Emotion Detection

John Wright

CPSC 4820, Clemson University, Clemson SC, USA

Email: jjwrigh@g.clemson.edu

Submitted: April 26th, 2022

Abstract: *In today's world, many of our decisions have been made using our emotions, rather than by being made with logical thinking. Unfortunately, this poses a problem where most emotion-based decisions are subject to error more often than logical decisions. Furthermore, examining someone's body language gives us great information about how someone is feeling in many situations. By developing technology that can help us predict what someone is feeling in intimate settings, we can avoid emotion based decisions, and learn more about how the mind works.*

---

1. INTRODUCTION

The inspiration of this project comes from wanting to expand upon topics that we learned in class. Specifically, in Assignment 5 when we first interacted with Haar Cascades, OpenCV, and face detection. I was interested to see what more I could do using these filters, and what benefit they could have. Creating an emotion detector seemed ambitious at first, considering that an understanding of machine learning is required in order to achieve this. Fortunately, there were plenty of resources available to me such as predetermined datasets, which made gathering thousands of images for my scripts to reference much easier.

Using my dataset, I developed an emotion detecting script that tracks you if you're happy, sad, or showing a "neutral" facial expression. I think there are many uses for a program like this, mostly in the medical field. It can be kind of strange to think of a computer telling you what you're feeling, but if regulated, it can be a great way to help us further understand the human mind.

2. APPROACH

Using OpenCV isn't going to be enough, since OpenCV doesn't have quite as useful machine learning capabilities as other programming libraries. Using Tensorflow and Keras would help this project immensely.

2.1 TENSORFLOW AND KERAS DESCRIPTION

TensorFlow is an open-source Python library for machine learning. It contains the tools necessary to create and train robust machine learning models. Keras is a deep learning API written in Python that is running on top of the machine learning platform TensorFlow.

2.2 FER-2013 DATASET

As I mentioned in the Introduction, I had used a pre-arranged dataset of thousands of 48x48 greyscale images to train my machine learning algorithm. This dataset is known as FER-2013, and it contains 7 classes for 7 different emotions. [1]

The FER-2013 Dataset was created by Pierre-Luc Carrier and Aaron Courville as part of a research project, and was released as a open source library commonly used in similar projects as this one.

2.3 IMPLEMENTATION OF TRAINING MODEL

In the training model, the most important thing to understand was what a batch size was and how it works. A batch size is basically how many samples of images from our library will be processed before the model gets updated. On top of this, we need to set a number of epochs, which is the number of times we pass through the dataset. [2]

Keras provides the opportunity to fit our model with image date augmentation, which is a fancy way to say that it allows us to expand the dataset by manipulating variables that track if an image has been flipped, rotated, or zoomed in, and more. It also validates this data using the same capabilities.

Before compiling the training model, the model needs checkpoints for when it goes through epochs. This will stop if the system runs out of memory, if the script passes an epoch and no improvement has occurred.

Lastly when its compiled, the model will optimize the categorical data, and measure accuracy as the main metric it's checking for. Compiling the training model produces a .h5 file, which is a file format produced by Keras that gives the set that is used in the driver code in order to see the results first hand. [3]

2.4 IMPLEMENTATION OF DRIVER CODE

After loading the Haar Cascade, as well as the set produced by the training model, as well as defining the classes by the emotions they represent (Happy, Sad, Neutral), we are ready to detect emotions. Using a while loop that quits once the user presses 'q', for every face that is recognized in the frame, call the Haar Cascade to draw a rectangle around it while simultaneously calling the training model to make a prediction on what emotion is based on a facial expression.

```
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)


    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)
```

Figure 1: Code that represents the conditions that must be met for making a prediction

More specifically, the code listed in the figure above will do the following. If roi_gray (array that stores a face and then gets resized to 48x48) is not equal to 0 (that is if a face is detected) then the array gets converted to a float then divided by 255 to get a value between 0 and 1 so the model can make an accurate prediction.

After a prediction has been made, it will display the emotion predicted on top of the rectangle hovering above the face.

3. RESULTS

The model predicts emotions fairly accurately. I ran into a couple of errors where if I have a neutral face it says I am sad, and then when I smile it says I am neutral, but usually the model predicts correctly. I do notice it's quite difficult for the model to state that it notices a neutral face, but overall I'm pleased with the results of the project.

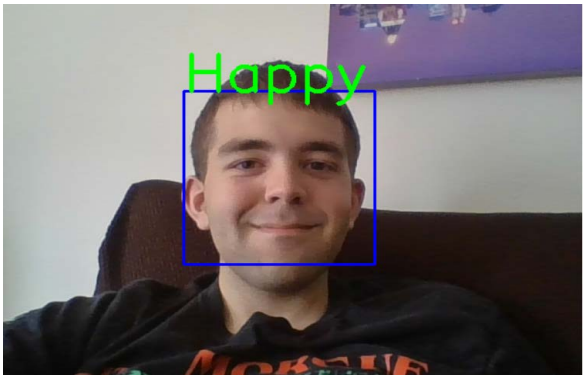

Figure 2: Myself testing to see if the model predicts if I am happy.

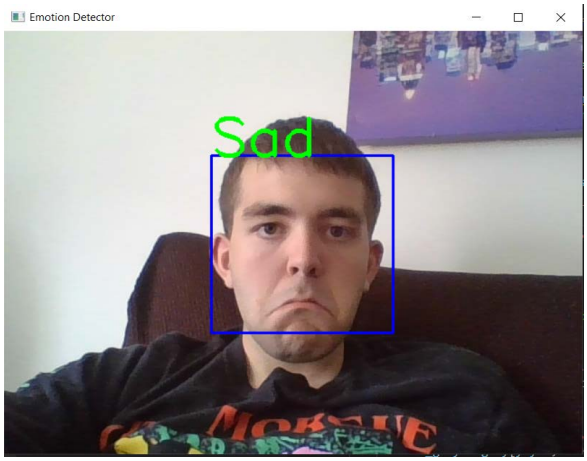

Figure 3: Myself testing to see if the model predicts if I am sad.

4. CONCLUSION

I think this was a great way to wrap up the semester and expose myself to how machine learning works as well as expanding upon how Haar Cascades work and how useful they are for facial recognition software like this. Although robust, this emotion detection software was a great way to test my understanding of computer vision, and I'm excited to see what the future holds for this field.

5. REFERENCES

1] Sambare, Manas. “Fer-2013.” Kaggle, 19 July 2020, https://www.kaggle.com/datasets/msambare/fer2013.

[2] “Custom Training: Walkthrough :   Tensorflow Core.” TensorFlow, 19 Feb. 2022, https://www.tensorflow.org/tutorials/customization/custom_training_walkthrough.

[3] “Save and Load Keras Models :   Tensorflow Core.” TensorFlow, 11 Jan. 2022, https://www.tensorflow.org/guide/keras/save_and_serialize#keras_h5_format.