REAL -TIME COMMUNICATION SYSTEM POWERED BY AI FOR SPECIALLY ABLED

NALAIYA THIRAN PROJECT BASED LEARNING ON PROFESSIONAL READLINESS FOR INNOVATION, EMPLOYNMENT AND ENTERPRENEURSHIP

A PROJECT REPORT

SURAJ K-311419104083

SURYA MOORTHY K - 311419104085

ZAFFER AHMED H M - 311419104100

SHADRACH GIDEON S.P - 311419104071

BACHELOR OF ENGINEERING IN COMPUTER SCIENCE AND ENGINEERING

MEENAKSHI COLLEGE OF ENGINEERING

CHENNAI – 600078

1. INTRODUCTION

1.1 Project Overview

Category: Artificial Intelligence

Team ID: PNT2022TMID27699

Skills Required:

Python, CNN, IBM Cloud, IBM Watson Studio, IBM Cloudant DB, Deep Learning, Python-Flask

Project Description:

In our society, we have people with disabilities. The technology is developing day by day but no significant developments are undertaken for the betterment of these people. Communications between deaf-mute and a normal person has always been a challenging task. It is very difficult for mute people to convey their message to normal people. Since normal people are not trained on hand sign language. In emergency times conveying their message is very difficult. The human hand has remained a popular choice to convey information in situations where other forms like speech cannot be used. 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.

The project aims to develop a system that converts the sign language into a human hearing voice in the desired language to convey a message to normal people, as well as convert speech into understandable sign language for the deaf and dumb. We are making use of a convolution neural network to create a model that is trained on different hand gestures. An app is built which uses this model. This app enables deaf and dumb people to convey their information using signs which get converted to human-understandable language and speech is given as output.

2. LITERATURE SURVEY

S.no	Title	Proposed Work	Tool/ Algorithm	Technology	Advantages/ Disadvantages
1.	Roger Voice	Can understand voice calls. You get to pilot your own calls. The caller's voice is analysed by an algorithm and Automatically transforms into text.	Automated real-time captioning, Android studio	Artificial Intelligence	[+] Can Recognize your specific voice [-] Less efficient noisy environment
2.	Amazon Echo	Control your other smart home devices including your TV. Make phone calls. Access information, news, weather, cooking tips, and basically anything else you want to know.	Natural Language Processing, java, python	Artificial Intelligence	[+] Easy to use all you got to do is speak to command it. [-] May not be safe as finger print and face recognition if it comes for safety.
3.	Otter voice meeting notes	It is a tool that converts voice conversations into smart notes by recording the	Real time transcription meeting notes, Google colab	Artificial Intelligence	[+] Good Features to edit and highlight text, ability to search keywords

		audio and to provide machine generated transcription. These transcriptions/ notes can be edited, shared, and easily searched.			[-] Editing words that are interpreted wrong is time consuming
4.	Google Assistant	It offers voice commands, voice searching, and voice activated device control, letting you complete a number of tasks after you've said the "OK Google" or "Hey Google" wake words. It is designed to give you conversationa l interactions.	Java Script, C++,Java	Artificial Intelligence	[+] Efficiently used [-] But the speaker should be loud otherwise it is hard to recognize their voice

2.1 EXISTING PROBLEM

Some of the existing solutions for solving this problem are:

Technology

One of the easiest ways to communicate is through technology such as a smart phone or laptop. A deaf person can type out what they want to say and a person who is blind or has low vision can use a screen reader to read the text out loud. A blind person can also use voice recognition software to convert what they are saying in to text so that a person who is Deaf can then read it.

Interpreter

If a sign language interpreter is available, this facilitates easy communication if the person who is deaf is fluent in sign language. The deaf person and person who is blind can communicate with each other via the interpreter. The deaf person can use sign language and the interpreter can speak what has been said to the person who is blind and then translate anything spoken by the blind person into sign language for the deaf person.

2.2 Proposed solution

This paper describes the system that overcomes the problem faced by the speech and hearing impaired. The objectives of the research are as follow:

1. To design and develop a system which lowers the communication gap between speech hearing impaired and normal world.
2. To build a communication system that enables communications between deaf-dumb person and a normal person.
3. A convolution neural network is being used to develop a model that is trained on various hand movements. This model is used to create an app. This programme allows deaf and hard of hearing persons to communicate using signs that are then translated into human readable text.

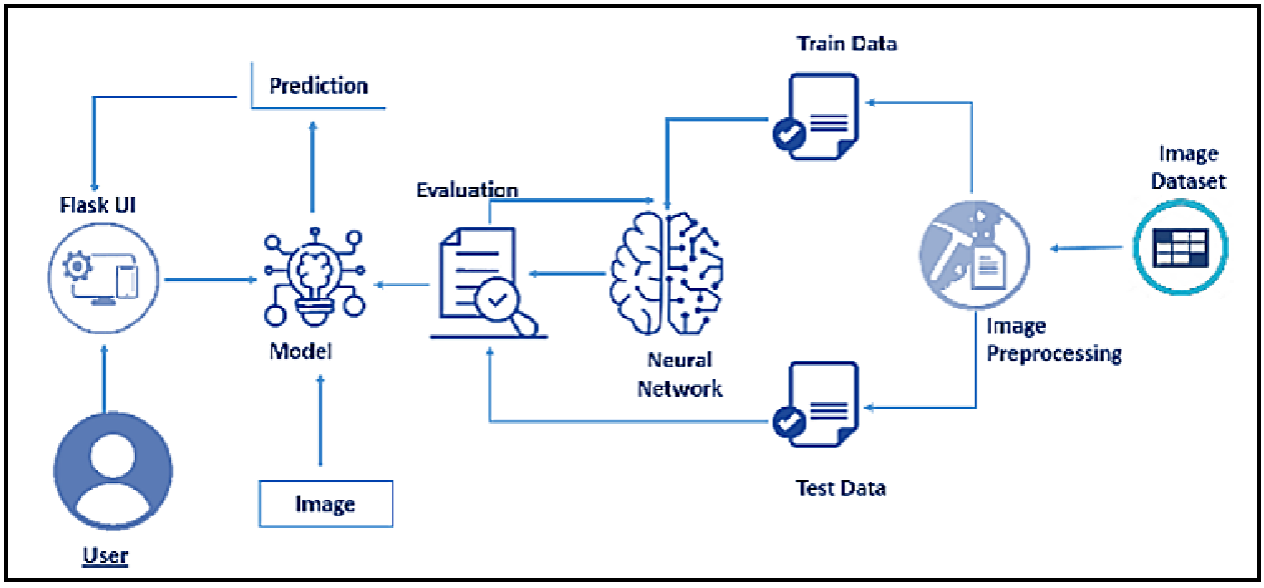
2.3 Problem Statement

I am	I'm trying to	But	Because	Which makes me feel
Disabled person	Communicate online, and use applications without any struggles	I can't send any mails or make phone calls	Of my disability	Upset

3. THEORITICAL ANALYSIS

3.1 Block diagram

Architecture:



3.2 Hardware / Software designing

Hardware Requirements

Operating System	Windows, Mac, Linux
CPU (for training)	Multi Core Processors (i3 or above/equivalent)
GPU (for training)	NVIDIA AI Capable / Google's TPU
WebCam	Integrated or External with FullHD Support

Software Requirements:

Python	v3.9.0 or Above
Python Packages	flask, tensorflow, opencv-python, keras, numpy, pandas, virtualenv, pillow
Web Browser	Mozilla Firefox, Google Chrome or any modern web browser
IBM Cloud (for training)	Watson Studio - Model Training & Deployment as Machine Learning Instance

4. EXPERIMENTAL INVESTIGATIONS

Training and Testing using Dataset Provided:

In [27]:
```
from tensorflow.keras.preprocessing.image import ImageDataGenerator
```

In [28]:
```
# Training Datagen
train_datagen = ImageDataGenerator(rescale=1/255,zoom_range=0.2,horizontal_flip=True,vertical_flip=False)
# Testing Datagen
test_datagen = ImageDataGenerator(rescale=1/255)
```

In [29]:
```
# Training Dataset
x_train=train_datagen.flow_from_directory('C:/Users/minec/Desktop/IBM PROJECT/Dataset/training_set',target_size=(64,64), class_mode='categorical',batc
# Testing Dataset
x_test=test_datagen.flow_from_directory('C:/Users/minec/Desktop/IBM PROJECT/Dataset/test_set',target_size=(64,64), class_mode='categorical',batch_size
```

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

In [30]:
```
import pandas
from sklearn import model_selection
from sklearn.linear_model import LogisticRegression
import pickle
```

In [30]:
```
import pandas
from sklearn import model_selection
from sklearn.linear_model import LogisticRegression
import pickle
```

In [31]:
```
url = "https://raw.githubusercontent.com/jbrownlee/Datasets/master/pima-indians-diabetes.data.csv"
names = ['preg', 'plas', 'pres', 'skin', 'test', 'mass', 'pedi', 'age', 'class']
dataframe = pandas.read_csv(url, names=names)
array = dataframe.values
X = array[:,0:8]
Y = array[:,8]
test_size = 0.33
seed = 7
X_train, X_test, Y_train, Y_test = model_selection.train_test_split(X, Y, test_size=test_size,
random_state=seed)

# Fit the model on training set
model = LogisticRegression()
model.fit(X_train, Y_train)
# save the model to disk
filename = 'finalized_model.sav'
pickle.dump(model, open(filename, 'wb'))

# load the model from disk
loaded_model = pickle.load(open(filename, 'rb'))
result = loaded_model.score(X_test, Y_test)
print(result)
```

```
0.7874015748031497
```

```
C:\Users\minec\anaconda3\lib\site-packages\sklearn\linear_model\_logistic.py:814: ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  n_iter_i = _check_optimize_result(
```

In [32]:
```
print("Len x-train : ", len(x_train))
print("Len x-test : ", len(x_test))
```

```
Len x-train :  18
Len x-test :  3
```

In [33]:
```
# The Class Indices in Training Dataset
x_train.class_indices
```

Out[33]:
```
{'A': 0, 'B': 1, 'C': 2, 'D': 3, 'E': 4, 'F': 5, 'G': 6, 'H': 7, 'I': 8}
```

In [34]:
```
# Importing Libraries
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Convolution2D,MaxPooling2D,Flatten,Dense
```

In [35]:
```
# Creating Model
model=Sequential()

# Adding Layers
model.add(Convolution2D(32,(3,3),activation='relu',input_shape=(64,64,3)))
```

In [36]:
```
model.add(MaxPooling2D(pool_size=(2,2)))
```

In [40]:
```
# Fitting the Model Generator
model.fit_generator(x_train,steps_per_epoch=len(x_train),epochs=10,validation_data=x_test,validation_steps=len(x_test))
```

```
C:\Users\minec\AppData\Local\Temp\ipykernel_12712\1042518445.py:2: UserWarning: `Model.fit_generator` is deprecated and will be removed in a future ve
rsion. Please use `Model.fit`, which supports generators.
  model.fit_generator(x_train,steps_per_epoch=len(x_train),epochs=10,validation_data=x_test,validation_steps=len(x_test))
Epoch 1/10
18/18 [==============================] - 28s 2s/step - loss: 1.1885 - accuracy: 0.6356 - val_loss: 0.3970 - val_accuracy: 0.9084
Epoch 2/10
18/18 [==============================] - 23s 1s/step - loss: 0.2429 - accuracy: 0.9309 - val_loss: 0.2971 - val_accuracy: 0.9409
Epoch 3/10
18/18 [==============================] - 22s 1s/step - loss: 0.0933 - accuracy: 0.9761 - val_loss: 0.1903 - val_accuracy: 0.9724
Epoch 4/10
18/18 [==============================] - 22s 1s/step - loss: 0.0483 - accuracy: 0.9889 - val_loss: 0.2213 - val_accuracy: 0.9733
Epoch 5/10
18/18 [==============================] - 22s 1s/step - loss: 0.0281 - accuracy: 0.9933 - val_loss: 0.2241 - val_accuracy: 0.9733
Epoch 6/10
18/18 [==============================] - 21s 1s/step - loss: 0.0201 - accuracy: 0.9953 - val_loss: 0.2540 - val_accuracy: 0.9756
Epoch 7/10
18/18 [==============================] - 22s 1s/step - loss: 0.0122 - accuracy: 0.9975 - val_loss: 0.2513 - val_accuracy: 0.9756
Epoch 8/10
18/18 [==============================] - 21s 1s/step - loss: 0.0089 - accuracy: 0.9984 - val_loss: 0.2877 - val_accuracy: 0.9769
Epoch 9/10
18/18 [==============================] - 23s 1s/step - loss: 0.0065 - accuracy: 0.9990 - val_loss: 0.2771 - val_accuracy: 0.9764
Epoch 10/10
18/18 [==============================] - 21s 1s/step - loss: 0.0055 - accuracy: 0.9991 - val_loss: 0.2952 - val_accuracy: 0.9760
```

Out[40]:

In [41]:
```
model.save('asl_model_84_54.h5')
```

In [ ]:

In [36]:
```
model.add(MaxPooling2D(pool_size=(2,2)))
```

In [37]:
```
model.add(Flatten())
```

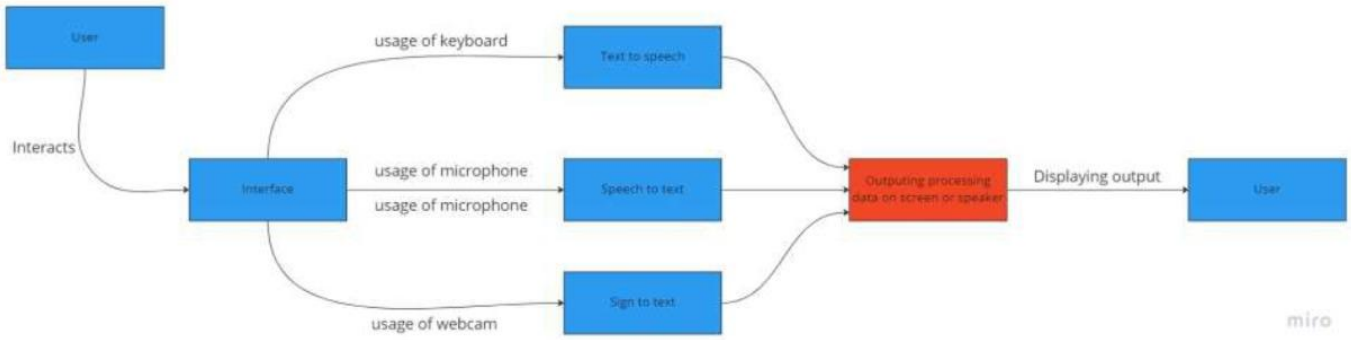
In [38]:
```
# Adding Dense Layers
model.add(Dense(300,activation='relu'))
model.add(Dense(150,activation='relu'))
model.add(Dense(9,activation='softmax'))
```

In [39]:
```
# Compiling the Model
model.compile(loss='categorical_crossentropy',optimizer='adam',metrics=['accuracy'])
```

In [40]:
```
# Fitting the Model Generator
model.fit_generator(x_train,steps_per_epoch=len(x_train),epochs=10,validation_data=x_test,validation_steps=len(x_test))
```

5. DATA FLOW

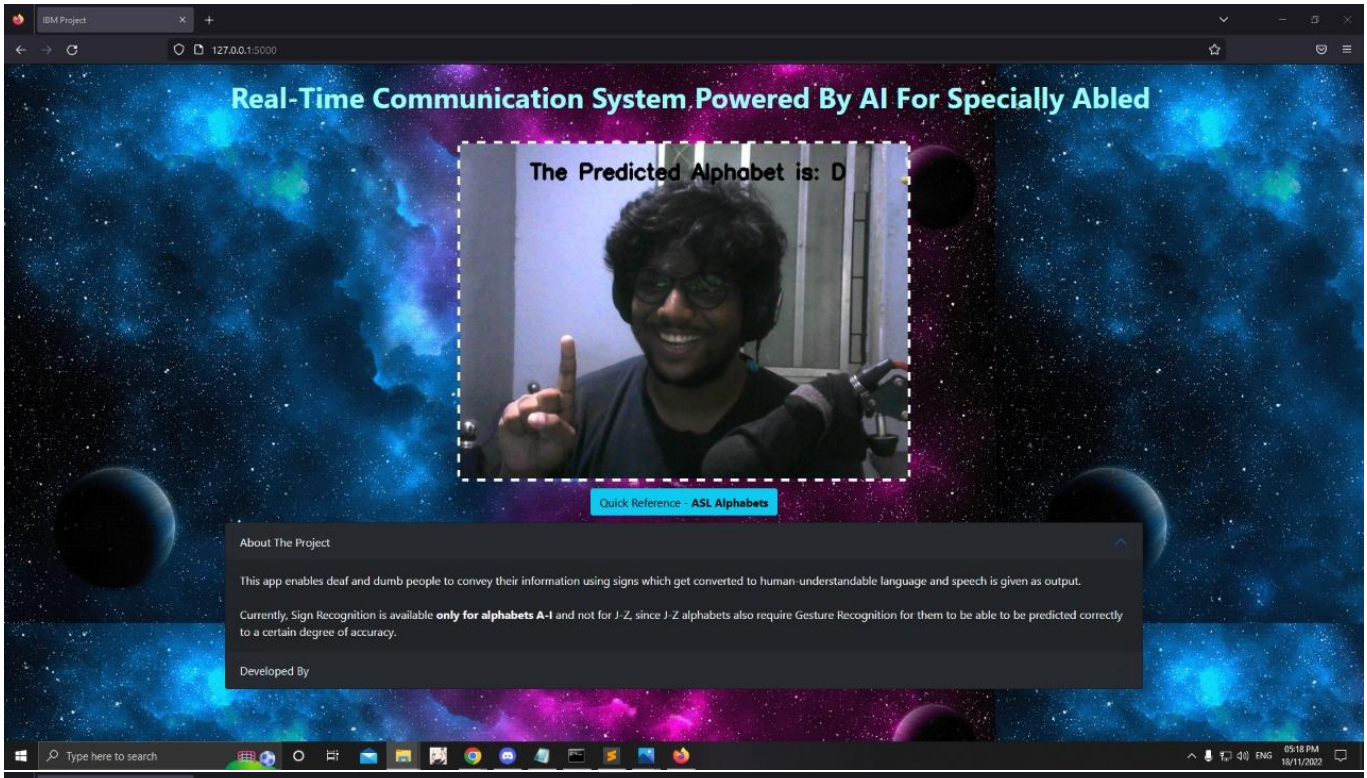
Data Flow Diagram:

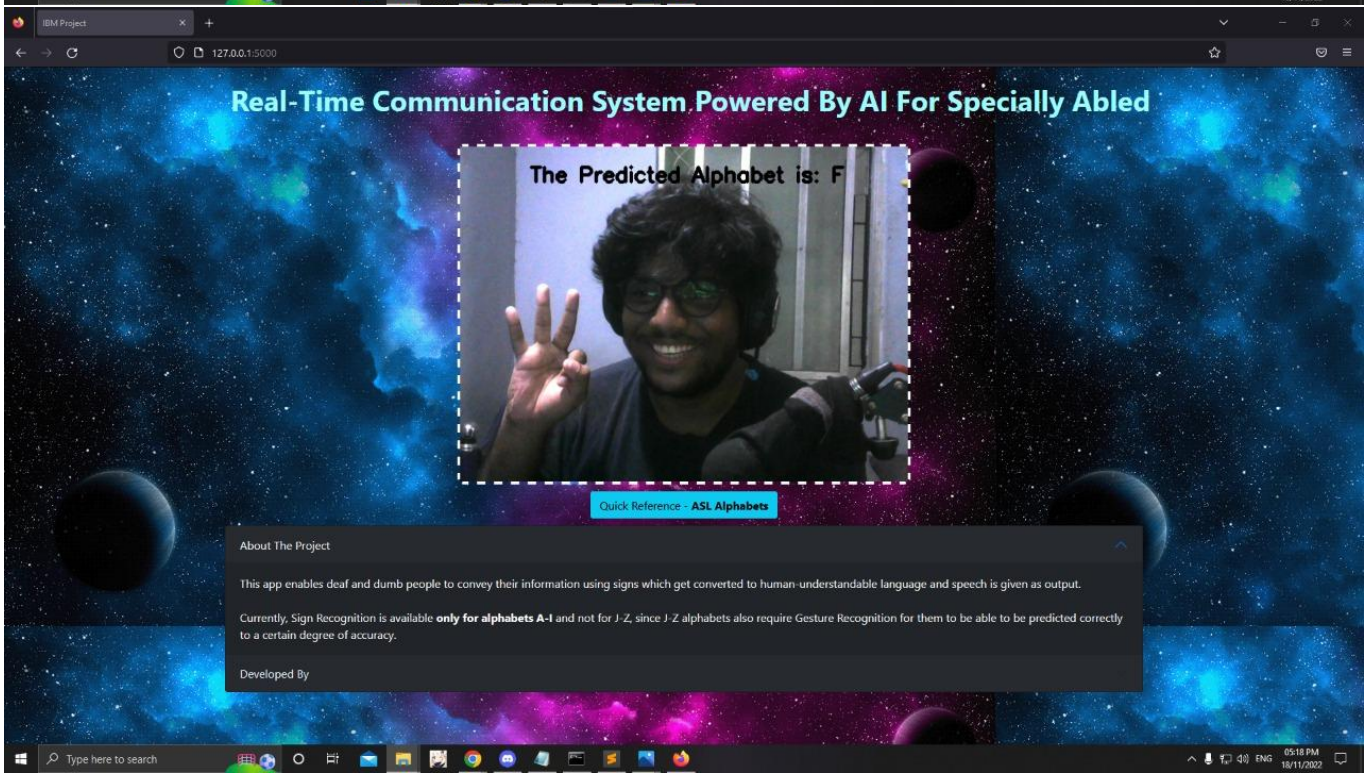


6. RESULT

The proposed procedure was implemented and tested with set of images. The set of 15750 images of Alphabets from “A” to “I” are used for training database and a set of 2250 images of Alphabets from “A” to “I” are used for testing database. Once the gesture is recognise the equivalent Alphabet is shown on the screen.

Some sample images of the output are provided below:


IBM Project
127.0.0.1:5000
Real-Time Communication System Powered By AI For Specially Abled
The Predicted Alphabet is: D
Quick Reference - ASL Alphabets
About The Project
This app enables deaf and dumb people to convey their information using signs which get converted to human-understandable language and speech is given as output.
Currently, Sign Recognition is available only for alphabets A-I and not for J-Z, since J-Z alphabets also require Gesture Recognition for them to be able to be predicted correctly to a certain degree of accuracy.
Developed By
Type here to search
ENG
05:18 PM
18/11/2022


IBM Project
127.0.0.1:5000
Real-Time Communication System Powered By AI For Specially Abled
The Predicted Alphabet is: F
Quick Reference - ASL Alphabets
About The Project
This app enables deaf and dumb people to convey their information using signs which get converted to human-understandable language and speech is given as output.
Currently, Sign Recognition is available only for alphabets A-I and not for J-Z, since J-Z alphabets also require Gesture Recognition for them to be able to be predicted correctly to a certain degree of accuracy.
Developed By
Type here to search
ENG
05:18 PM
18/11/2022

7. ADVANTAGES & DISADVANTAGES

Advantages:

1. It is possible to create a mobile application to bridge the communication gap between deaf and dumb persons and the general public.

2. As different sign language standards exist, their dataset can be added, and the user can choose which sign language to read.

Disadvantages:

1. The current model only works from alphabets A to I.

2. In absence of gesture recognition, alphabets from J cannot be identified as they require some kind of gesture input from the user.

3. As the quantity/quality of images in the dataset is low, the accuracy is not great, but that can easily be improved by change in dataset.

8. APPLICATIONS

1. It will contribute to the development of improved communication for the deafened. The majority of people are unable to communicate via sign language, which creates a barrier to communication.
2. As a result, others will be able to learn and comprehend sign language and communicate with the deaf and dumb via the web app.
3. According to scientific research, learning sign language improves cognitive abilities, attention span, and creativity.

9. CONCLUSION

Sign language is a useful tool for facilitating communication between deaf and hearing people. Because it allows for two-way communication, the system aims to bridge the communication gap between deaf people and the rest of society. The proposed methodology translates language into English alphabets that are understandable to humans.

This system sends hand gestures to the model, who recognises them and displays the equivalent Alphabet on the screen. Deaf-mute people can use their hands to perform sign language, which will then be converted into alphabets, thanks to this project.

10. FUTURE SCOPE

Having a technology that can translate hand sign language to its corresponding alphabet is a game changer in the field of communication and Ai for the specially abled people such as deaf and dumb. With introduction of gesture recognition, the web app can easily be expanded to recognize letters beyond 'I', digits and other symbols plus gesture recognition can also allow controlling of software/hardware interfaces.

11. BIBILOGRAPHY

1. Environment Setup: https://www.youtube.com/watch?v=5mDYijMfSzs
2. Sign Languages Dataset: https://drive.google.com/file/d/1ITbDvhLwyTTkuUYfNjOKhcIZh7hDgi64/view?usp=sharing
3. Keras Image Processing Doc: https://keras.io/api/preprocessing/image/
4. Keras ImageDataset From Directory Doc: https://keras.io/api/preprocessing/image/#imagedatasetfromdirectory-function
5. CNN using Tensorflow: https://www.youtube.com/watch?v=umGJ30-15_A
6. OpenCV Basics of Processing Image: https://www.youtube.com/watch?v=mjKd1Tzl70I
7. Flask Basics: https://www.youtube.com/watch?v=lj4I_CvBnt0
8. IBM Academic Partner Account Creation: https://www.youtube.com/watch?v=x6i43M7BAqE
9. CNN Deployment and Download through IBM Cloud: https://www.youtube.com/watch?v=BzouqMGJ41k

12. APPENDIX

Source Code for Model Training and Saving:

C:\Users\minec\Desktop\IBM THINGS\Application\Final\Real-Time-Communication-Specially-Abled\ProjectFiles\Flask\app.py - Sublime Text (UNREGISTERED)

File Edit Selection Find View Goto Tools Project Preferences Help

app.py | camera.py | index.html | main.py

```
1  from flask import Flask, Response, render_template
2  from camera import Video
3
4  app = Flask(__name__)
5  @app.route('/')
6  def index():
7      return render_template('index.html')
8
9  def gen(camera):
10     while True:
11         frame = camera.get_frame()
12         yield(b'--frame\r\n'
13             b'Content-Type: image/jpeg\r\n\r\n' + frame +
14             b'\r\n\r\n')
15
16 @app.route('/video_feed')
17 def video_feed():
18     video = Video()
19     return Response(gen(video), mimetype='multipart/x-mixed-replace; boundary = frame')
20
21
22 if __name__ == '__main__':
23     app.run()
```

C:\Users\minec\Desktop\IBM THINGS\Application\Final\Real-Time-Communication-Specially-Abled\ProjectFiles\Flask\camera.py - Sublime Text (UNREGISTERED)

File Edit Selection Find View Goto Tools Project Preferences Help

camera.py | app.py | index.html | main.py

```python
import cv2
import numpy as np
from tensorflow.keras.models import load_model
from tensorflow.keras.preprocessing import image

class Video(object):
    def __init__(self):
        self.video = cv2.VideoCapture(0)
        self.roi_start = (50, 150)
        self.roi_end = (250, 350)
        self.model = load_model('asl_model.h5') # Execute Local Trained Model
        # self.model = load_model('IBM_Communication_Model.h5') # Execute IBM Trained Model
        self.index=['A','B','C','D','E','F','G','H','I']
        self.y = None
    def __del__(self):
        self.video.release()
    def get_frame(self):
        ret,frame = self.video.read()
        frame = cv2.resize(frame, (640, 480))
        copy = frame.copy()
        copy = copy[150:150+200,50:50+200]
        # Prediction Start
        cv2.imwrite('image.jpg',copy)
        copy_img = image.load_img('image.jpg', target_size=(64,64))
        x = image.img_to_array(copy_img)
        x = np.expand_dims(x, axis=0)
        pred = np.argmax(self.model.predict(x), axis=1)
        self.y = pred[0]
        cv2.putText(frame,'The Predicted Alphabet is: '+str(self.index[self.y]),(100,50),cv2.FONT_HERSHEY_SIMPLEX,1,(0,0,0),3)
        ret,jpg = cv2.imencode('.jpg', frame)
        return jpg.tobytes()
```

C:\Users\minec\Desktop\IBM THINGS\Application\Final\Real-Time-Communication-Specially-Abled\ProjectFiles\Flask\templates\index.html - Sublime Text (UNREGISTERED)

File Edit Selection Find View Goto Tools Project Preferences Help

index.html | camera.py | app.py | main.py

```html
<!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>IBM Project</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="assets/css/Banner-Heading-Image.css">
    <link rel="stylesheet" href="assets/css/Navbar-Centered-Brand.css">
    <link rel="stylesheet" href="assets/css/styles.css">
</head>

<style>
    body{
    background-image: url("{{ url_for('static', filename='img/back.png') }}")
    }
</style>

<body>
    <nav class="navbar navbar-light navbar-expand-md py-3" style="background-image: linear-gradient(to right, purple, light blue);">
        <div class="container">
            <div></div><a class="navbar-brand d-flex align-items-center" href="#"><span
                    class="bs-icon-sm bs-icon-rounded bs-icon-primary d-flex justify-content-center align-items-center me-2 bs-icon"></span>
                    <span style="color: rgb(155,255,255);"><h1><strong>Real-Time Communication
                    System Powered By AI&amp;nbsp;For Specially Abled</strong></h1></span></a>
            <div></div>
        </div>
    </nav>
    <section>
        <div class="d-flex flex-column justify-content-center align-items-center">
            <div class="d-flex flex-column justify-content-center align-items-center" id="div-video-feed"
                style="width: 640px;height: 480px;margin: 10px;min-height: 480px;min-width: 640px;border-radius: 10px;border: 4px dashed rgb(255,255,255) ;">
                <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!">
            </div>
        </div>
        <div class="d-flex flex-column justify-content-center align-items-center" style="margin-bottom: 10px;"><button
                class="btn btn-info" type="button" data-bs-target="#modal-1" data-bs-toggle="modal">Quick Reference
                -<strong> ASL Alphabets</strong></button></div>
    </section>
    <section>
        <div class="container">
            <div class="accordion text-white" role="tablist" id="accordion-1">
                <div class="accordion-item" style="background: rgb(33,37,41);">
                    <h2 class="accordion-header" role="tab"><button class="accordion-button" data-bs-toggle="collapse"
                            data-bs-target="#accordion-1 .item-1" aria-expanded="true"
                            aria-controls="accordion-1 .item-1"
                            style="background: rgb(39,43,48);color: rgb(255,255,255);">About The Project</button></h2>
```

```
                    <div class="accordion-collapse collapse show item-1" role="tabpanel" data-bs-parent="#accordion-1">
                        <div class="accordion-body">
                            <p class="mb-0">This app enables deaf and dumb people to convey their information using signs which get converted to human-understandable language and speech is given as output.<br><br>Currently, sign Recognition is available <strong>only for
                                    alphabets A-I</strong> and not for J-Z, since J-Z alphabets also require Gesture
                                Recognition for them to be able to be predicted correctly to a certain degree of
                                accuracy.</p>
                        </div>
                    </div>
                </div>
                <div class="accordion-item" style="background: rgb(33,37,41);">
                    <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="background: rgb(33,43,48);color: rgb(231,241,255);">Developed By</button></h2>
                    <div class="accordion-collapse collapse item-2" role="tabpanel" data-bs-parent="#accordion-1">
                        <div class="accordion-body">
                            <p class="mb-0">Students of Meenakshi College of Engineering,
                                <br>1. <strong>Suraj Kc</strong> <br>2.
                                <strong>Surya Moorthy Kc</strong> <br>3. <strong>Shadrach Gideon S Pc</strong> <br>4. <strong>Zaffer Ahmed H Mc</strong>
                            </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>
```

C:\Users\minec\Desktop\IBM THINGS\Application\Final\Real-Time-Communication-Specially-Abled\Project

File Edit Selection Find View Goto Tools Project Preferences Help

main.py | index.html | camera.py | app.py

```python
import cv2

video = cv2.VideoCapture(0)

while True:
    ret, frame = video.read()
    cv2.imshow("Frame", frame)
    k = cv2.waitKey(1)
    if k == ord('q'):
        break

video.release()
cv2.destroyAllWindows()
```

American Sign Language Standard Reference: