CONVERSATION ENGINE FOR DEAF AND DUMB USING IBM WATSON

Mini Project documentation submitted to

JAWAHARLAL NEHRU TECNOLOGICAL UNIVERSITY, HYDERABAD

In partial fulfillment of the requirements for the award of the degree

BACHELOR OF TECHNOLOGY

In

COMPUTER SCIENCE AND ENGINEERING

Submitted By

AKSHARA	**18UK1A05B8**
NIRANJAN	**18UK1A05D4**
AKHIL	**18UK1A0598**
SHIVA KUMAR	**18UK1A0585**

Under the guidance of

MS. S. ANOOSHA

(Assistant Professor)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

VAAGDEVI ENGINEERING COLLEGE

Affiliated to JNTUH, HYDERABAD

BOLLIKUNTA, WARANGAL (T.S) – 506005

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

VAAGDEVI ENGINEERING COLLEGE

WARANGAL

CERTIFICATE

This is to certify that the Mini project report entitled **“CONVERSATION ENGINE FOR DEAF AND DUMB USING IBM WASTON”**isbeingsubmitted by **AKSHARA(18UK1A05B8),NIRANJAN(18UK1A05D4),AKHIL(18UK1A0598),SHIVA (18UK1A0585)** in partial fulfillment of the requirements for the award of the degree of Bachelor of Technology in Computer Science & Engineering to Jawaharlal Nehru Technological University Hyderabad during the academic year 2021- 2022.

Project Guide **HOD**

MS. S. ANOOSHA **Dr. R. NAVEEN KUMAR**

EXTERNAL

ACKNOWLEDGEMENT

We wish to take this opportunity to express our sincere gratitude and deep sense of respect to our beloved **Dr. P. Prasad Rao**, Principal, Vaagdevi Engineering College for making us available all the required assistance and for his support and inspiration to carry out this Mini Project in the institute.

We extend our heartfelt thanks to **Dr. R. Naveen Kumar**, Head of the Department of CSE, Vaagdevi Engineering College for providing us necessary infrastructure and thereby giving us freedom to carry out the Mini Project.

We express heartfelt thanks to the guide, **MS. S. ANOOSHA** , Assistant Professor, Department of CSE for his constant support and giving necessary guidance for completion of this Mini Project.

Finally, we express our sincere thanks and gratitude to our family members, friends for their encouragement and outpouring their knowledge and experiencing throughout thesis.

ABSTRACT

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. About nine billion people in the world are deaf and dumb. 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 in any language.

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 converting 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.

AKSHARA(18UK1A05B8)

NIRANJAN(18UK1A05D4)

AKHIL(18UK1A0598)

SHIVA KUMAR(18UK1A0585)

TABLE OF CONTENTS

INDEX

6.RESULT

Final finding outputs of the project along with screenshots

7.ADVANTAGES&DISADVANTAGES

List of advantages and disadvantages of the proposed

8.Applications

The areas where the solutions can be applied

9.CONCLUSION

Conclusion summarizing the entrie work and findings

10.FUTURE SCOPE

Enhancement that can be made in the future.

11.BIBILOGRAPHY

References of the pervious year works or website visited.

Analysis about the project, solution previous finding etc..

APPENDIX

A .Source code

Attach the code for the solution built.

Introduction

To establish a communication or interaction with Deaf and Mute people is of utter importance nowadays. These people interact through hand gestures or signs. Gestures are basically the physical action form performed by a person to convey some meaningful information. Gestures are a powerful means of communication among humans. In fact gesturing is so deeply rooted in our communication that people often continue gesturing when speaking on the telephone. There are various signs which express complex meanings and recognising them is a challenging task for people who have no understanding for that language. It becomes difficult finding a well experienced and educated translator for the sign language every time and everywhere but human-computer interaction system for this can be installed anywhere possible. The motivation for developing such helpful application came from the fact that it would prove to be of utmost importance for socially aiding people and how it would help increasingly for social awareness as well. The remarkable ability of the human vision is the gesture recognition, it is noticeable mainly in deaf people when they communicating with each other via sign language and with hearing people as well. In this paper we take up one of the social challenges to give this set of mass a permanent solution in communicating with normal human beings. Sign language is categorized in accordance to regions like Indian, American, Chinese, Arabic and so on and researches on hand gesture recognition, pattern recognitions, image processing have been carried by supposedly countries as well to improve the applications and bring them to the best levels.

Overview

It often comes as a surprise that many deaf individuals refer to themselves as being members of the Deaf community and ascribe to Deaf culture. These individuals view themselves as a unique cultural and linguistic minority who use sign language as their primary language. The characteristics of Deaf culture are formed out of many shared life experiences rooted in a visual world designed for communication ease.

Purpose

The main challenge that this special person facing is the communication gap between special person and normal person. deaf and dumb people always finds difficulties to communicate with normal person. The huge challenge makes them uncomfortable in society. hence they never able to express their feelings.

Literature survey

As mentioned in Introduction that numbers of researches have been carried out as it has become a very influential topic and has been gaining heights of increasing interest. Some methods are explained below: The paper Real Time Hand Gesture Recognition Paper included the algorithm in which first the video was captured and then divided into various frames and the frame with the image was extracted and further from that frame various features like Difference of . Scale space Feature Detector and etc were extracted though SIFT which helped in gesture recognition. model and then recognizing gesture through Genetic Algorithm, in the following applying camshift and HSV model was difficult because making it compatible with different MATLAB versions was not easy and genetic algorithm takes huge amount of time for its development.[2] A method had been developed by P Subha Rajan and Dr G Balakrishnan for recognising gestures for Indian Sign Language where the proposed that each gesture would be recognised through 7 bit orientation and generation process through RIGHT and LEFT scan. The following process required approximately six modules and was a tedious method of recognising signs[3]. A method had been developed by T. Shanableh for recognizing isolated Arabic sign language gestures in a user independent mode. In this method the signers wore gloves to simplify the process of segmenting out the hands of the signer via color segmentation. The effectiveness of the proposed user-independent feature extraction scheme was assessed by two different classification techniques; namely, K-NN and polynomial networks. Many researchers utilized special devices to recognize the Sign Language[4]. Byung - woo min et al, presented the visual recognition of static gesture or image plane, without any external devices. Gestures were spotted by a task specific state transition based on natural human articulation[8]. Static gestures were recognized using image moments of hand posture, while dynamic gestures were recognized by analysing their moving trajectories on the Hidden Markov Models (HMMs). 3. Proposed Method.

Existing problem

In existing system the module was developed for dumb person using flex sensor, there user hand is attached with the flex sensors. On this module the flex sensor reacts on bend of each finger individually. By taking that value controller starts to react with speech, each flex sensor holds unique voice stored in APR Kit and for each sign it will play unique voice. And in other existing system, the work is done only for some alphabets and for the words or sentences.

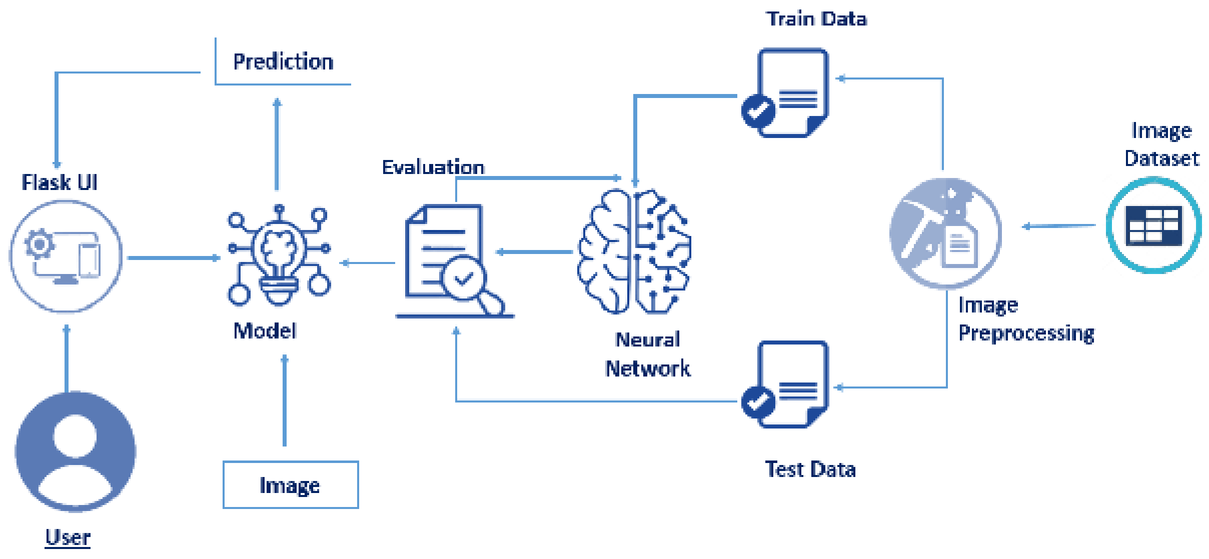
Proposed solution

In the proposed system the unable or dumb person should provide a gesture or sign image to the system. The system evaluates the sign input with matlab image processing technique and classifies the input to the recognized identification. Later it initiates the voice media through the system when the input image matches with the given dataset. And the output will be shown in the text format too. This is a prototype to develop the concept of converting the sign language to speech and text. The aim of this paper is to provide an application to the society to establish the ease of communication between the deaf and mute people by making use of image processing algorithm.

Theoretical Analysis

The local gradient data, used above, is also used to create keypoint descriptors. The gradient information is rotated to line up with the orientation of the keypoint and then weighted by a Gaussian with variance of 1.5 * keypoint scale. This data is then used to create a set of histograms over a window centred on the keypoint[6]. Keypoint descriptors typically uses a set of 16 histograms, aligned in a 4x4 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. These resulting vectors are known as SIFT keys and are used in a nearest-neighbours approach to identify possible objects in an image. Collections of keys that agree on a possible model are identified, when 3 or more keys agree on the model parameters this model is evident in the image with high probability. Due to the large number of SIFT keys in an image of an object, typically a 500x500 pixel image will generate in the region of 2000 features, substantial levels of occlusion are possible while the image is still recognised by this technique. 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 SIFT defined functions the application will first calculate the k of the input image after the 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 of each image one by one and finds the number of matched k , the comparisons with highest matched 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 .

Block Diagram



Software designing

1.Jupyter Notebook Environment

2.Spyder

3.Deep Learning Algorithm

4.Python

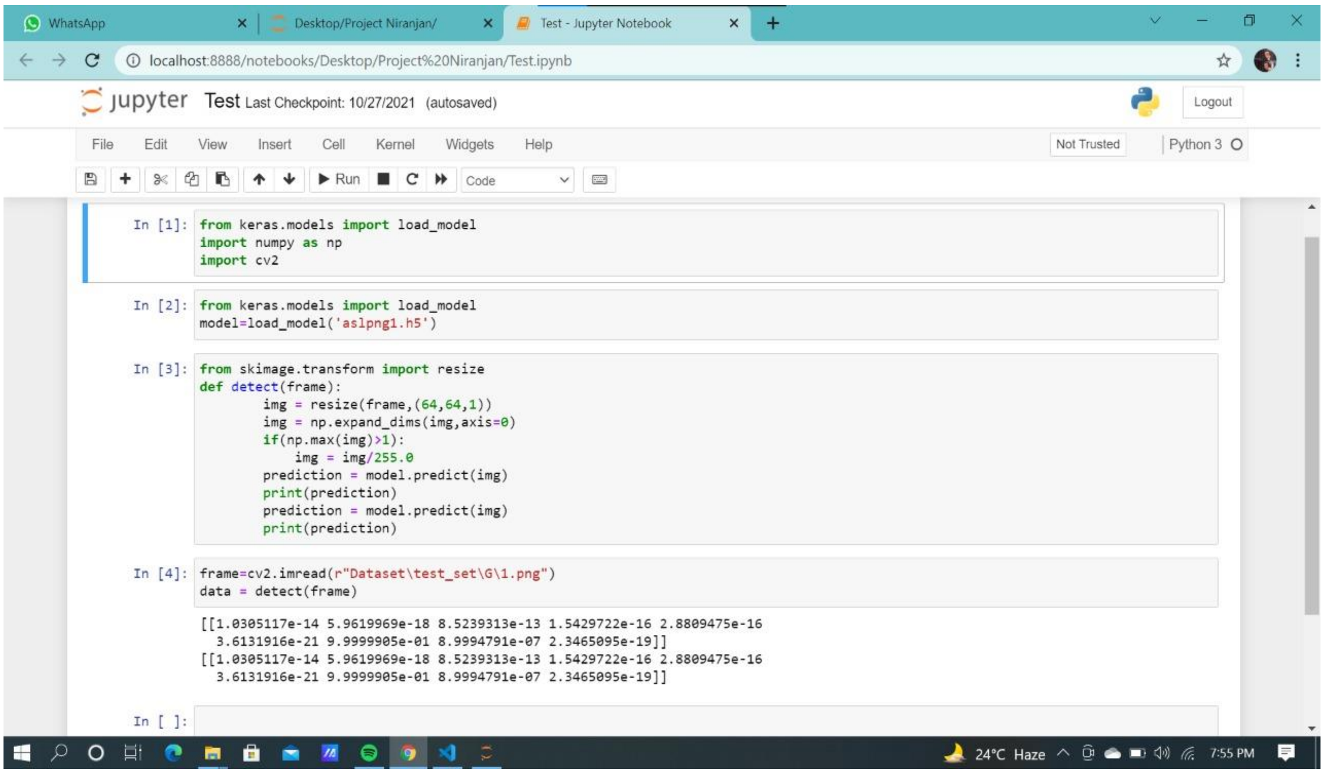
5.Flask

6.HTML

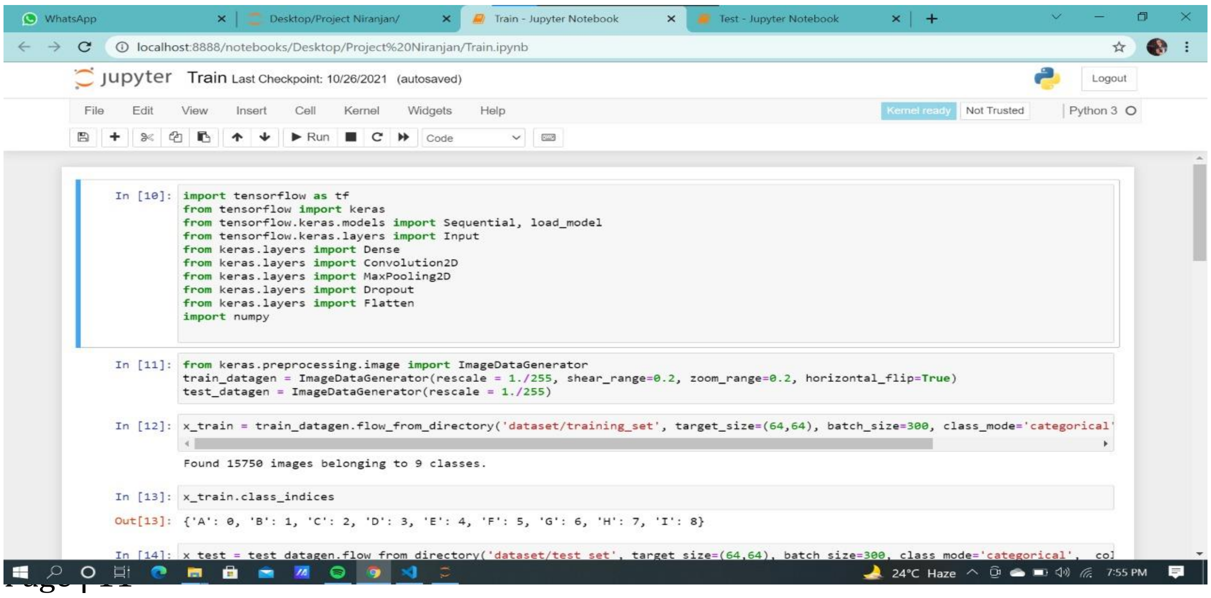
language along with deep learning algorithm such as CNN. For coding we used the Jupyter Notebook of Anaconda distributions and Spyder, We developed this conversation engine of deaf and dumb by using the python language, which is a high level programming an integrated scientific programming in python language. Flask is used as a interface for the prediction. Hyper text mark up language [html] is the standard mark up language for documents designed to be displayed in a web browser.

Experimental investigation

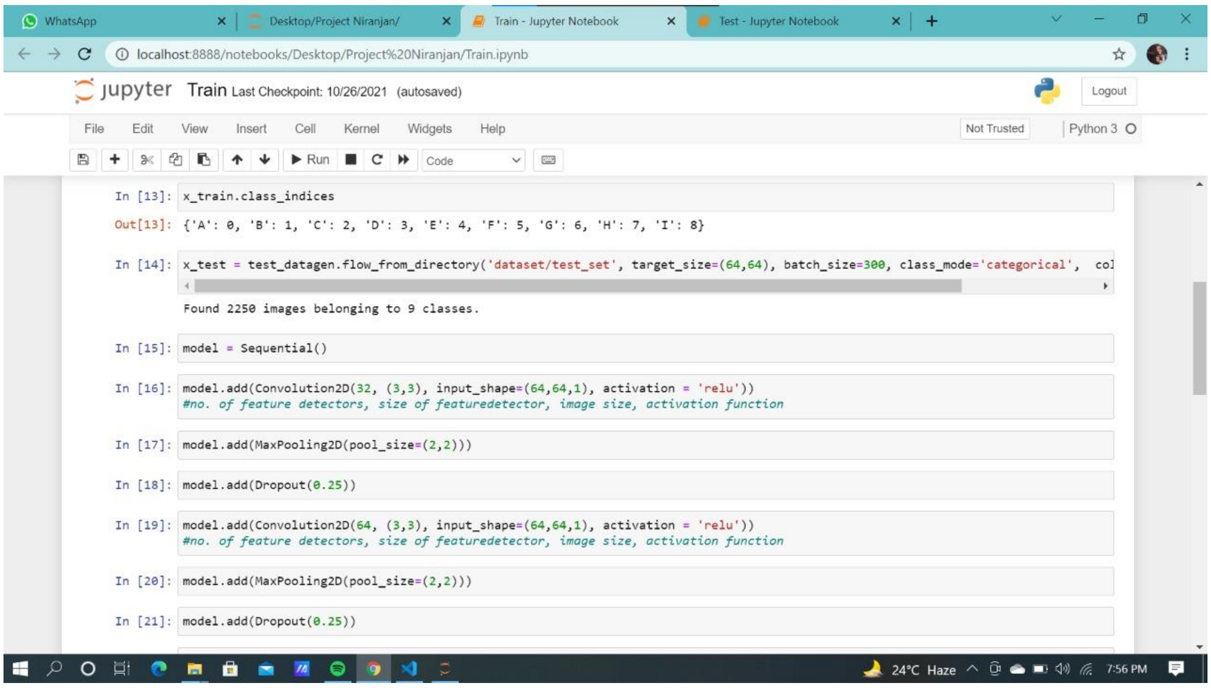
In our project we have used converstion engine for deaf and dumb.This data set contains two folders test and training.Here we have the images having habd gesture posters similarly the training posters also.



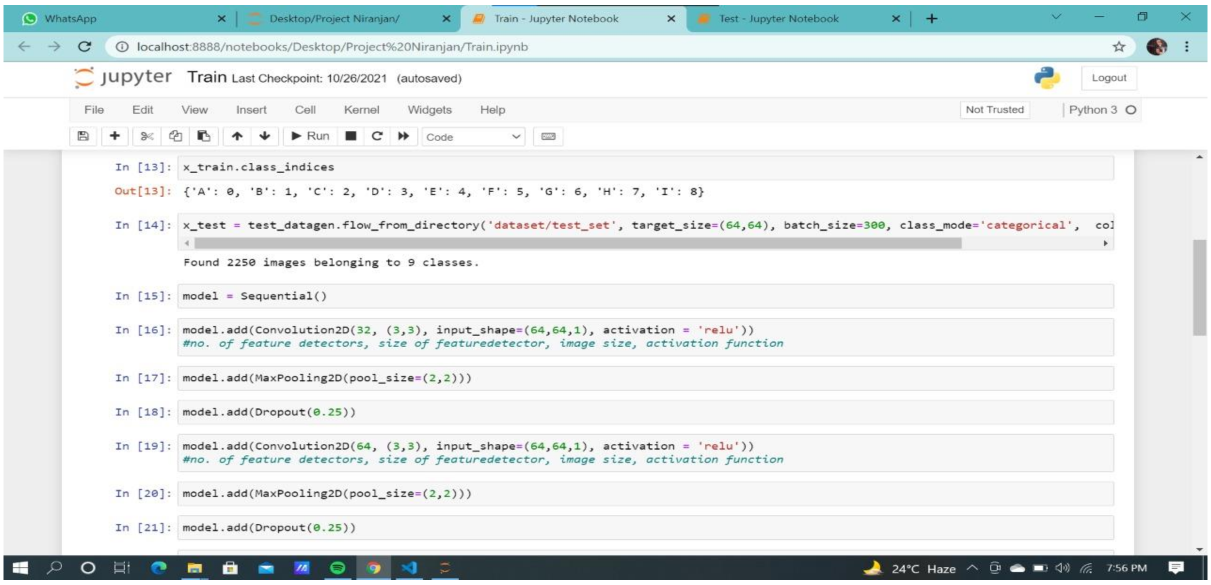
```
In [1]: from keras.models import load_model
        import numpy as np
        import cv2

In [2]: from keras.models import load_model
        model=load_model('aslpng1.h5')

In [3]: from skimage.transform import resize
        def detect(frame):
                img = resize(frame,(64,64,1))
                img = np.expand_dims(img,axis=0)
                if(np.max(img)>1):
                    img = img/255.0
                prediction = model.predict(img)
                print(prediction)
                prediction = model.predict(img)
                print(prediction)

In [4]: frame=cv2.imread(r"Dataset\test_set\G\1.png")
        data = detect(frame)

        [[1.0305117e-14 5.9619969e-18 8.5239313e-13 1.5429722e-16 2.8809475e-16
          3.6131916e-21 9.9999905e-01 8.9994791e-07 2.3465095e-19]]
        [[1.0305117e-14 5.9619969e-18 8.5239313e-13 1.5429722e-16 2.8809475e-16
          3.6131916e-21 9.9999905e-01 8.9994791e-07 2.3465095e-19]]

In [ ]:
```



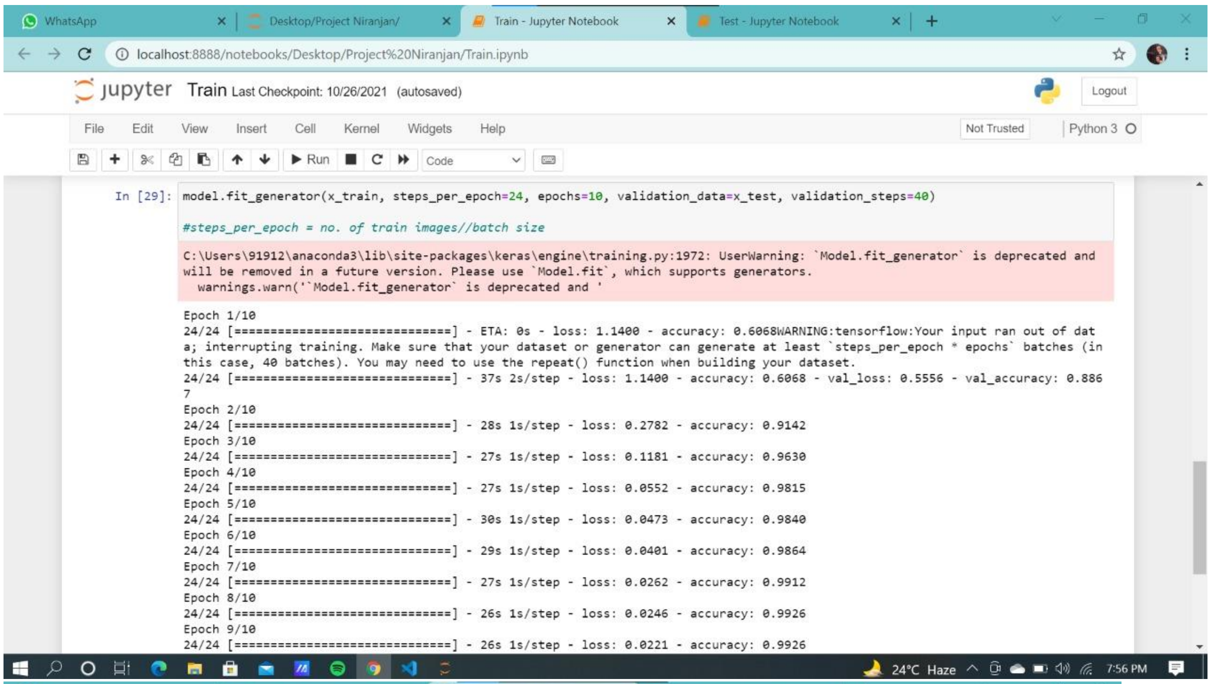
```
In [10]: import tensorflow as tf
         from tensorflow import keras
         from tensorflow.keras.models import Sequential, load_model
         from tensorflow.keras.layers import Input
         from keras.layers import Dense
         from keras.layers import Convolution2D
         from keras.layers import MaxPooling2D
         from keras.layers import Dropout
         from keras.layers import Flatten
         import numpy

In [11]: 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)

In [12]: x_train = train_datagen.flow_from_directory('dataset/training_set', target_size=(64,64), batch_size=300, class_mode='categorical'

         Found 15750 images belonging to 9 classes.

In [13]: x_train.class_indices
Out[13]: {'A': 0, 'B': 1, 'C': 2, 'D': 3, 'E': 4, 'F': 5, 'G': 6, 'H': 7, 'I': 8}

In [14]: x_test = test_datagen.flow_from_directory('dataset/test_set', target_size=(64,64), batch_size=300, class_mode='categorical',  col
```


WhatsApp
Desktop/Project Niranjan/
Train - Jupyter Notebook
Test - Jupyter Notebook
localhost:8888/notebooks/Desktop/Project%20Niranjan/Train.ipynb
jupyter Train Last Checkpoint: 10/26/2021 (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3
Run
Code
24°C Haze
7:56 PM

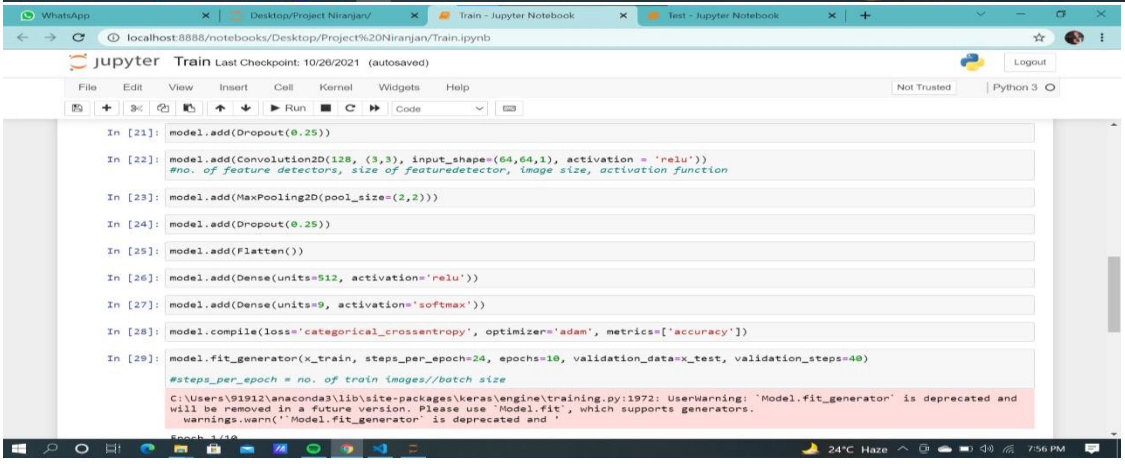
```
In [13]: x_train.class_indices

Out[13]: {'A': 0, 'B': 1, 'C': 2, 'D': 3, 'E': 4, 'F': 5, 'G': 6, 'H': 7, 'I': 8}

In [14]: x_test = test_datagen.flow_from_directory('dataset/test_set', target_size=(64,64), batch_size=300, class_mode='categorical',  col

         Found 2250 images belonging to 9 classes.

In [15]: model = Sequential()

In [16]: model.add(Convolution2D(32, (3,3), input_shape=(64,64,1), activation = 'relu'))
         #no. of feature detectors, size of featuredetector, image size, activation function

In [17]: model.add(MaxPooling2D(pool_size=(2,2)))

In [18]: model.add(Dropout(0.25))

In [19]: model.add(Convolution2D(64, (3,3), input_shape=(64,64,1), activation = 'relu'))
         #no. of feature detectors, size of featuredetector, image size, activation function

In [20]: model.add(MaxPooling2D(pool_size=(2,2)))

In [21]: model.add(Dropout(0.25))
```



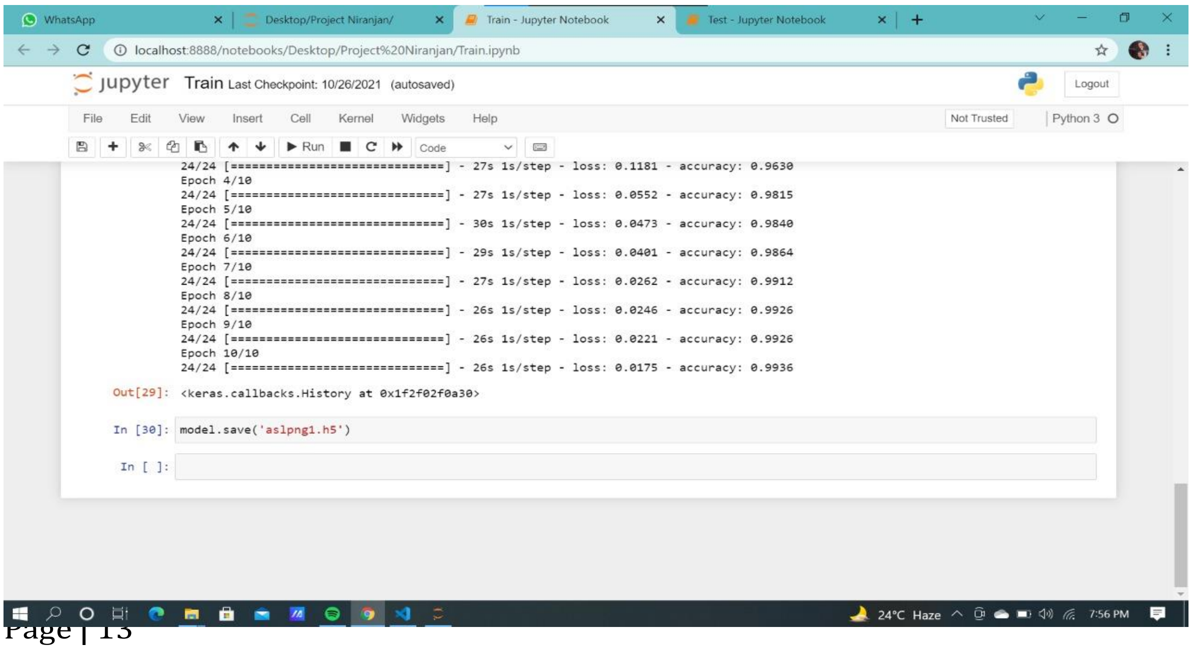
```
In [13]: x_train.class_indices

Out[13]: {'A': 0, 'B': 1, 'C': 2, 'D': 3, 'E': 4, 'F': 5, 'G': 6, 'H': 7, 'I': 8}

In [14]: x_test = test_datagen.flow_from_directory('dataset/test_set', target_size=(64,64), batch_size=300, class_mode='categorical',  col

         Found 2250 images belonging to 9 classes.

In [15]: model = Sequential()

In [16]: model.add(Convolution2D(32, (3,3), input_shape=(64,64,1), activation = 'relu'))
         #no. of feature detectors, size of featuredetector, image size, activation function

In [17]: model.add(MaxPooling2D(pool_size=(2,2)))

In [18]: model.add(Dropout(0.25))

In [19]: model.add(Convolution2D(64, (3,3), input_shape=(64,64,1), activation = 'relu'))
         #no. of feature detectors, size of featuredetector, image size, activation function

In [20]: model.add(MaxPooling2D(pool_size=(2,2)))

In [21]: model.add(Dropout(0.25))
```


WhatsApp
Desktop/Project Niranjan/
Train - Jupyter Notebook
Test - Jupyter Notebook
localhost:8888/notebooks/Desktop/Project%20Niranjan/Train.ipynb
jupyter Train Last Checkpoint: 10/26/2021 (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3
Run
Code
24°C Haze
7:56 PM

```
In [29]: model.fit_generator(x_train, steps_per_epoch=24, epochs=10, validation_data=x_test, validation_steps=40)

         #steps_per_epoch = no. of train images//batch size

C:\Users\91912\anaconda3\lib\site-packages\keras\engine\training.py:1972: UserWarning: `Model.fit_generator` is deprecated and
will be removed in a future version. Please use `Model.fit`, which supports generators.
  warnings.warn('`Model.fit_generator` is deprecated and '

Epoch 1/10
24/24 [==============================] - ETA: 0s - loss: 1.1400 - accuracy: 0.6068WARNING:tensorflow:Your input ran out of dat
a; interrupting training. Make sure that your dataset or generator can generate at least `steps_per_epoch * epochs` batches (in
this case, 40 batches). You may need to use the repeat() function when building your dataset.
24/24 [==============================] - 37s 2s/step - loss: 1.1400 - accuracy: 0.6068 - val_loss: 0.5556 - val_accuracy: 0.886
7
Epoch 2/10
24/24 [==============================] - 28s 1s/step - loss: 0.2782 - accuracy: 0.9142
Epoch 3/10
24/24 [==============================] - 27s 1s/step - loss: 0.1181 - accuracy: 0.9630
Epoch 4/10
24/24 [==============================] - 27s 1s/step - loss: 0.0552 - accuracy: 0.9815
Epoch 5/10
24/24 [==============================] - 30s 1s/step - loss: 0.0473 - accuracy: 0.9840
Epoch 6/10
24/24 [==============================] - 29s 1s/step - loss: 0.0401 - accuracy: 0.9864
Epoch 7/10
24/24 [==============================] - 27s 1s/step - loss: 0.0262 - accuracy: 0.9912
Epoch 8/10
24/24 [==============================] - 26s 1s/step - loss: 0.0246 - accuracy: 0.9926
Epoch 9/10
24/24 [==============================] - 26s 1s/step - loss: 0.0221 - accuracy: 0.9926
```

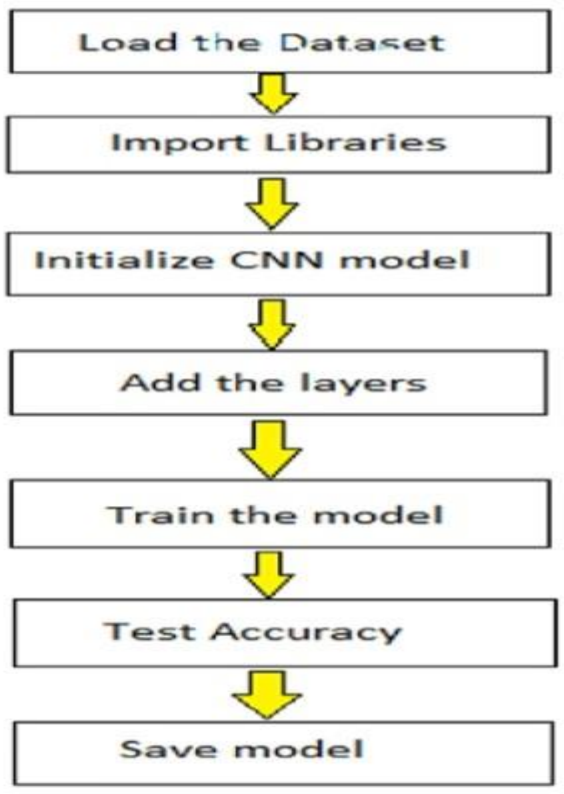

WhatsApp
Desktop/Project Niranjan/
Train - Jupyter Notebook
Test - Jupyter Notebook
localhost:8888/notebooks/Desktop/Project%20Niranjan/Train.ipynb
jupyter Train Last Checkpoint: 10/26/2021 (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3
Run
Code
24°C Haze
7:56 PM

```
In [21]: model.add(Dropout(0.25))

In [22]: model.add(Convolution2D(128, (3,3), input_shape=(64,64,1), activation = 'relu'))
         #no. of feature detectors, size of featuredetector, image size, activation function

In [23]: model.add(MaxPooling2D(pool_size=(2,2)))

In [24]: model.add(Dropout(0.25))

In [25]: model.add(Flatten())

In [26]: model.add(Dense(units=512, activation='relu'))

In [27]: model.add(Dense(units=9, activation='softmax'))

In [28]: model.compile(loss='categorical_crossentropy', optimizer='adam', metrics=['accuracy'])

In [29]: model.fit_generator(x_train, steps_per_epoch=24, epochs=10, validation_data=x_test, validation_steps=40)

         #steps_per_epoch = no. of train images//batch size

C:\Users\91912\anaconda3\lib\site-packages\keras\engine\training.py:1972: UserWarning: `Model.fit_generator` is deprecated and
will be removed in a future version. Please use `Model.fit`, which supports generators.
  warnings.warn('`Model.fit_generator` is deprecated and '
```

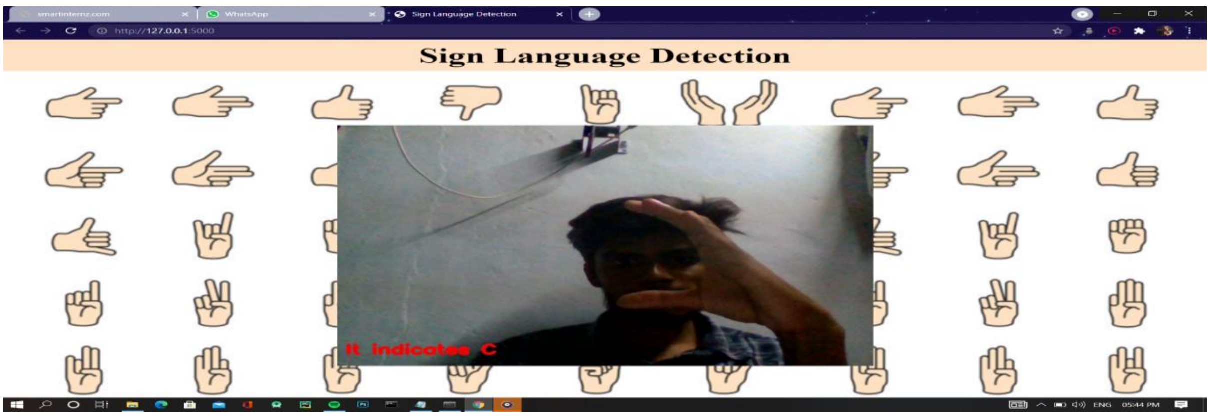

WhatsApp
Desktop/Project Niranjan/
Train - Jupyter Notebook
Test - Jupyter Notebook
localhost:8888/notebooks/Desktop/Project%20Niranjan/Train.ipynb
jupyter Train Last Checkpoint: 10/26/2021 (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3
Run
Code
24°C Haze
7:56 PM

```
24/24 [==============================] - 27s 1s/step - loss: 0.1181 - accuracy: 0.9630
Epoch 4/10
24/24 [==============================] - 27s 1s/step - loss: 0.0552 - accuracy: 0.9815
Epoch 5/10
24/24 [==============================] - 30s 1s/step - loss: 0.0473 - accuracy: 0.9840
Epoch 6/10
24/24 [==============================] - 29s 1s/step - loss: 0.0401 - accuracy: 0.9864
Epoch 7/10
24/24 [==============================] - 27s 1s/step - loss: 0.0262 - accuracy: 0.9912
Epoch 8/10
24/24 [==============================] - 26s 1s/step - loss: 0.0246 - accuracy: 0.9926
Epoch 9/10
24/24 [==============================] - 26s 1s/step - loss: 0.0221 - accuracy: 0.9926
Epoch 10/10
24/24 [==============================] - 26s 1s/step - loss: 0.0175 - accuracy: 0.9936

Out[29]: <keras.callbacks.History at 0x1f2f02f0a30>

In [30]: model.save('aslpng1.h5')

In [ ]:
```

Flowchart



Result



Advantages&disadvantages

1.conversation engine of deaf and dumb is easy to Advantages and Disadvantages

implement and understand.

2. Its operates in real time.

3.It is applications in training and testing.

1. Latency, It takes approximately 8 sec to capture image

2. Highly expensive

3. Each image occupies nearly 50Kb memory

4. Complex algorithms for data processing.

5.in image and video processing includes variant lighting conditions, backgrounds and field of view constraints.

Applications

1.fast processing and immediate results with high accurary

2.easy and simple interface to understand and help the user.

Conclusion

Hand Gesture recognition and voice conversion for dumb and deaf person was successfully executed using image processing. The method takes image as input and gives text and speech as an output. Implementation of this system gives up to 90% accuracy and works successfully in most of the test case.

Future scope

proposed system can be developed and implemented using Raspberry Pi. Image Processing part should be improved so that System would be able to communicate in both directions i.e.it should be capable of converting normal language to sign language and vice versa. We will try to recognize signs which include motion. Moreover we will focus on converting the sequence of gestures into text i.e. word and sentences and then converting it into the speech .

Bibliography

Model building

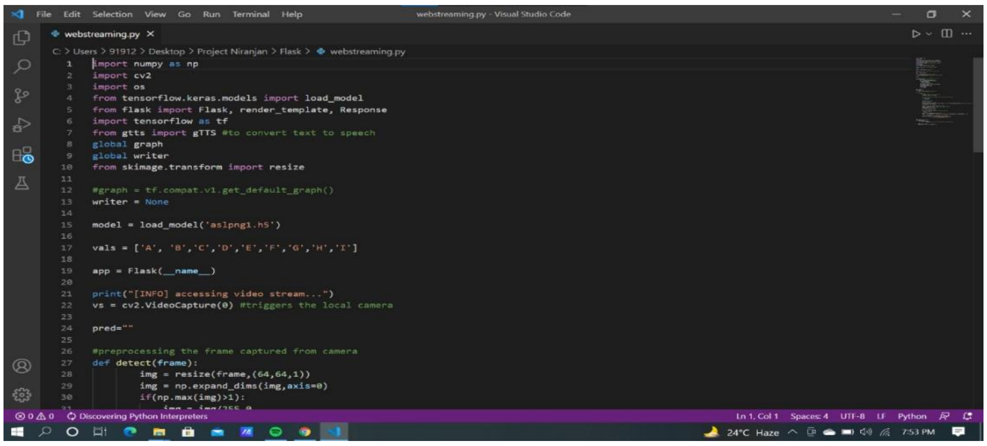
1 Dataset
2 Jupter Notebook

Application Building

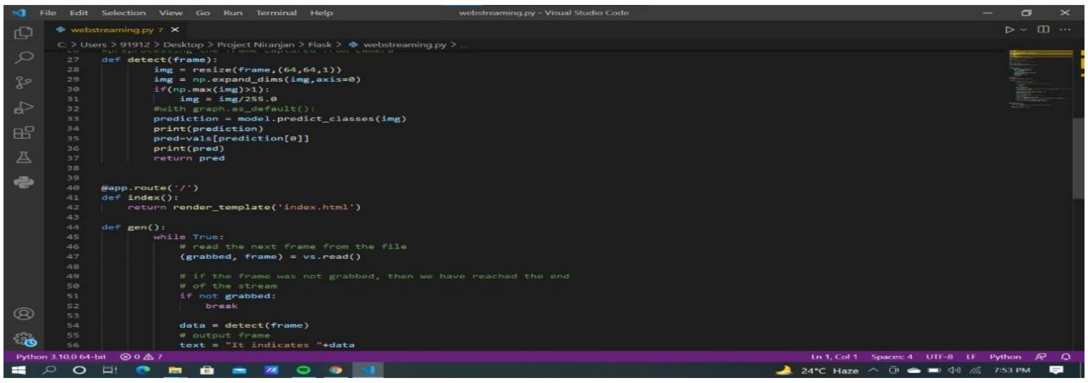
1 Flask app
2 HTML

Appendix

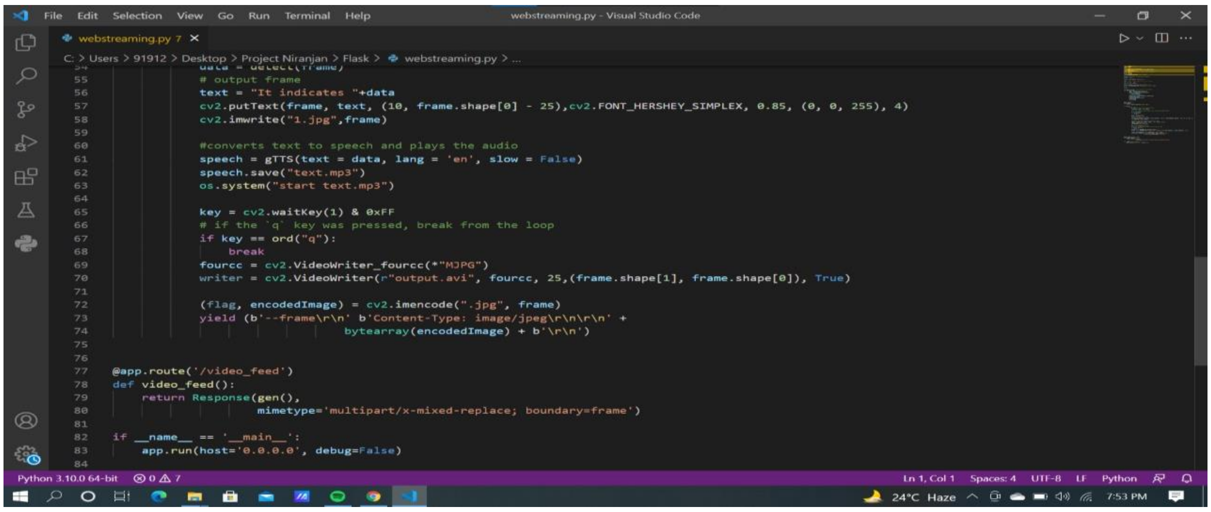
Flask



```python
import numpy as np
import cv2
import os
from tensorflow.keras.models import load_model
from flask import Flask, render_template, Response
import tensorflow as tf
from gtts import gTTS #to convert text to speech
global graph
global writer
from skimage.transform import resize

#graph = tf.compat.v1.get_default_graph()
writer = None

model = load_model('aslpng1.h5')

vals = ['A', 'B','C','D','E','F','G','H','I']

app = Flask(__name__)

print("[INFO] accessing video stream...")
vs = cv2.VideoCapture(0) #triggers the local camera

pred=""

#preprocessing the frame captured from camera
def detect(frame):
        img = resize(frame,(64,64,1))
        img = np.expand_dims(img,axis=0)
        if(np.max(img)>1):
```



```python
def detect(frame):
        img = resize(frame,(64,64,1))
        img = np.expand_dims(img,axis=0)
        if(np.max(img)>1):
            img = img/255.0
        #with graph.as_default():
        prediction = model.predict_classes(img)
        print(prediction)
        pred=vals[prediction[0]]
        print(pred)
        return pred


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

def gen():
        while True:
            # read the next frame from the file
            (grabbed, frame) = vs.read()

            # if the frame was not grabbed, then we have reached the end
            # of the stream
            if not grabbed:
                break

            data = detect(frame)
            # output frame
            text = "It indicates "+data
```



```python
            # output frame
            text = "It indicates "+data
            cv2.putText(frame, text, (10, frame.shape[0] - 25),cv2.FONT_HERSHEY_SIMPLEX, 0.85, (0, 0, 255), 4)
            cv2.imwrite("1.jpg",frame)

            #converts text to speech and plays the audio
            speech = gTTS(text = data, lang = 'en', slow = False)
            speech.save("text.mp3")
            os.system("start text.mp3")

            key = cv2.waitKey(1) & 0xFF
            # if the `q` key was pressed, break from the loop
            if key == ord("q"):
                break
            fourcc = cv2.VideoWriter_fourcc(*"MJPG")
            writer = cv2.VideoWriter(r"output.avi", fourcc, 25,(frame.shape[1], frame.shape[0]), True)

            (flag, encodedImage) = cv2.imencode(".jpg", frame)
            yield (b'--frame\r\n' b'Content-Type: image/jpeg\r\n\r\n' +
                            bytearray(encodedImage) + b'\r\n')


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

if __name__ == '__main__':
    app.run(host='0.0.0.0', debug=False)
```