FCI MINI PROJECT REPORT

Submitted By: Surya Bhan Singh
UID: 2017140056
Batch: C
BE IT

GITHUB:https://github.com/suryabhansinghit/FCI

PROBLEM STATEMENT:

Humans interact mostly with speech and body gestures to emphasize certain parts of their speech and display emotions. One way humans display emotions is through facial expressions, which is a crucial aspect of communication. Although nothing is said vocally, but there is much to be understood about these messages we communicate through nonverbal communication. Facial expressions convey nonverbal cues, and they play an important role in interpersonal relations. Automatic recognition of facial expressions could be very useful of natural human machine interfaces. Although we humans recognize facial expressions virtually, but reliable expression recognition by machine is still a challenge. So, we present a Convolutional Neural Networks (CNN) based approach for Realtime emotion recognition. The input will be realtime video and using CNN to predict the facial expression label we should be able to determine one of these emotion: anger, happiness, sadness, surprised and neutral.

WHY I CHOSE THIS TOPIC:

As we can see today that many companies are trying to analyze user data and based on that they show recommendation to users. Example, Youtube recommends you videos based on the videos you watch everyday similarly spotify recommends you new songs based on your song's history. So I thought there can be a system which recommends you certain songs,movies, books or videos based on the current emotion state of the person.

DATASET USED:

The data consists of 48x48 pixel grayscale images of faces. The faces have been automatically registered so that the face is more or less centered and occupies about the same amount of space in each image. The task is to categorize each face based on the emotion shown in the facial expression in to one of seven categories (0=Angry, 1=Disgust, 2=Fear, 3=Happy, 4=Sad, 5=Surprise, 6=Neutral). It contains two columns, "emotion" and "pixels". The "emotion" column contains a numeric code ranging from 0 to 6, inclusive, for the emotion that is present in the image. The "pixels" column contains a string surrounded in quotes for each image. The contents of this string a space-separated pixel values in row major order.

Link:
https://www.kaggle.com/c/challenges-in-representation-learning-facial-expression-recognition-challenge/data

MODEL USED:

Convolutional Neural Network(CNN)

The usage of CNNs are motivated by the fact that they can capture / are able to learn relevant features from an image /video. In terms of performance, CNNs outperform NNs on conventional image recognition tasks and many other tasks. For a completely new task / problem CNNs are very good **feature extractors.** This means that you can extract useful attributes from an already trained CNN with its trained weights by feeding your data on each level and tune the CNN a bit for the specific task.

PROJECT:

```
import sys, os
import pandas as pd
import numpy as np
from keras.models import Sequential
from keras.layers import Dense, Dropout, Activation, Flatten
from keras.layers import Conv2D, MaxPooling2D, BatchNormalization,AveragePooling2D
from keras.losses import categorical_crossentropy
from keras.optimizers import Adam
from keras.regularizers import l2
from keras.utils import np_utils
```

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

```
    Mounted at /content/drive
```

```
df=pd.read_csv('/content/drive/MyDrive/fer2013.csv')
```

```
print(df.info())
```

```
    <class 'pandas.core.frame.DataFrame'>
    RangeIndex: 35887 entries, 0 to 35886
    Data columns (total 3 columns):
     #   Column   Non-Null Count  Dtype 
    ---  ------   --------------  ----- 
     0   emotion  35887 non-null  int64 
     1   pixels   35887 non-null  object
     2   Usage    35887 non-null  object
    dtypes: int64(1), object(2)
    memory usage: 841.2+ KB
    None
```

```
print(df["Usage"].value_counts())
```

```
    Training       28709
    PublicTest      3589
    PrivateTest     3589
    Name: Usage, dtype: int64
```

```
print(df.head())
```

```
       emotion                                             pixels     Usage
    0        0  70 80 82 72 58 58 60 63 54 58 60 48 89 115 121...  Training
    1        0  151 150 147 155 148 133 111 140 170 174 182 15...  Training
    2        2  231 212 156 164 174 138 161 173 182 200 106 38...  Training
    3        4  24 32 36 30 32 23 19 20 30 41 21 22 32 34 21 1...  Training
    4        6  4 0 0 0 0 0 0 0 0 0 0 0 3 15 23 28 48 50 58 84...  Training
```

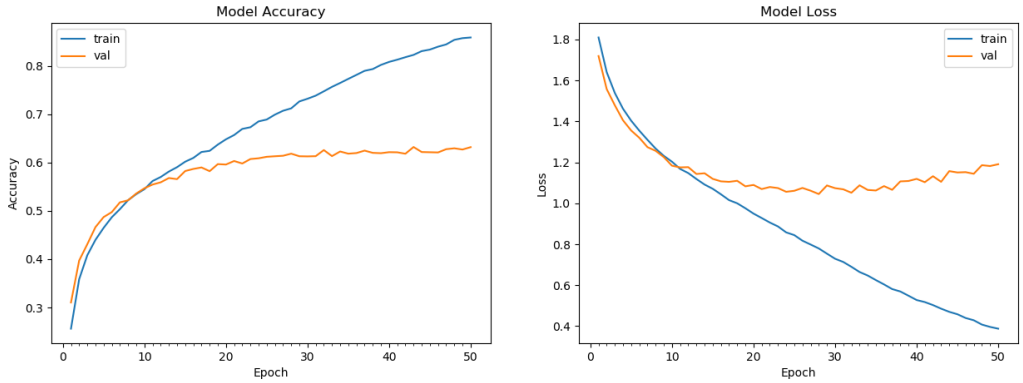
```
X_train,train_y,X_test,test_y=[],[],[],[]
```

```
for index, row in df.iterrows():
    val=row['pixels'].split(" ")
```

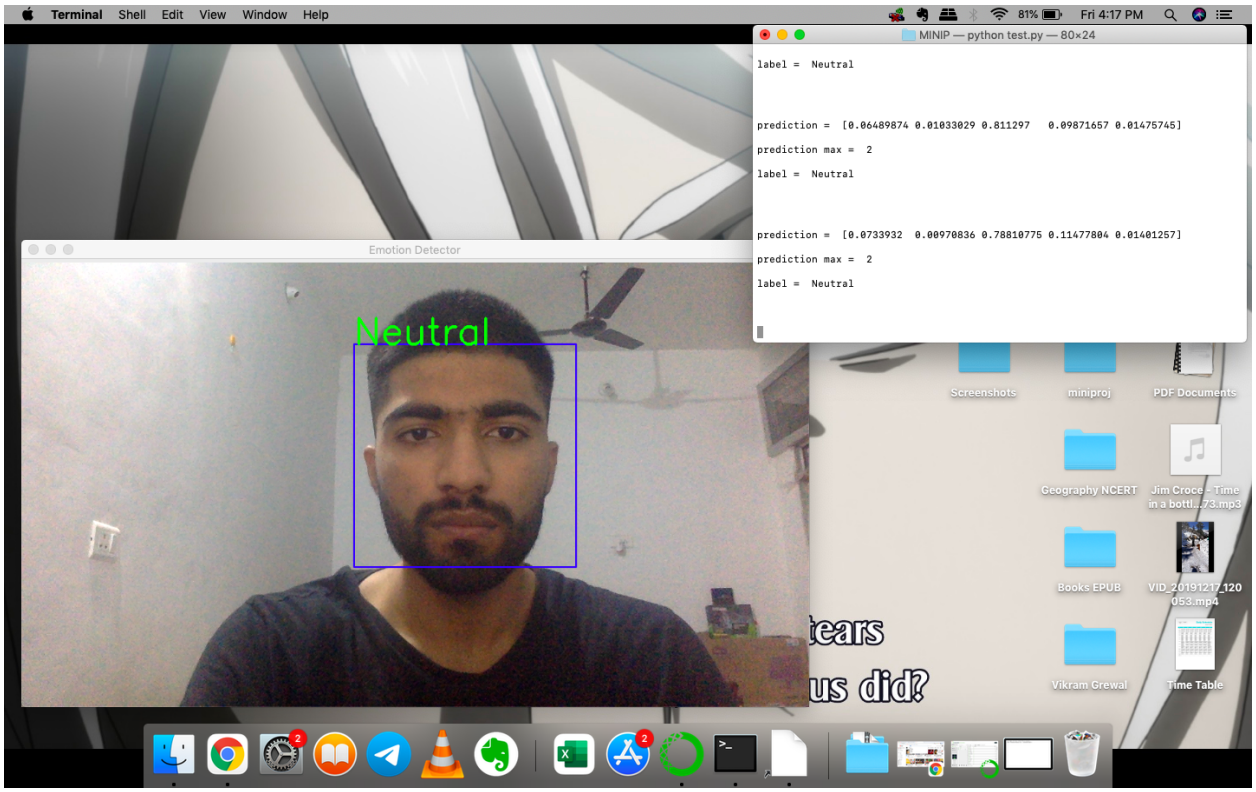
```
        try:
            if 'Training' in row['Usage']:
               X_train.append(np.array(val,'float32'))
               train_y.append(row['emotion'])
            elif 'PublicTest' in row['Usage']:
               X_test.append(np.array(val,'float32'))
               test_y.append(row['emotion'])
        except:
            print(f"error occured at index :{index} and row:{row}")


    num_features = 64
    num_labels = 7
    batch_size = 64
    epochs = 500
    width, height = 48, 48


    X_train = np.array(X_train,'float32')
    train_y = np.array(train_y,'float32')
    X_test = np.array(X_test,'float32')
    test_y = np.array(test_y,'float32')

    train_y=np_utils.to_categorical(train_y, num_classes=num_labels)
    test_y=np_utils.to_categorical(test_y, num_classes=num_labels)


    #normalizing data between 0 and 1
    X_train -= np.mean(X_train, axis=0)
    X_train /= np.std(X_train, axis=0)

    X_test -= np.mean(X_test, axis=0)
    X_test /= np.std(X_test, axis=0)

    X_train = X_train.reshape(X_train.shape[0], 48, 48, 1)

    X_test = X_test.reshape(X_test.shape[0], 48, 48, 1)

    ##designing the cnn
    #1st convolution layer
    model = Sequential()

    model.add(Conv2D(64, kernel_size=(3, 3), activation='relu', input_shape=(X_train.s
    model.add(Conv2D(64,kernel_size= (3, 3), activation='relu'))
    model.add(MaxPooling2D(pool_size=(2,2), strides=(2, 2)))
    model.add(Dropout(0.5))

    #2nd convolution layer
    model.add(Conv2D(64, (3, 3), activation='relu'))
    model.add(Conv2D(64, (3, 3), activation='relu'))

    model.add(MaxPooling2D(pool_size=(2,2), strides=(2, 2)))
    model.add(Dropout(0.5))

    #3rd convolution layer
```

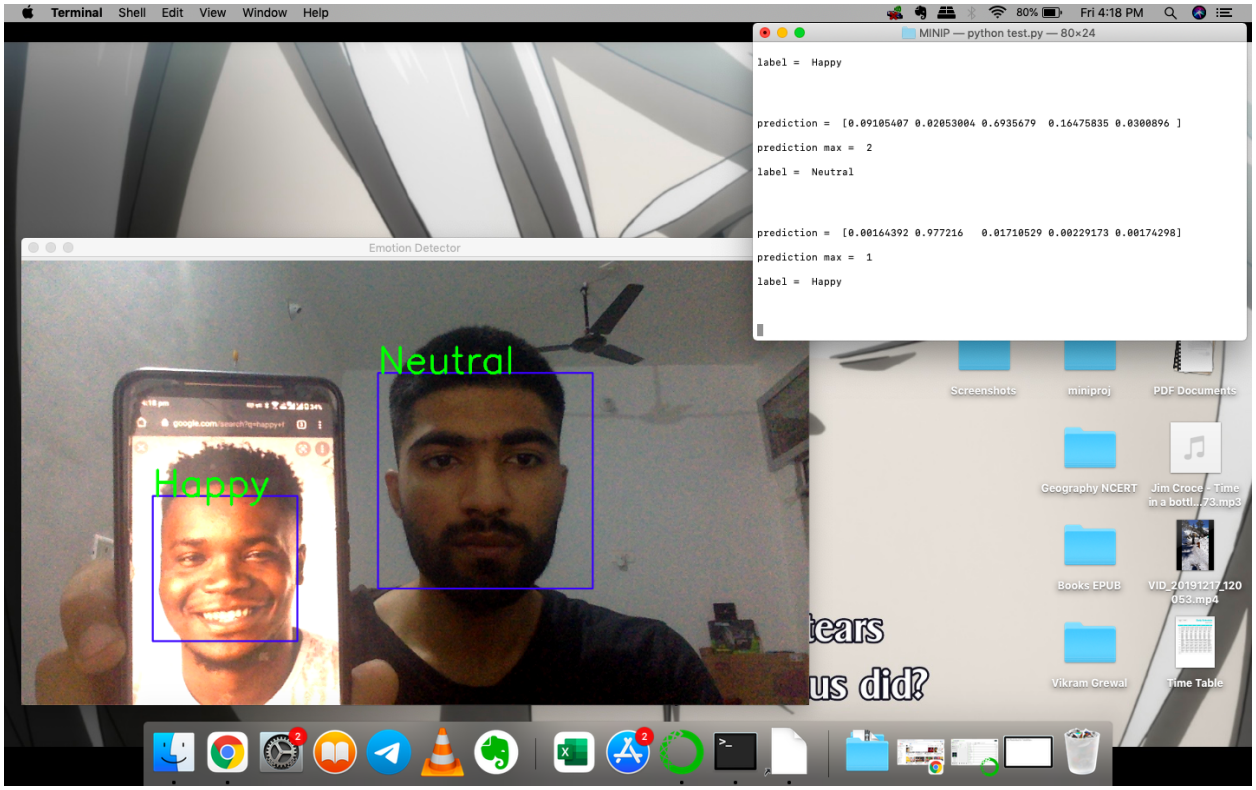
```
model.add(Conv2D(128, (3, 3), activation='relu'))
model.add(Conv2D(128, (3, 3), activation='relu'))
model.add(MaxPooling2D(pool_size=(2,2), strides=(2, 2)))

model.add(Flatten())

#fully connected neural networks
model.add(Dense(1024, activation='relu'))
model.add(Dropout(0.2))
model.add(Dense(1024, activation='relu'))
model.add(Dropout(0.2))

model.add(Dense(num_labels, activation='softmax'))


#Compliling the model
model.compile(loss=categorical_crossentropy,
              optimizer=Adam(),
              metrics=['accuracy'])

#Training the model
model.fit(X_train, train_y,
          batch_size=batch_size,
          epochs=epochs,
          verbose=1,
          validation_data=(X_test, test_y),
          shuffle=True)


#Saving the  model to  use it later on
fer_json = model.to_json()
with open("fer.json", "w") as json_file:
    json_file.write(fer_json)
model.save_weights("fer.h5")
```

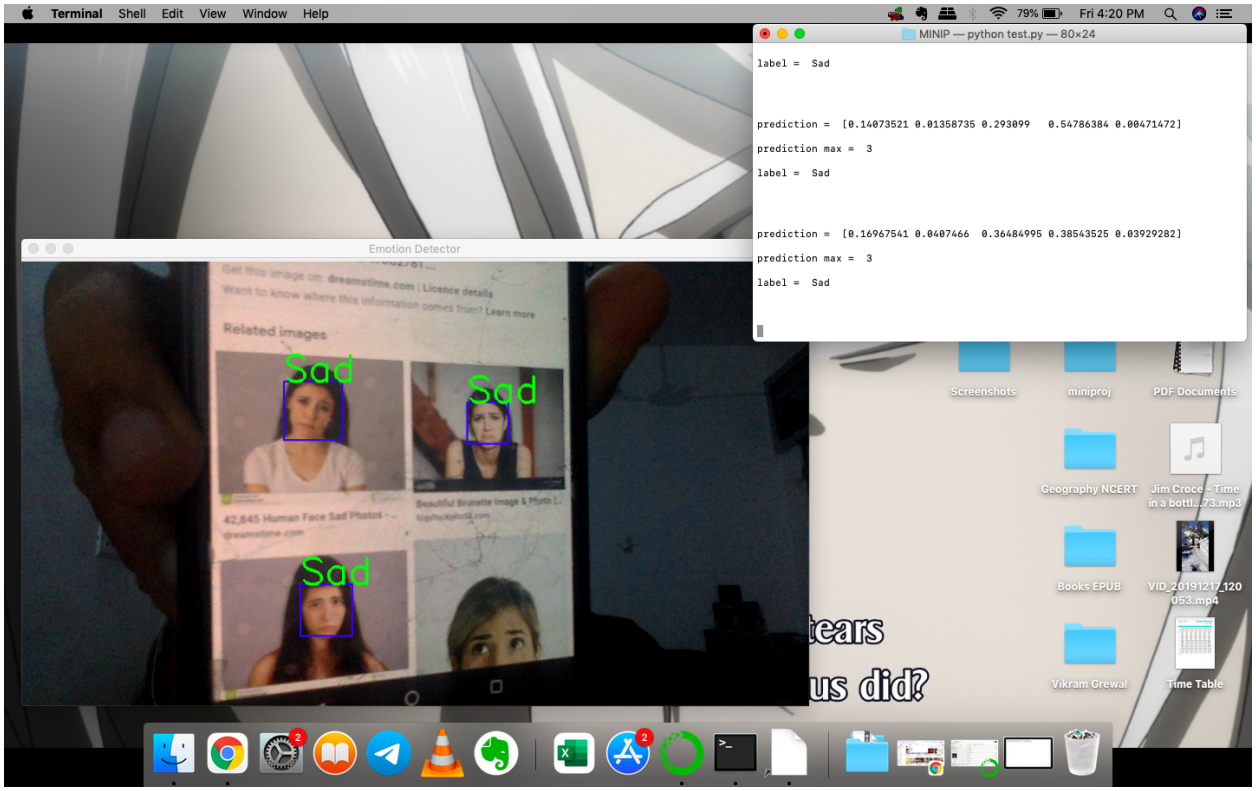
```
Epoch 1/500
```


Model Accuracy
train
val
Accuracy
Epoch
Model Loss
train
val
Loss
Epoch

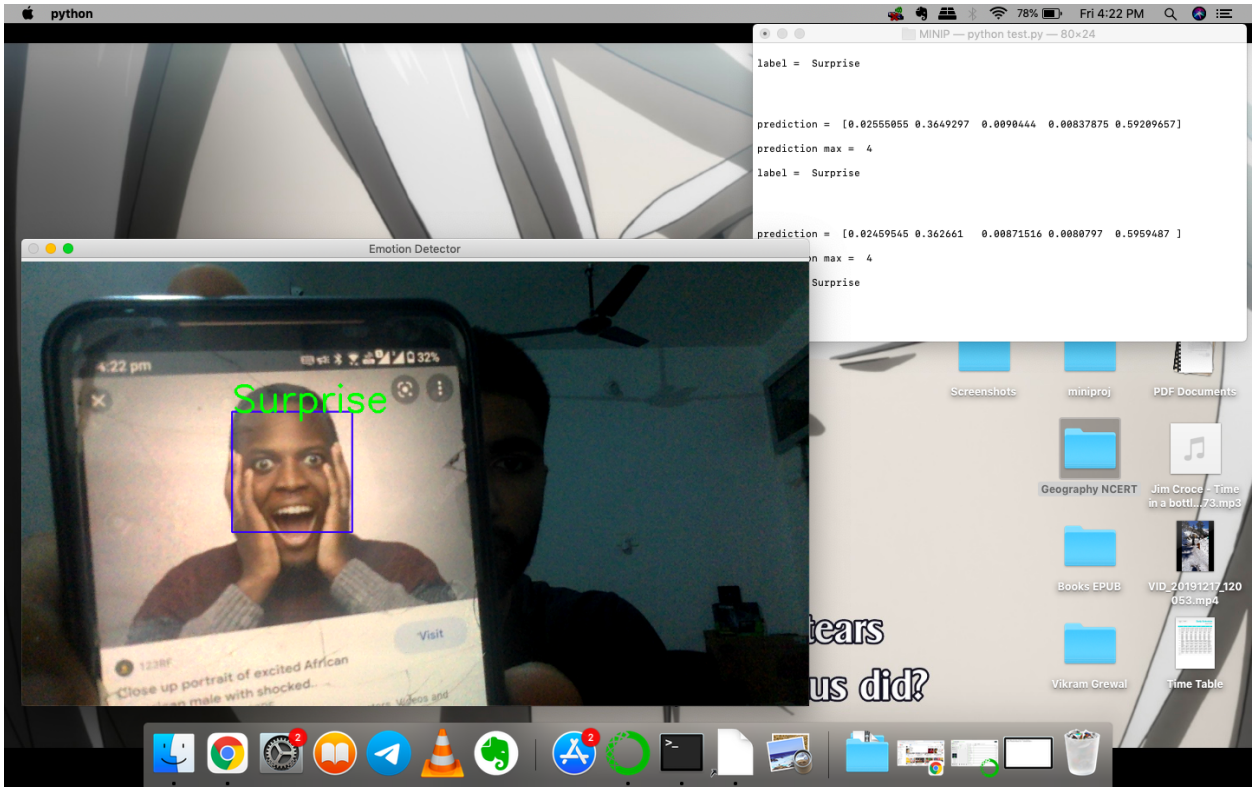
RESULT:


Terminal Shell Edit View Window Help
Fri 4:17 PM
MINIP — python test.py — 80×24
label = Neutral
prediction = [0.06489874 0.01033029 0.811297 0.09871657 0.01475745]
prediction max = 2
label = Neutral
prediction = [0.0733932 0.00970836 0.78810775 0.11477804 0.01401257]
prediction max = 2
label = Neutral
Emotion Detector
Neutral
Screenshots
miniproj
PDF Documents
Geography NCERT
Books EPUB
Vikram Grewal
Time Table


Terminal Shell Edit View Window Help
80% Fri 4:18 PM
MINIP — python test.py — 80×24
label = Happy
prediction = [0.09105407 0.02053004 0.6935679 0.16475835 0.0300896]
prediction max = 2
label = Neutral
prediction = [0.00164392 0.977216 0.01710529 0.00229173 0.00174298]
prediction max = 1
label = Happy
Emotion Detector
Neutral
Happy
Screenshots
miniproj
PDF Documents
Geography NCERT
Jim Croce - Time in a bottl...73.mp3
Books EPUB
VID_20191217_120053.mp4
Vikram Grewal
Time Table


Terminal Shell Edit View Window Help
79% Fri 4:20 PM
MINIP — python test.py — 80×24
label = Sad
prediction = [0.14073521 0.01358735 0.293099 0.54786384 0.00471472]
prediction max = 3
label = Sad
prediction = [0.16967541 0.0407466 0.36484995 0.38543525 0.03929282]
prediction max = 3
label = Sad
Emotion Detector
Related images
Sad
Sad
Sad
Screenshots
miniproj
PDF Documents
Geography NCERT
Jim Croce - Time in a bottl...73.mp3
Books EPUB
VID_20191217_120053.mp4
Vikram Grewal
Time Table


Preview File Edit View Go Tools Window Help
78% Fri 4:21 PM
MINIP — python test.py — 80×24
label = Angry
prediction = [0.5976382 0.07105386 0.04864789 0.0842386 0.19842145]
prediction max = 0
label = Angry
prediction = [0.5569741 0.09064753 0.04854548 0.08309402 0.22073895]
prediction max = 0
label = Angry
Emotion Detector
Angry
Screenshots
miniproj
PDF Documents
Geography NCERT
Jim Croce - Time in a bottl...73.mp3
Books EPUB
VID_20191217_120053.mp4
Vikram Grewal
Time Table


python
78% Fri 4:22 PM
MINIP — python test.py — 80×24
label = Surprise
prediction = [0.02555055 0.3649297 0.0090444 0.00837875 0.59209657]
prediction max = 4
label = Surprise
prediction = [0.02459545 0.362661 0.00871516 0.0080797 0.5959487]
Surprise
Emotion Detector
Surprise
Screenshots
miniproj
PDF Documents
Geography NCERT
Jim Croce - Time in a bottl...73.mp3
Books EPUB
VID_20191217_120053.mp4
Vikram Grewal
Time Table

CONCLUSION:

Initially we defined a network and trained it so that it becomes capable to classifying the correct emotion and then using that trained model we tried to identify the emotions in real time environment. The model we made gave us a decent accuracy but took long time to train. This also gives us an idea that we can identify human emotions by reading and comparing the faces with images or data which is stored in knowledge base.