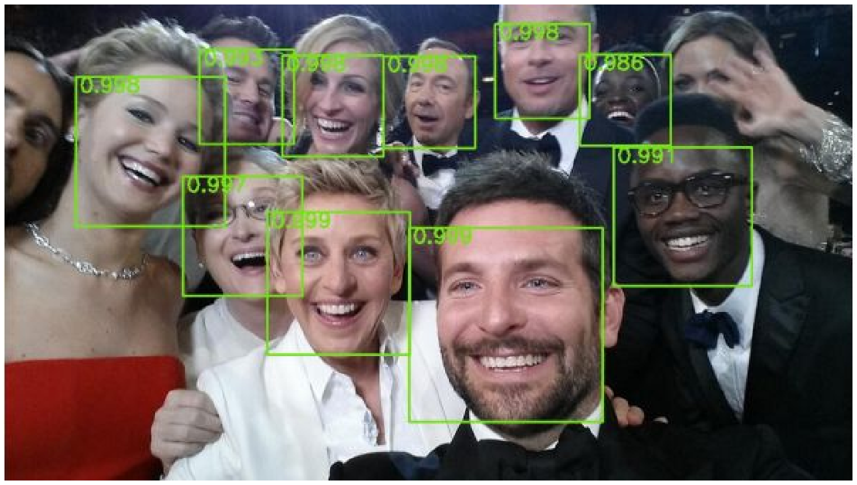
March 8th, 2019
Code Author: Neeraj Dharmadhikari

Project Summary

This project in my opinion out-does all my previous projects since it has a small version of machine learning component. It requires us to train the machine to learn what a face looks like and to find it in different images. Cameras nowadays are more and more intuitive and can tell us where in each picture frame a face is. It also tells us whether or not the face it finds is in good lighting/proper exposure. To figure out if and where a particular object is in an image, we need to train our machine to know what the object looks like and then let our machine detect that object/face from what it has learned.

To help visualize the concept, if I were to create a template for a face and then give the above image and ask to find all the faces that follow the template, then I should get this result:

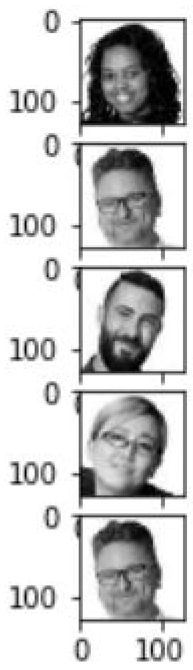


Algorithm summary: We'll break this down in multiple parts -

1. Given an image, convert it to grayscale, then based on pixel values we will need to create an image gradient pair of 2d arrays. One array containing gradient magnitudes at each location and the other array containing the orientation of the gradient.
2. We will go a select nxn block of pixels through the image and create a histogram oriented gradient descriptor that will serve as a "template" of the image.
3. We will create this HOG descriptor for multiple face images that we can use to make a better template. (the more face images we train it with, the better our template will be)
4. We will then also do the same with the actual image we are looking for faces in.
5. Go through comparing the HOG descriptor of the image in question with the template we created in part 3, to look for a match. The stronger the match, the greener the color of the box we'll put around the matched spot. If it's a bad match, then we'll make it red. For the sake of this project we'll say that the number of faces it should look for should be given. In the future I'll try to implement a way to make this work without given this parameter.

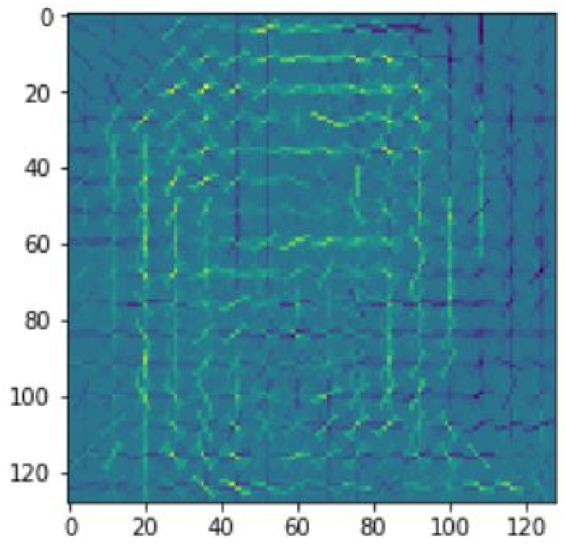
My results:

The images I fed the program:



The template it created:

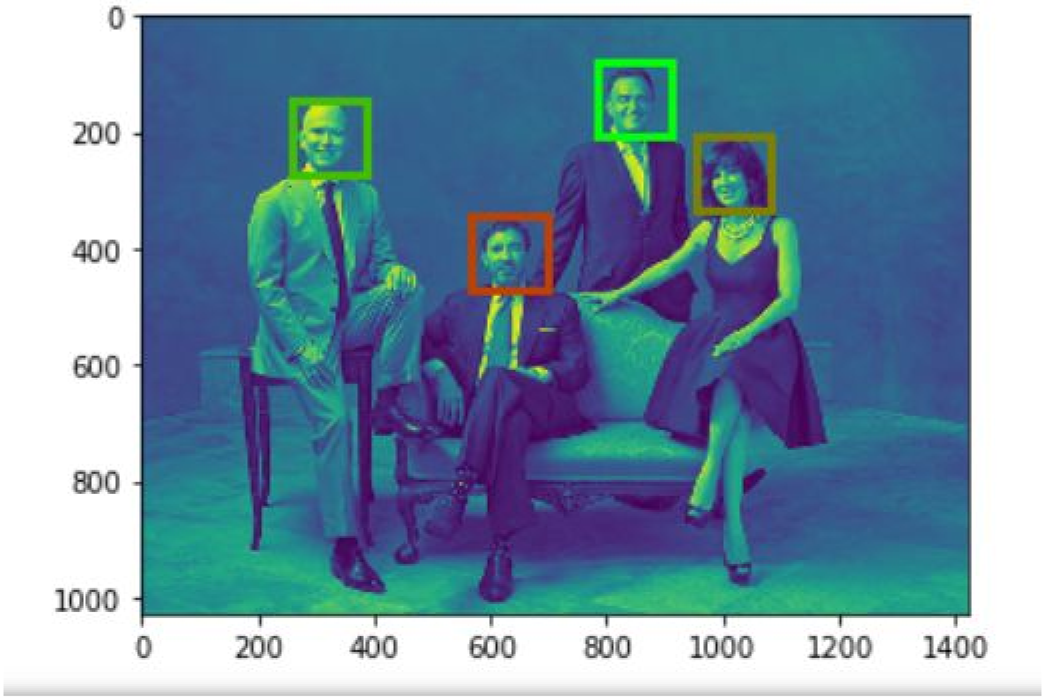
HOG descriptor version of template:



I also fed it negative images that did not contain faces and told it to subtract from the HOG descriptor version of the template to make our template more accurate.

The image I'm passing for detection test (and manually telling the program to detect 4 faces) :

The results are astonishingly wonderful :)



I hope that this document was able to give you a general idea of the ideas behind the project and how my results turned out. Refer to the python notebook to understand specific parts of the algorithm as the code has plenty of comments.