Graphical user interface, text, application

Description automatically generated

#but first lets import a package called drive and assert google drive in this notebook

#google colab library

#drive model

from google.colab import drive

#mounting to google drive

drive.mount ('/content/drive')



# necessary libraries

#tensorflow will be used

import tensorflow as tf

# tensforflow

from tensorflow import keras

# image data generator from tensorflow

from tensorflow.keras.preprocessing.image import ImageDataGenerator

#establishing variables for assigning training and validating datasets paths

#training dataset path

training\_dataset\_path = '/content/drive/MyDrive/sentiment analyser (classifier)/datasets/train'

#validating dataset path

validating\_dataset\_path = '/content/drive/MyDrive/sentiment analyser (classifier)/datasets/validation'

#generating set of images through ImageDataGenerator

#flow from directory has very good use as it allows to just flow through the diretory

training\_set=ImageDataGenerator(rescale=1./255)\

             .flow\_from\_directory(

                 #setting  for training set

                 directory=training\_dataset\_path,

                 #target image size (resize)

                 target\_size=(48,48),

                 #number of classes

                 batch\_size=50,

                 class\_mode="categorical",

                 #mode of color

                 color\_mode= "grayscale"

             )

validating\_set=ImageDataGenerator(rescale=1./255)\

             .flow\_from\_directory(

                 #setting  for validating set

                 directory=validating\_dataset\_path,

                  #target image size (resize)

                 target\_size=(48,48),

                  #number of classes

                 batch\_size=50,

                 class\_mode="categorical",

                 #mode of color

                 color\_mode="grayscale"

             )

Text

Description automatically generated

Graphical user interface

Description automatically generated with low confidence

#generating a batch of image with corresponding labels

images, labels=next(training\_set)

#importing necessary libraries

import matplotlib.pyplot as plt

Text

Description automatically generated#printing labels for each image , labels are classes or categories

print(labels)

A collage of a person

Description automatically generated with medium confidence

A collage of a person

Description automatically generated with medium confidence

Table

Description automatically generated

Graphical user interface, text, application

Description automatically generated

#creating sequential model structure

#creating sequential model instance

sentiment\_analysis\_model=Sequential()

#first block

#creating first hidden layer which is convulational 2 dimension layer with 32 and input shape of 48 48 1 , 1 is grayscale

#activation funtion is relu

#first convulational layer

sentiment\_analysis\_model.add(Conv2D(32, kernel\_size=(3,3), activation='relu', input\_shape=(48,48,1)))

#second hidden layer

sentiment\_analysis\_model.add(Conv2D(64, kernel\_size=(3,3), activation='relu'))

sentiment\_analysis\_model.add(MaxPooling2D(pool\_size=(2,2)))

#fourth hidden layer

sentiment\_analysis\_model.add(Dropout(0.25))

#second block

#fifth hidden layer

sentiment\_analysis\_model.add(Conv2D(64, kernel\_size=(3,3), activation='relu'))

#sixth hidden layer

sentiment\_analysis\_model.add(MaxPooling2D(pool\_size=(2,2)))

#seventh hidden layer

sentiment\_analysis\_model.add(Conv2D(64, kernel\_size=(3,3), activation='relu'))

#eigth hidden layer

sentiment\_analysis\_model.add(MaxPooling2D(pool\_size=(2,2)))

#ninth hidden layer

sentiment\_analysis\_model.add(Dropout(0.25))

#tenth hidden layer

sentiment\_analysis\_model.add(Flatten())

#eeleventh hidden layer

sentiment\_analysis\_model.add(Dense(1024,activation='relu'))

#twelfth hidden layer

sentiment\_analysis\_model.add(Dropout(0.25))

#output layer

sentiment\_analysis\_model.add(Dense(7,activation='softmax'))

#obtaining model structure

sentiment\_analysis\_model.summary()

Table

Description automatically generated

Table

Description automatically generated

#compiling optimizers and loss function to the model created

Text

Description automatically generated

Graphical user interface, text, application

Description automatically generated with medium confidence

sentiment\_analysis\_model\_info=sentiment\_analysis\_model.fit(

    #assigning training set of images

    x=training\_set,

    #assigning length of tranining set as steps per epoch

    steps\_per\_epoch=len(training\_set),

    #assigning 50 epochs

    epochs=50,

    #assigning validating set of images

    validation\_data=validating\_set,

    #assigning length of validating set as steps per epoch

    validation\_steps=len(validating\_set),

)

Text

Description automatically generated

Text, table

Description automatically generated with medium confidence

Text

Description automatically generated

A screenshot of a computer

Description automatically generated with medium confidence

#preparing test sets

#establishing variables for assigning testing dataset path

#testing dataset path

testing\_dataset\_path = '/content/drive/MyDrive/sentiment analyser (classifier)/datasets/test'

#generating set of images through ImageDataGenerator

#flow from directory has very good use as it allows to just flow through the diretory

testing\_set=ImageDataGenerator(rescale=1./255)\

             .flow\_from\_directory(

                 #assigning testing dataset directory

                 directory=testing\_dataset\_path,

                 #resizing image for target

                 target\_size=(48,48),

                 #assigning classes

                 classes=['angry','disgust','fear','happy','neutral','sad','surprise'],

                 #assigning batch size

                 batch\_size=50,

                 #assigning class code categorical

                 class\_mode="categorical",

                 #assigning color mode grayscale

                 color\_mode="grayscale",

                 #setting shuffle to fakse

                 shuffle=False

             )



Logo

Description automatically generated

#assigning testing iamges and labels to next batch of testing set

testing\_images, test\_labels = next(testing\_set)

#calling plot images function previously defined

plotImages(testing\_images)

#printing test labels which are corresponding classes for images

print(test\_labels)

A collage of a person

Description automatically generated with medium confidence

A collage of a person's face

Description automatically generated

A collage of a person's face

Description automatically generated with medium confidence

Graphical user interface

Description automatically generated with low confidence

#generating unshuffled true lables for test images

testing\_set.classes

Text

Description automatically generated with medium confidence

#predict function for testing purpose

#predictions variable

#testing sets

#no output

predictions = sentiment\_analysis\_model.predict(x=testing\_set, verbose=0)

#importing numpy python library as np

import numpy as np

#rounding every prediction to one value

np.round(predictions)

Table

Description automatically generated

#importing confusion matrix class from sklearn matrics

from sklearn.metrics import confusion\_matrix

#creation of confusion matrix

cm = confusion\_matrix(y\_true=testing\_set.classes, y\_pred=np.argmax(predictions, axis=-1))

#generating testing classes and corresponding indices

testing\_set.class\_indices

Text

Description automatically generated

#assigning plot labels in correct order

plot\_labels = ['angry','disgust','fear','happy','neutral','sad','surprise']

#plotting

plot\_confusion\_matrix(cm=cm, classes=plot\_labels, title='Confusion Matrix')

Chart, scatter chart

Description automatically generated

Logo, company name

Description automatically generated

import json

#saving model structure in json file

model\_json=json.loads(sentiment\_analysis\_model)

with open("/content/drive/MyDrive/sentiment analyser (classifier)/models/sentiment\_analysis\_model.json", "w") as json\_file:

    json\_file.write(str(model\_json))

print("Your file is saved")

