Capstone Project-5

Face Emotion Recognition

Member

Sourabh Pramanik

Contents

1. **Problem Statement**
2. **Data Summary**
3. **Input data**
4. **Modelling**
5. **Deployment**
6. **Conclusion**
7. **Reference**
8. **Q/A**

Problem Statement

The Project is related to online teaching classroom. Online teaching classrooms may be giving quality content but when it comes to understanding whether a student is understanding or not it’s a challenging problem.

So in this project we will create a Face emotion recognition system to understand the behaviour of students during teaching by watching their facial expressions.

Data Summary

We will complete this project by using following steps-

- At first we will check the count of Test and Train images.
- We will use Transfer learning technique ‘Inceptionv3’.
- Now we will choose the optimizer and will do data augmentation.
- Now we will train the model and will save in h5 file.
- After that we will use the Flask web application we will deploy our model in AWS EC2 platform.

Data Summary

Outcome of this Project -

Different type of facial expression will show the reaction of students and it will help to detect that the understanding of students.

Input Data

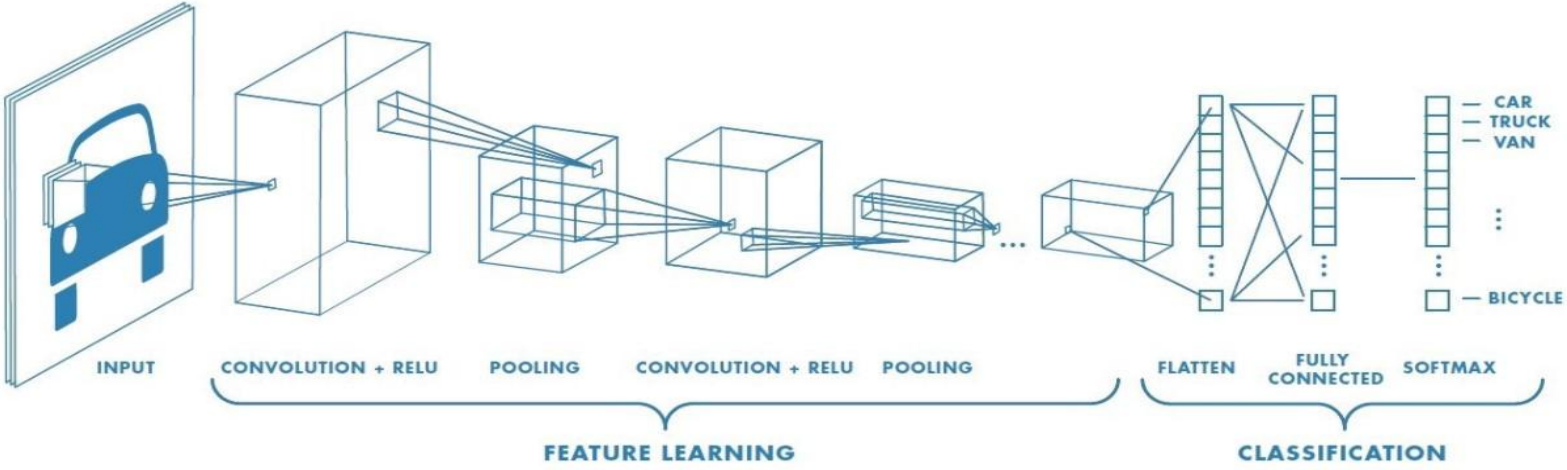
Different types of Facial Expression-

- Angry
- Disgust
- Fear
- Happy
- Neutral
- Sad
- Surprise

Modelling

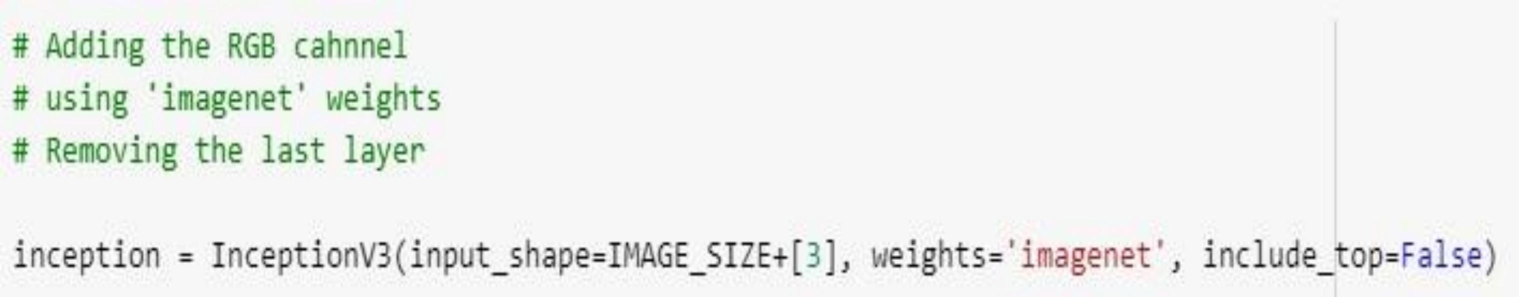
Convolution Neural Network-

A convolutional neural network (CNN) is a type of artificial neural network used in image recognition and processing that is specifically designed to process pixel data



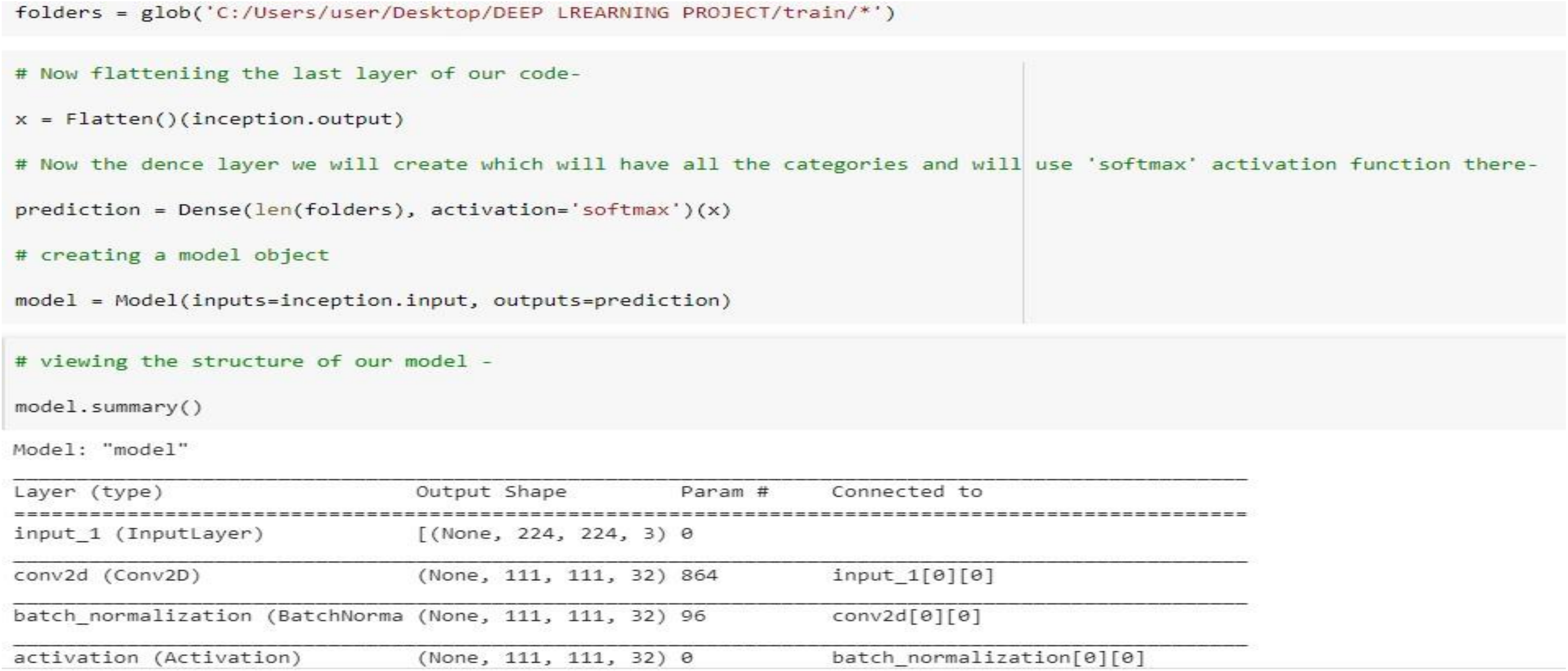
Modelling

Her we will use Inceptionv3 image recognition CNN technique which has been shown to attain greater than 78.1% accuracy in ImageNet dataset.



```
# Adding the RGB cahnnel
# using 'imagenet' weights
# Removing the last layer

inception = InceptionV3(input_shape=IMAGE_SIZE+[3], weights='imagenet', include_top=False)
```

Modelling



```
folders = glob('C:/Users/user/Desktop/DEEP LREARNING PROJECT/train/*')
```

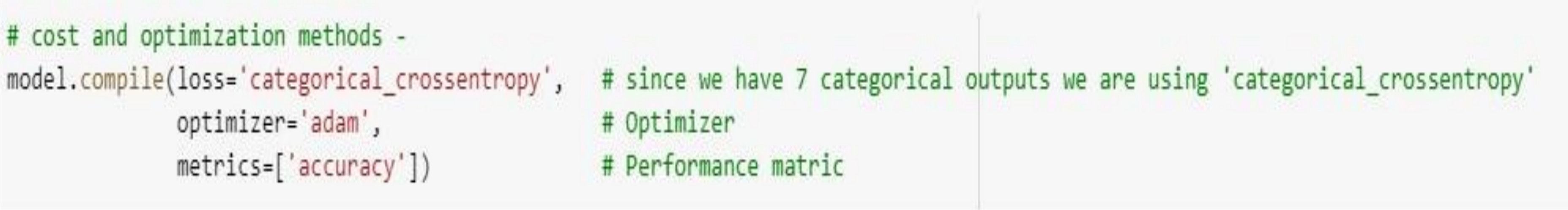
```
# Now flatteniing the last layer of our code-

x = Flatten()(inception.output)

# Now the dence layer we will create which will have all the categories and will use 'softmax' activation function there-

prediction = Dense(len(folders), activation='softmax')(x)

# creating a model object

model = Model(inputs=inception.input, outputs=prediction)
```

```
# viewing the structure of our model -

model.summary()
```

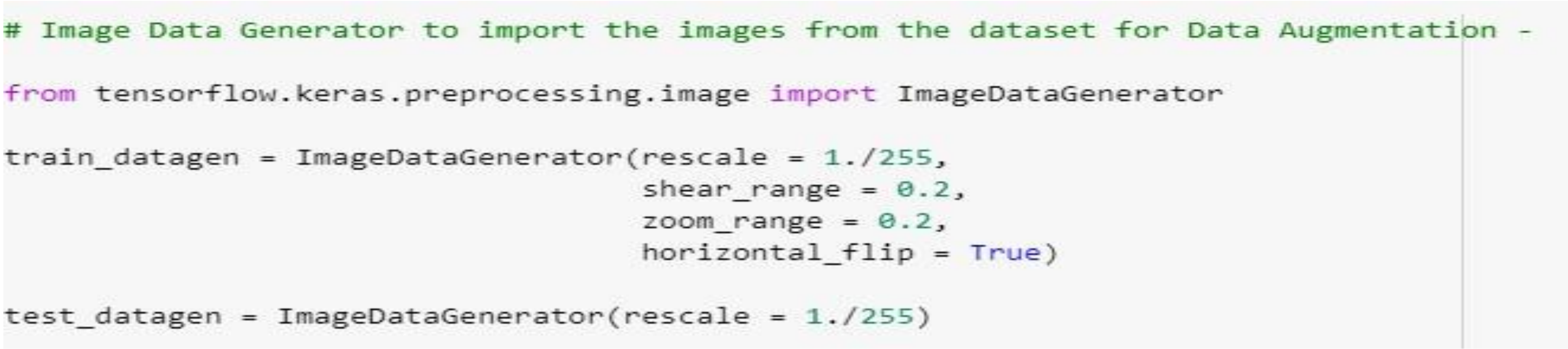
```
Model: "model"
__________________________________________________________________________________________________
Layer (type)                    Output Shape         Param #     Connected to
==================================================================================================
input_1 (InputLayer)            [(None, 224, 224, 3) 0
__________________________________________________________________________________________________
conv2d (Conv2D)                 (None, 111, 111, 32) 864         input_1[0][0]
__________________________________________________________________________________________________
batch_normalization (BatchNorma (None, 111, 111, 32) 96          conv2d[0][0]
__________________________________________________________________________________________________
activation (Activation)         (None, 111, 111, 32) 0           batch_normalization[0][0]
```

Modelling

Now we will set the Optimizer ‘Adam’ and apply Data Augmentation-

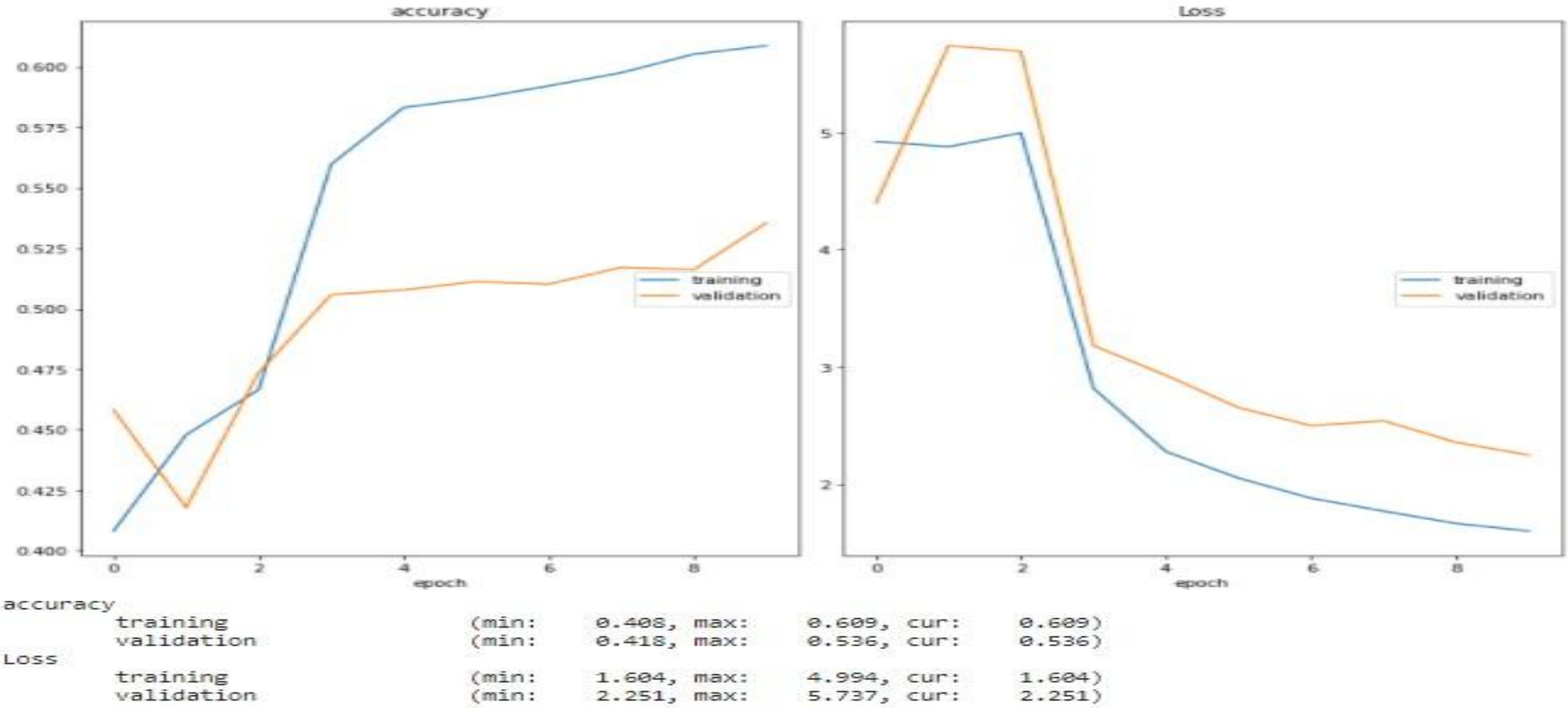


```
# cost and optimization methods -
model.compile(loss='categorical_crossentropy',   # since we have 7 categorical outputs we are using 'categorical_crossentropy'
              optimizer='adam',                  # Optimizer
              metrics=['accuracy'])              # Performance matric
```



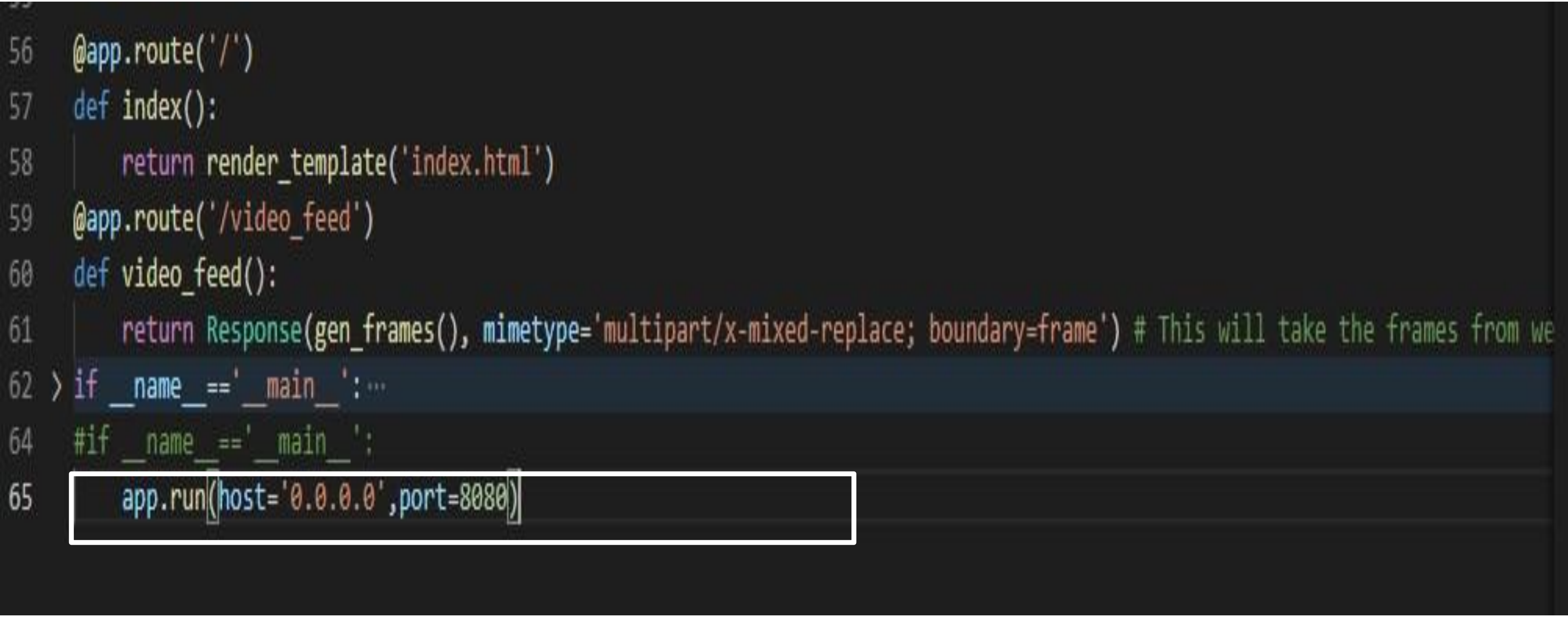
```
# Image Data Generator to import the images from the dataset for Data Augmentation -

from tensorflow.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)
```

Modelling



```
accuracy
        training             (min:    0.408, max:    0.609, cur:    0.609)
        validation           (min:    0.418, max:    0.536, cur:    0.536)
Loss
        training             (min:    1.604, max:    4.994, cur:    1.604)
        validation           (min:    2.251, max:    5.737, cur:    2.251)
```

Deployment

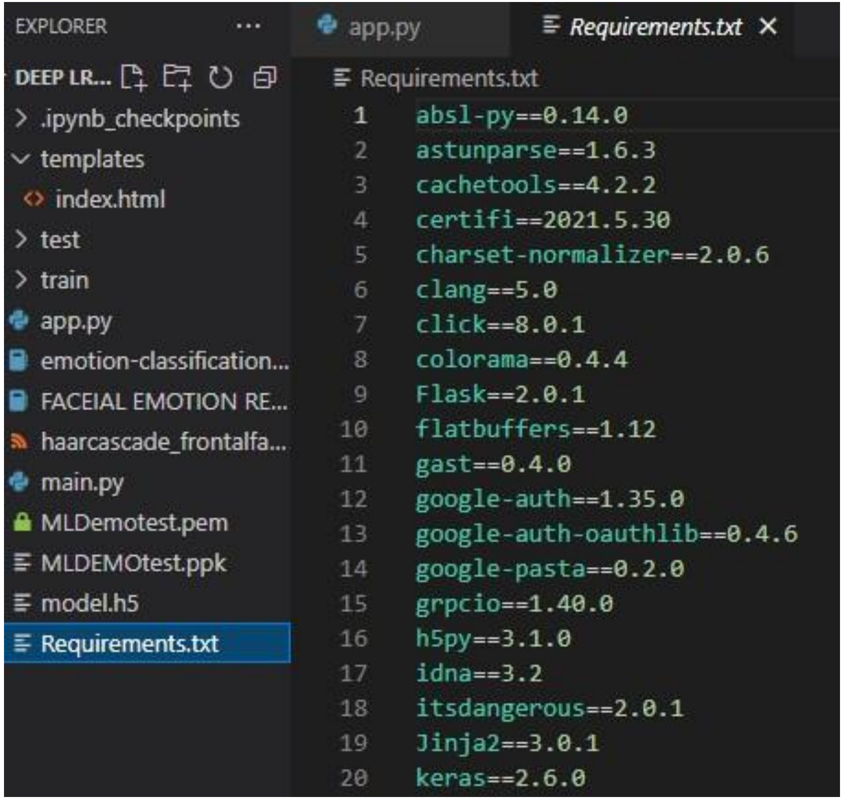


```
@app.route('/')
def index():
    return render_template('index.html')
@app.route('/video_feed')
def video_feed():
    return Response(gen_frames(), mimetype='multipart/x-mixed-replace; boundary=frame') # This will take the frames from we
if __name__=='__main__':⋯
#if __name__=='__main__':
    app.run(host='0.0.0.0',port=8080)
```

Deployment

Requirement.txt

Contains all the Library used.



```
absl-py==0.14.0
astunparse==1.6.3
cachetools==4.2.2
certifi==2021.5.30
charset-normalizer==2.0.6
clang==5.0
click==8.0.1
colorama==0.4.4
Flask==2.0.1
flatbuffers==1.12
gast==0.4.0
google-auth==1.35.0
google-auth-oauthlib==0.4.6
google-pasta==0.2.0
grpcio==1.40.0
h5py==3.1.0
idna==3.2
itsdangerous==2.0.1
Jinja2==3.0.1
keras==2.6.0
```

Deployment

Used AWS EC2 Instance for Model Deployment.

Link -

ec2-3-141-11-173.us-east-2.compute.amazonaws.com:8080

Challenges

High amount of training time.

Partially hidden faces creating problems.

Conclusion

Our model is performing well with using InceptionV3 CNN model.

So now this application can be used during online classrooms.

We can use the same concept for Face detection, Object detection etc.

References

https://towardsdatascience.com/

https://stackoverflow.com/

Q & A