LAB-03

Aim: To build a Convolutional Neural Network and use it to classify faces

```
!unzip /content/Face-Images
```

```
Archive:  /content/Face-Images.zip
   creating: Face Images/
  inflating: Face Images/.DS_Store
   creating: __MACOSX/
   creating: __MACOSX/Face Images/
  inflating: __MACOSX/Face Images/._.DS_Store
   creating: Face Images/Final Training Images/
  inflating: Face Images/Final Training Images/.DS_Store
   creating: __MACOSX/Face Images/Final Training Images/
  inflating: __MACOSX/Face Images/Final Training Images/._.DS_Store
   creating: Face Images/Final Training Images/face12/
  inflating: Face Images/Final Training
Images/face12/image_0281_Face_1.jpg
   creating: __MACOSX/Face Images/Final Training Images/face12/
  inflating: __MACOSX/Face Images/Final Training
Images/face12/._image_0281_Face_1.jpg
  inflating: Face Images/Final Training
Images/face12/image_0284_Face_1.jpg
  inflating: __MACOSX/Face Images/Final Training
Images/face12/._image_0284_Face_1.jpg
  inflating: Face Images/Final Training
Images/face12/image_0283_Face_2.jpg
  inflating: __MACOSX/Face Images/Final Training
Images/face12/._image_0283_Face_2.jpg
  inflating: Face Images/Final Training
Images/face12/image_0279_Face_1.jpg
  inflating: __MACOSX/Face Images/Final Training
Images/face12/._image_0279_Face_1.jpg
  inflating: Face Images/Final Training
Images/face12/image_0286_Face_1.jpg
  inflating: __MACOSX/Face Images/Final Training
Images/face12/._image_0286_Face_1.jpg
  inflating: Face Images/Final Training
Images/face12/image_0277_Face_1.jpg
  inflating: __MACOSX/Face Images/Final Training
Images/face12/._image_0277_Face_1.jpg
  inflating: Face Images/Final Training
Images/face12/image_0285_Face_1.jpg
  inflating: __MACOSX/Face Images/Final Training
Images/face12/._image_0285_Face_1.jpg
  inflating: Face Images/Final Training
```

```
Images/face1/._4face1.jpg
  inflating: Face Images/Final Testing Images/face1/2face1.jpg
  inflating: __MACOSX/Face Images/Final Testing
Images/face1/._2face1.jpg
  inflating: Face Images/Final Testing Images/face1/3face1.jpg
  inflating: __MACOSX/Face Images/Final Testing
Images/face1/._3face1.jpg
  inflating: Face Images/Final Testing Images/face1/1face1.jpg
  inflating: __MACOSX/Face Images/Final Testing
Images/face1/._1face1.jpg
  inflating: __MACOSX/Face Images/Final Testing Images/._face1
  inflating: __MACOSX/Face Images/._Final Testing Images
  inflating: __MACOSX/._Face Images
```

```
from keras.preprocessing.image import ImageDataGenerator
trainpath='/content/Face Images/Final Training Images'
train_datagen=ImageDataGenerator(shear_range=0.1, zoom_range=0.1,
horizontal_flip=True)

test_datagen = ImageDataGenerator()

training_set = train_datagen.flow_from_directory(trainpath,
target_size=(64, 64), batch_size=32, class_mode='categorical')
```

```
Found 244 images belonging to 16 classes.
```

```
test_set = test_datagen.flow_from_directory(trainpath,
target_size=(64, 64), batch_size=32, class_mode='categorical')
```

```
Found 244 images belonging to 16 classes.
```

```
test_set.class_indices
```

```
{'face1': 0,
 'face10': 1,
 'face11': 2,
 'face12': 3,
 'face13': 4,
 'face14': 5,
 'face15': 6,
 'face16': 7,
 'face2': 8,
 'face3': 9,
 'face4': 10,
 'face5': 11,
 'face6': 12,
 'face7': 13,
 'face8': 14,
 'face9': 15}
```

```
TrainClasses=training_set.class_indices
```

```
ResultMap={}
for faceValue,faceName in
zip(TrainClasses.values(),TrainClasses.keys()):
  ResultMap[faceValue]=faceName

import pickle
with open("ResultsMap.pkl", 'wb') as fileWriteStream:
  pickle.dump(ResultMap, fileWriteStream)

print("Mapping of Face and its ID",ResultMap)
```

```
Mapping of Face and its ID {0: 'face1', 1: 'face10', 2: 'face11', 3:
'face12', 4: 'face13', 5: 'face14', 6: 'face15', 7: 'face16', 8:
'face2', 9: 'face3', 10: 'face4', 11: 'face5', 12: 'face6', 13:
'face7', 14: 'face8', 15: 'face9'}
```

```
OutputNeurons=len(ResultMap)
print('\n The Number of output neurons: ', OutputNeurons)
```

```
 The Number of output neurons:  16
```

```
from keras.models import Sequential
from keras.layers import Convolution2D
from keras.layers import MaxPool2D
from keras.layers import Flatten
from keras.layers import Dense

classifier= Sequential()
classifier.add(Convolution2D(32, kernel_size=(5, 5), strides=(1, 1),
input_shape=(64,64,3), activation='relu'))
classifier.add(MaxPool2D(pool_size=(2,2)))
classifier.add(Convolution2D(64, kernel_size=(5, 5), strides=(1, 1),
activation='relu'))
classifier.add(MaxPool2D(pool_size=(2,2)))
classifier.add(Flatten())
classifier.add(Dense(64, activation='relu'))
classifier.add(Dense(OutputNeurons, activation='softmax'))
classifier.compile(loss='categorical_crossentropy', optimizer =
'adam', metrics=["accuracy"])

import time
StartTime=time.time()
classifier.fit( training_set, steps_per_epoch=10, epochs=20,
validation_data=test_set, validation_steps=10)
EndTime=time.time()
```

```
Epoch 1/20
 8/10 [=======================>......] - ETA: 0s - loss: 1.9515 -
accuracy: 0.3975
```

```
WARNING:tensorflow:Your input ran out of data; interrupting training.
Make sure that your dataset or generator can generate at least
`steps_per_epoch * epochs` batches (in this case, 200 batches). You
may need to use the repeat() function when building your dataset.
WARNING:tensorflow:Your input ran out of data; interrupting training.
Make sure that your dataset or generator can generate at least
`steps_per_epoch * epochs` batches (in this case, 10 batches). You may
need to use the repeat() function when building your dataset.

 10/10 [==============================] - 4s 399ms/step - loss: 1.9515
- accuracy: 0.3975 - val_loss: 1.5407 - val_accuracy: 0.5697
```

```
import numpy as np
from keras.preprocessing import image
ImagePath='/content/Face Images/Final Testing
Images/face12/1face12.jpg'
test_image=image.load_img(ImagePath,target_size=(64, 64))
test_image=image.img_to_array(test_image)
test_image=np.expand_dims(test_image,axis=0)
result=classifier.predict(test_image,verbose=0)
print('Prediction is: ',ResultMap[np.argmax(result)])
```

```
Prediction is:  face12
```