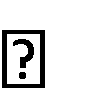
**Arya Teli**

**2213138**

**AIA-1-A**

**Assignment No. 4**

**Aim:** Develop face recognition system using CNN. Create a dataset of minimum 20 students from your class. Check and validate the accuracy of the model.

 Apply dimensionality reduction on input image and plot the change in accuracy of system.

**Objectives:**

1. To learn Data set creation
2. To learn data normalization

**Theory:**

**Dataset creation steps**

* 1. Articulate the problem early.
  2. Establish data collection mechanisms.
  3. Format data to make it consistent.
  4. Reduce data.
  5. Complete data cleaning.
  6. Decompose data.
  7. Rescale data.
  8. Discretize data.

**Image Augmentation**

Image data augmentation is a technique that can be used to artificially expand the size of a training dataset by creating modified versions of images in the dataset. Image data augmentation is used to expand the training dataset in order to improve the performance and ability of the model to generalize.

The intent is to expand the training dataset with new, plausible examples. This means, variations of the training set image that are likely to be seen by the model. For example, a horizontal flip of a picture of a cat may make sense, because the photo could have been taken from the left or right. A vertical flip of the photo of a cat does not make sense and would probably not be appropriate given that the model is very unlikely to see a photo of an upside-down cat.

**Libraries for image augmentation**

There are a lot of image augmentations packages

* skimage
* opencv
* imgaug
* Albumentations
* Augmentor
* Keras(ImageDataGenerator class)

**Fit\_generator, validate\_generator, predict\_generator fit\_generator**

fit\_generator( generator, steps\_per\_epoch=None, epochs=1, verbose=1, callbacks=None, validation\_data=None, validation\_steps=None, validation\_freq=1, class\_weight=None, max\_queue\_size=10, workers=1, use\_multiprocessing=False, shuffle=True, initial\_epoch=0

)

Fits the model on data yielded batch-by-batch by a Python generator.

# predict\_generator

predict\_generator(

generator, steps=None, callbacks=None, max\_queue\_size=10, workers=1, use\_multiprocessing=False, verbose=0

)

Generates predictions for the input samples from a data generator.

# evaluate\_generator

evaluate\_generator(

generator, steps=None, callbacks=None, max\_queue\_size=10, workers=1, use\_multiprocessing=False, verbose=0

)

Evaluates the model on a data generator.

**Code:**

**MTCNN**

import matplotlib.pyplot as plt from matplotlib.patches import Rectangle from matplotlib.patches import Circle from mtcnn.mtcnn import MTCNN from PIL import Image from numpy import asarray def draw\_faces(filename, result\_list):

data = plt.imread(filename) for i in range(len(result\_list)): # get coordinates x1, y1, width, height = result\_list[i]['box'] x2, y2 = x1 + width, y1 + height if x1 <0: x1=0 if x2 <0: x2=0 if y1 <0: y1=0 if y2 <0: y2=0 plt.subplot(1, len(result\_list), i+1) plt.axis('off') plt.imshow(data[y1:y2, x1:x2]) cv2.imwrite(filename,data[y1:y2, x1:x2]) plt.show()

import glob import cv2

path = glob.glob("FaceRecog/\*.jpg") cv\_img = [] for img in path: filename = img image = Image.open(filename) image = image.convert('RGB') pixels = asarray(image) detector = MTCNN() faces = detector.detect\_faces(pixels)

draw\_faces(filename, faces)

**MODEL** import pandas as pd import tensorflow as tf from tensorflow.keras import models,Sequential,layers,preprocessing import keras import os

import mtcnn

file\_names=os.listdir("FaceRecog") NameArray=[] for name in file\_names:

category=name.split('.')[0] if category=='gourishankar': NameArray.append('Gourishankar') elif category=='Aditya\_Panchwagh': NameArray.append("Aditya") elif category=="Dhananjay\_Jha": NameArray.append("Dhananjay") elif category=='Habil\_Bhagat': NameArray.append("Habil") elif category=="Karan\_Mahajan": NameArray.append("Karan") elif category=='Kartik\_Jawanjal': NameArray.append("Kartik") elif category=="Krish\_Shah": NameArray.append("Krish") elif category=='Manas\_Oswal': NameArray.append("Manas") elif category=="Mayank\_Modi": NameArray.append("Mayank") elif category=='Shubham\_Pagare': NameArray.append("Shubham") elif category=="Vishal\_Kasa":

NameArray.append("Kasa")

train=pd.DataFrame({ 'filename':file\_names,

'category':NameArray

}) from sklearn.model\_selection import train\_test\_split from keras.preprocessing.image import ImageDataGenerator,load\_img train\_df,validate\_df = train\_test\_split(train,test\_size=0.2, random\_state=0) train\_df = train\_df.reset\_index(drop=True) validate\_df = validate\_df.reset\_index(drop=True) training = preprocessing.image.ImageDataGenerator(rotation\_range=5, rescale=1. /255, shear\_range=0.1, zoom\_range=0.2, horizontal\_flip=True, width\_shift\_range

=0.1, height\_shift\_range=0.1)

trainingdata = training.flow\_from\_dataframe(train\_df,"FaceRecog",x\_col='filena me',y\_col='category',target\_size=(224,224),class\_mode='categorical') validation = ImageDataGenerator(rotation\_range=5, rescale=1./255, shear\_range= 0.1, zoom\_range=0.2, horizontal\_flip=True, width\_shift\_range=0.1, height\_shift \_range=0.1) validationdata = validation.flow\_from\_dataframe(validate\_df,"FaceRecog", x\_col

='filename',y\_col='category',target\_size=(224,224),class\_mode='categorical')

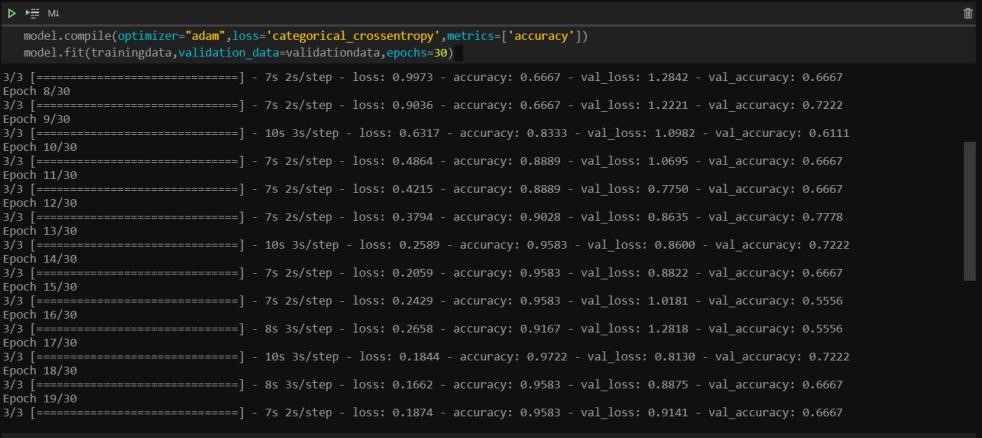
from tensorflow.keras.applications.vgg16 import VGG16 base = VGG16(weights='imagenet',include\_top=False,input\_shape=(224,224,3)) base.trainable = False model = models.Sequential() model.add(base) model.add(layers.Flatten()) model.add(layers.Dense(400, activation='relu')) model.add(layers.Dense(10, activation='softmax'))

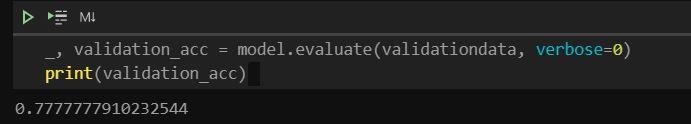
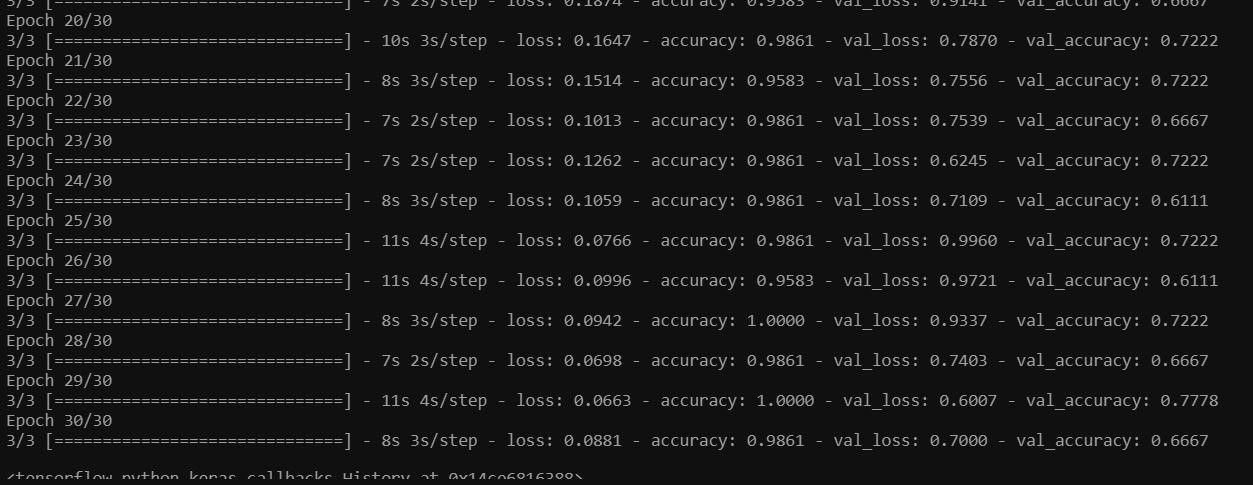
model.compile(optimizer="adam",loss='categorical\_crossentropy',metrics=['accur acy'])

model.fit(trainingdata,validation\_data=validationdata,epochs=30)

\_, validation\_acc = model.evaluate(validationdata, verbose=0) print(validation\_acc)

**Results:**





**Conclusion:**

Thus, we have understood how to create Face recognition system is used and how it is programmed in TensorFlow.

Plus we have also learned to use various face detection algorithms