

Emotion Classifier

June 4, 2020

1 Prerequisites

```
[1]: import keras
      from keras.models import load_model
      from keras.utils import CustomObjectScope
      import tensorflow as tf
      from keras.models import model_from_json
      import numpy as np
      import os
      from keras.initializers import glorot_uniform
      import cv2
      import numpy as np
```

Using TensorFlow backend.

2 Loading in the Model

2.1 First, we open the JSON file for the Neural Network structure:

```
[2]: with open('model.json', 'r') as json_file:
      json_savedModel= json_file.read()
      model = tf.keras.models.model_from_json(json_savedModel)
```

2.2 Then, we load in the weights of the model from the H5 file:

```
[3]: model.load_weights('model.h5')
```

3 Running the Test Application

3.1 We define some objects for the test:

```
[4]: # Classes for each emotion
      emotion_labels = ['Angry', 'Disgust', 'Fear', 'Happy',
                       'Sad', 'Surprise', 'Neutral']
```

```

# Classifier for detecting faces in the video
face_classifier = cv2.CascadeClassifier('haarcascade_frontalface_default.xml')

# Access to the webcam
cap = cv2.VideoCapture(0)

# Font for the text on the video
font = cv2.FONT_HERSHEY_SIMPLEX

```

3.2 Then, we activate the webcam and perform live analysis of emotion:

```

[5]: while True:
    # Capture frame-by-frame
    ret, frame = cap.read()

    # Grayscale the image and detect the face
    gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
    faces = face_classifier.detectMultiScale(gray)

    for (x, y, w, h) in faces:
        # Draw a rectangle around the faces
        cv2.rectangle(frame, (x, y), (x+w, y+h), (0, 255, 0), 2)

        # Perform emotion calculation
        img = gray[x:x+w,y:y+h]
        model_img = cv2.resize(img, (48,48))
        input_img = np.reshape(model_img,
                                (1,48,48,1)).astype(np.float32)/255.0
        pred = model.predict(input_img)
        emotion = emotion_labels[np.argmax(pred)]
        score = np.max(pred)
        cv2.putText(frame,
                    emotion+" "+str(score*100)+'%', (x, y),
                    font, 1, (0, 255, 0), 2)

        # Display the resulting frame
        cv2.imshow('Emotion Classifier', frame)

        # Press 'q' to close the webcam
        if cv2.waitKey(1) & 0xFF == ord('q'):
            break

    # Once completed, deactivate the webcam and close all windows
    cap.release()
    cv2.destroyAllWindows()

```