PROJECT REPORT

SENTIMENT DETECTION USING FACE

STUDENT DETAILS

NAME : Keshav Kumar

ROLL NO- 19001011031

FACULTY DETAILS

MENTOR NAME

(Dr. Komal Kumar Bhatia)

INTRODUCTION

- Human emotion detection is implemented in many areas requiring additional security or information about the person. It can be seen as a second step to face detection where we may be required to set up a second layer of security, where along with the face, the emotion is also detected.

- In this we will add both face expression and audio for the detection of emotions for better results.

- Human emotions can be classified as: fear, contempt, disgust, anger, surprise, sad, happy, and neutral.

OBJECTIVE

- The primary objective of our project is to detect emotions of any person during video call or simple interaction
- Using the face fact that our facial features undergo significant changes with emotions.
- Using speech features include tone, energy, pitch, formant frequency, etc. and identifying emotions through changes in these.
- Emotion detection has a remarkable contribution in various industries to include healthcare, marketing, entertainment, surveillance, retail, e-commerce, HR, and more.

Topic we have studied:

- Data cleaning, preprocessing.
- Machine learning algorithms.
- Neural network model training.
- Opencv for camera use and emotion recognition.

Modules studied:

- Numpy
- Pandas
- Matplotlib
- Seaborn
- Sci- kit learn
- Tensorflow keras
- Opencv
- Librosa

Work done in project till now:

- We have designed deep neural network model for face emotion recognition.
  - In this we he collected data from GitHub and trained model using tensorflow keras module.
- Opencv code done to detect emotions live through camera.
- Audio emotion detection model created.

Model Code Screenshot:

```
[ ] !git clone https://github.com/muxspace/facial_expressions.git
```

```
Cloning into 'facial_expressions'...
remote: Enumerating objects: 14214, done.
remote: Total 14214 (delta 0), reused 0 (delta 0), pack-reused 14214
Receiving objects: 100% (14214/14214), 239.65 MiB | 11.45 MiB/s, done.
Resolving deltas: 100% (223/223), done.
Checking out files: 100% (13996/13996), done.
```

```
import csv
data={}
with open('/content/facial_expressions/data/legend.csv') as f:
  reader=csv.reader(f)
  next(reader)
  for row in reader:
    key=row[2].lower()
    if key in data:
      data[key].append(row[1])
    else:
      data[key]=[row[1]]
```

```
[ ] emotion_list=list(data.keys())
    emotion_list
```

```
import os
os.mkdir('master_data')
os.mkdir('master_data/training')
os.mkdir('master_data/testing')
```

```
[ ] for emotion in emotion_list:
      os.mkdir(os.path.join('master_data/training/',emotion))
      os.mkdir(os.path.join('master_data/testing/',emotion))
```

```
[ ] from shutil import copyfile
    split_size=0.8
    for emotion,images in data.items():
      train_size=int(split_size*len(images))
      train_images=images[:train_size]
      test_images=images[train_size:]
      for image in train_images:
        source=os.path.join('/content/facial_expressions/images',image)
        dest=os.path.join('/content/master_data/training',emotion,image)
        copyfile(source,dest)
      for image in test_images:
        source=os.path.join('/content/facial_expressions/images',image)
        dest=os.path.join('/content/master_data/testing',emotion,image)
        copyfile(source,dest)
```

```
import tensorflow as tf
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras.optimizers import Adam
from tensorflow.keras.callbacks import EarlyStopping
from tensorflow.keras.layers import Conv2D,MaxPooling2D,Flatten,Dense
```

```
[ ] model=tf.keras.models.Sequential([
        Conv2D(16,(3,3),activation='relu',input_shape=(100,100,3)),
        MaxPooling2D(2,2),
        Conv2D(32,(3,3),activation='relu'),
        MaxPooling2D(2,2),
        Conv2D(64,(3,3),activation='relu'),
        MaxPooling2D(2,2),
        Flatten(),
        Dense(512,activation='relu'),
        Dense(8,activation='softmax')

    ])
    model.compile(optimizer=Adam(lr=0.01),loss='categorical_crossentropy',metrics=['accuracy'])
    model.summary()
```

```
train_dir='/content/master_data/training'
test_dir='/content/master_data/testing'

train_datagen=ImageDataGenerator(rescale=1.0/255)
train_generator=train_datagen.flow_from_directory(
                                                  train_dir,
                                                  target_size=(100,100),
                                                  class_mode='categorical',
                                                  batch_size=128


)

test_datagen=ImageDataGenerator(rescale=1.0/255)
test_generator=test_datagen.flow_from_directory(
                                                  test_dir,
                                                  target_size= (100,100),
                                                  class_mode='categorical',
                                                  batch_size=128


)
```

```
Found 10941 images belonging to 8 classes.
Found 2742 images belonging to 8 classes.
```

```
es=EarlyStopping(monitor='val_accuracy',patience=2,min_delta=0.01)
```

```
model.fit_generator(train_generator,
                    epochs=10,
                    verbose=1,
                    validation_data=test_generator,
                    callbacks=[es])
```

```
/usr/local/lib/python3.7/dist-packages/ipykernel_laun
  """
Epoch 1/10
86/86 [==============================] - 96s 1s/step
Epoch 2/10
86/86 [==============================] - ETA: 0s - lo
```

```
[11] model.evaluate(test_generator)
```

```
y_predict=model.predict(test_generator)
```

Opencv code:

```python
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

face_classifier = cv2.CascadeClassifier(r'C:\Users\Admin\OneDrive\Desktop\emotion_detection\haarcascade_frontalface_d
classifier = load_model(r'C:\Users\Admin\OneDrive\Desktop\emotion_detection\facial_expression.h5')

emotion_labels = ['Angry','Disgust','Fear','Happy','Neutral', 'Sad', 'Surprise']

cap = cv2.VideoCapture(0)


while True:
    _, frame = cap.read()
    labels = []
    gray = cv2.cvtColor(frame,cv2.COLOR_BGR2GRAY)
    faces = face_classifier.detectMultiScale(gray)

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



        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)

            prediction = classifier.predict(roi)[0]
            label=emotion_labels[prediction.argmax()]
            label_position = (x,y-10)
```

```python
emotion_labels = ['Angry','Disgust','Fear','Happy','Neutral', 'Sad', 'Surprise']

cap = cv2.VideoCapture(0)


while True:
    _, frame = cap.read()
    labels = []
    gray = cv2.cvtColor(frame,cv2.COLOR_BGR2GRAY)
    faces = face_classifier.detectMultiScale(gray)

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



        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)

            prediction = classifier.predict(roi)[0]
            label=emotion_labels[prediction.argmax()]
            label_position = (x,y-10)
            cv2.putText(frame,label,label_position,cv2.FONT_HERSHEY_SIMPLEX,1,(0,255,0),2)
        else:
            cv2.putText(frame,'No Faces',(30,80),cv2.FONT_HERSHEY_SIMPLEX,1,(0,255,0),2)
    cv2.imshow('Emotion Detector',frame)
    if cv2.waitKey(1) & 0xFF == ord('q'):
        break

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

Audio Emotion Detection Model code:

```
[ ] from google.colab import drive
    drive.mount('/content/drive')

    Mounted at /content/drive

[ ] import os
    Root = "/content/drive/MyDrive/Colab Notebooks"
    os.chdir(Root)

[ ] ls

     facial_expression.ipynb
     modelForPrediction1.sav
    'speech-emotion-recognition-ravdess-data.zip (Unzipped Files)'/

[ ] import librosa
    import soundfile
    import os, glob, pickle
    import numpy as np
```

```
[ ] import librosa
    import soundfile
    import os, glob, pickle
    import numpy as np
    from sklearn.model_selection import train_test_split
    from sklearn.neural_network import MLPClassifier
    from sklearn.metrics import accuracy_score

[ ] #Extract features (mfcc, chroma, mel) from a sound file
    def extract_feature(file_name, mfcc, chroma, mel):
        with soundfile.SoundFile(file_name) as sound_file:
            X = sound_file.read(dtype="float32")
            sample_rate=sound_file.samplerate
            if chroma:
                stft=np.abs(librosa.stft(X))
            result=np.array([])
            if mfcc:
                mfccs=np.mean(librosa.feature.mfcc(y=X, sr=sample_rate, n_mfcc=40).T, axis=0)
                result=np.hstack((result, mfccs))
            if chroma:
                chroma=np.mean(librosa.feature.chroma_stft(S=stft, sr=sample_rate).T,axis=0)
                result=np.hstack((result, chroma))
            if mel:
```

Future work we will add:

- Record voice live and predict emotion.
- Implementation together with the faical expression model.

Thank you