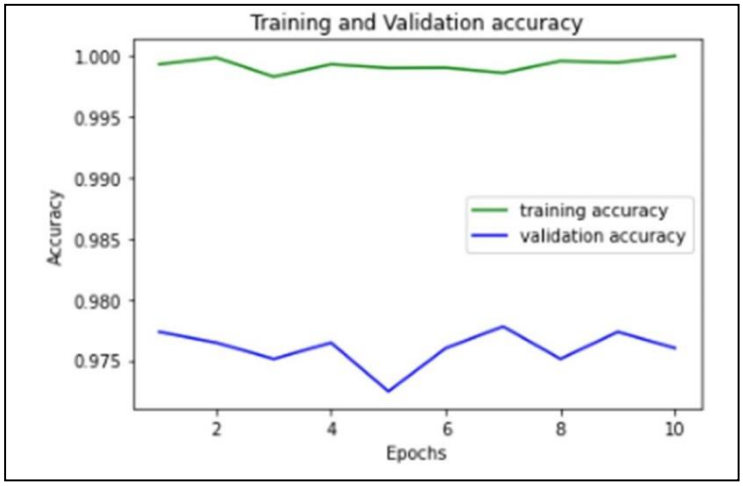
```
Confusion Matrix
[[38 31 33 26 29 22 31 19 21]
 [31 28 25 27 26 26 33 26 28]
 [22 18 28 34 30 36 33 21 28]
 [32 21 23 34 30 24 42 22 22]
 [29 23 29 18 25 30 32 30 34]
 [20 29 27 26 32 25 32 22 37]
 [27 30 26 32 21 31 33 26 24]
 [26 41 25 26 24 26 30 25 27]
 [25 29 33 28 33 30 29 14 29]]

Classification Report
              precision    recall  f1-score   support

           A       0.15      0.15      0.15       250
           B       0.11      0.11      0.11       250
           C       0.11      0.11      0.11       250
           D       0.14      0.14      0.14       250
           E       0.10      0.10      0.10       250
           F       0.10      0.10      0.10       250
           G       0.11      0.13      0.12       250
           H       0.12      0.10      0.11       250
           I       0.12      0.12      0.12       250

    accuracy                           0.12      2250
   macro avg       0.12      0.12      0.12      2250
weighted avg       0.12      0.12      0.12      2250
```

- Accuracy



10. ADVANTAGES & DISADVANTAGES

Advantages

- The application is conveniently simple for the end user.
- The user interface is not complex.

Disadvantages

- The dataset in limited.
- The alphabets only range from 'A' to 'I'.
- As of now, only static gestures are converted.

11. CONCLUSION

The main objective of this project is to develop gesture recognition so that the deaf can communicate with normal individuals. One of the crucial tasks is the extraction of features, and various gestures should yield various, effectively distinguishable characteristics. To identify the character from the gesture images, we used a trained dataset for the CNN algorithm. These features combined with a labelled data enable accurate real- time ASL alphabet recognition. Our analysis found that accuracy is influenced by a variety of elements, including the camera, dataset, and approach. The accuracy drastically declines in low light and noisy backgrounds

12. FUTURE SCOPE

The proposed system can be translated into multiple languages, enhancing its dependability and effectiveness. In the near future, it might only be accessible through mobile devices, making the system more convenient and portable. This system is unable to detect gestures made with both hands. Therefore, detecting gestures done with both hands could be a future project.

13. APPENDIX

13.1 SOURCE CODE

- Source Code is available in the GitHub link provided in Section 13.2.

13.2. GITHUB & PROJECT DEMO LINK

- GitHub Link: https://github.com/IBM-EPBL/IBM-Project-17580-1659673617
- Project Demo Link: https://youtu.be/zV0nUu1EURA