EMOJIFY

TRAINING CODE

```
from re import T
import keras
from keras.preprocessing.image import ImageDataGenerator
from keras.models import Sequential
from keras.layers import Dense,Dropout,Activation,Flatten,BatchNormalization
from keras.layers import Conv2D,MaxPooling2D

# from pandas import Categorical

num_classes=5
img_rows,img_cols=48,48
batch_size=8

train_data_dir=r"C:\Users\satya\OneDrive\Documents\codechef_contest\questions\face-expression-recognition-dataset\face-expression-recognition-dataset\train"
validation_data_dir=r"C:\Users\satya\OneDrive\Documents\codechef_contest\questions\face-expression-recognition-dataset\face-expression-recognition-dataset\validation"

train_datagen=ImageDataGenerator(rescale=1.0/255,rotation_range=30,zoom_range=0.3,horizontal_flip=True,vertical_flip=True)
validation_datagen=ImageDataGenerator(rescale=1.0/255)

train_data=train_datagen.flow_from_directory(train_data_dir,color_mode="grayscale",target_size=(48,48),batch_size=batch_size,class_mode='categorical',shuffle=True)
validation_data=validation_datagen.flow_from_directory(validation_data_dir,color_mode="grayscale",target_size=(48,48),batch_size=batch_size,class_mode='categorical',shuffle=True)



# testing


model = Sequential()

# Block-1

model.add(Conv2D(32,(3,3),padding='same',kernel_initializer='he_normal',input_shape=(img_rows,img_cols,1)))
model.add(Activation('elu'))
model.add(BatchNormalization())
model.add(Conv2D(32,(3,3),padding='same',kernel_initializer='he_normal',input_shape=(img_rows,img_cols,1)))
model.add(Activation('elu'))
model.add(BatchNormalization())
model.add(MaxPooling2D(pool_size=(2,2)))
model.add(Dropout(0.2))

# Block-2
```

```
model.add(Conv2D(64,(3,3),padding='same',kernel_initializer='he_normal'))
model.add(Activation('elu'))
model.add(BatchNormalization())
model.add(Conv2D(64,(3,3),padding='same',kernel_initializer='he_normal'))
model.add(Activation('elu'))
model.add(BatchNormalization())
model.add(MaxPooling2D(pool_size=(2,2)))
model.add(Dropout(0.2))

# Block-3

model.add(Conv2D(128,(3,3),padding='same',kernel_initializer='he_normal'))
model.add(Activation('elu'))
model.add(BatchNormalization())
model.add(Conv2D(128,(3,3),padding='same',kernel_initializer='he_normal'))
model.add(Activation('elu'))
model.add(BatchNormalization())
model.add(MaxPooling2D(pool_size=(2,2)))
model.add(Dropout(0.2))

# Block-4

model.add(Conv2D(256,(3,3),padding='same',kernel_initializer='he_normal'))
model.add(Activation('elu'))
model.add(BatchNormalization())
model.add(Conv2D(256,(3,3),padding='same',kernel_initializer='he_normal'))
model.add(Activation('elu'))
model.add(BatchNormalization())
model.add(MaxPooling2D(pool_size=(2,2)))
model.add(Dropout(0.2))

# Block-5

model.add(Flatten())
model.add(Dense(64,kernel_initializer='he_normal'))
model.add(Activation('elu'))
model.add(BatchNormalization())
model.add(Dropout(0.5))

# Block-6

model.add(Dense(64,kernel_initializer='he_normal'))
model.add(Activation('elu'))
model.add(BatchNormalization())
model.add(Dropout(0.5))

# Block-7

model.add(Dense(num_classes,kernel_initializer='he_normal'))
model.add(Activation('softmax'))

print(model.summary())
```

```
from keras.optimizers import rmsprop_v2,sgd_experimental,adam_v2
from keras.callbacks import ModelCheckpoint, EarlyStopping, ReduceLROnPlateau

checkpoint = ModelCheckpoint('Emotion_little_vgg.h5',
                             monitor='val_loss',
                             mode='min',
                             save_best_only=True,
                             verbose=1)

earlystop = EarlyStopping(monitor='val_loss',
                          min_delta=0,
                          patience=3,
                          verbose=1,
                          restore_best_weights=True
                          )

reduce_lr = ReduceLROnPlateau(monitor='val_loss',
                              factor=0.2,
                              patience=3,
                              verbose=1,
                              min_delta=0.0001)

callbacks = [earlystop,checkpoint,reduce_lr]

model.compile(loss='categorical_crossentropy',
              metrics=['accuracy'])


nb_train_samples = 24176
nb_validation_samples = 3006
epochs=25

history=model.fit_generator(
                train_data,
                steps_per_epoch=nb_train_samples//batch_size,
                epochs=epochs,
                callbacks=callbacks,
                validation_data=validation_data,
                validation_steps=nb_validation_samples//batch_size)
```

TEST CODE

```
from keras.models import load_model
from time import sleep
from keras.preprocessing.image import img_to_array
from keras.preprocessing import image
import cv2
import numpy as np

output= {
    'Happy':r"C:\Users\satya\OneDrive\Documents\codechef_contest\questions\face-
expression-recognition-dataset\face-expression-recognition-dataset\OutPut
empji\happy.jpg",
    'Angry':r"C:\Users\satya\OneDrive\Documents\codechef_contest\questions\face-
expression-recognition-dataset\face-expression-recognition-dataset\OutPut
empji\angry.jpg",
    'Sad':r"C:\Users\satya\OneDrive\Documents\codechef_contest\questions\face-expression-
recognition-dataset\face-expression-recognition-dataset\OutPut empji\sad.jpg",
    'Neutral':r"C:\Users\satya\OneDrive\Documents\codechef_contest\questions\face-
expression-recognition-dataset\face-expression-recognition-dataset\OutPut empji\sad.jpg",
    'Surprise':r"C:\Users\satya\OneDrive\Documents\codechef_contest\questions\face-
expression-recognition-dataset\face-expression-recognition-dataset\OutPut empji\sad.jpg",
    # 'Sad':r"C:\Users\satya\OneDrive\Documents\codechef_contest\questions\face-
expression-recognition-dataset\face-expression-recognition-dataset\OutPut empji\sad.jpg",
    # 'Sad':r"C:\Users\satya\OneDrive\Documents\codechef_contest\questions\face-
expression-recognition-dataset\face-expression-recognition-dataset\OutPut empji\sad.jpg"
}


face_classifier =
cv2.CascadeClassifier(r'C:\Users\satya\OneDrive\Documents\codechef_contest\questions\haarc
ascade_frontalface_default.xml')
classifier
=load_model(r'C:\Users\satya\OneDrive\Documents\codechef_contest\questions\Emotion_little_
vgg.h5')

class_labels = ['Angry','Happy','Neutral','Sad','Surprise']

cap = cv2.VideoCapture(0)



while True:
    # Grab a single frame of video
    ret, frame = cap.read()
    labels = []
    gray = cv2.cvtColor(frame,cv2.COLOR_BGR2GRAY)
    faces = face_classifier.detectMultiScale(gray,1.3,5)

    for (x,y,w,h) in faces:
        cv2.rectangle(frame,(x,y),(x+w,y+h),(255,0,0),2)
        roi_gray = gray[y:y+h,x:x+w]
        roi_gray = cv2.resize(roi_gray,(48,48),interpolation=cv2.INTER_AREA)
    # rect,face,image = face_detector(frame)
```

```
        if np.sum([roi_gray])!=0:
            roi = roi_gray.astype('float')/255.0
            roi = img_to_array(roi)
            roi = np.expand_dims(roi,axis=0)

        # make a prediction on the ROI, then lookup the class

            preds = classifier.predict(roi)[0]
            label=class_labels[preds.argmax()]
            label_position = (x,y)
            cv2.putText(frame,label,label_position,cv2.FONT_HERSHEY_SIMPLEX,2,(0,255,0),3)
        else:
            cv2.putText(frame,'No Face
Found',(640,480),cv2.FONT_HERSHEY_SIMPLEX,2,(0,255,0),3)
    img1=frame
    img2=cv2.resize(cv2.imread(output['Angry']),(20,480))

    # cv2.imshow('Emotion Detector',frame)
    Hori = np.concatenate((img1, img2), axis=1)
    img2=cv2.resize(cv2.imread(output['Angry']),(640,480))
    Verti = np.concatenate((img1, img2), axis=0)

    cv2.imshow('HORIZONTAL', Hori)
    cv2.imshow('VERTICAL', Verti)
    # print(label)

    if cv2.waitKey(1) & 0xFF == ord('q'):
        break

cap.release()
cv2.destroyAllWindows()
```