Weekly Status Report 1
Week 1
Name: Joe Lanzi

Activities and Accomplishments

- Designed and create pipelines for facial recognition

Problems

- Facial Mask eliminates simple methods and causes expensive computations that must be solved with dedicated GPU's locally or cloud-based computing

Plans

- Create Datasets for facial recognition with facial mask and get results

Individual Hours Worked This Week on the Project

(Note: This is the total number of hours you actually worked on Capstone. Eg, if you have a team of 4 people, you worked together for 2 hours this week. **Individually, you worked 2 hours/4 =0.5 hour this week. You put 0.5 here.)**

I worked on the project for **30 hours**

The group met for a total of **2 hours/5 people = 0.4**

30.4

Discussion

The report below includes my own work for the winter and during the first week of Spring 2021.

I've informed the group to start working on the power of the whole system, integrate all hardware systems together, and start collecting testing data.

Facial Detection & Recognition

To understand facial recognition, one must understand the process of image segmentation. One of the first step in image segmentation is locating the region of interest [ROI] within an image. This step can be ignored if the whole image is being classified as one entity. But since the facial recognition will include multiple objects and/or people in the same image, regions of proposals must be identified and specified to run a pipeline able of multiclass classifications.

I used 2 methods in locating the coordinates of the faces in an image. A Convolutional Neural Network [CNN] & a Histogram of Oriented Gradients [HOG].

Facial Detection

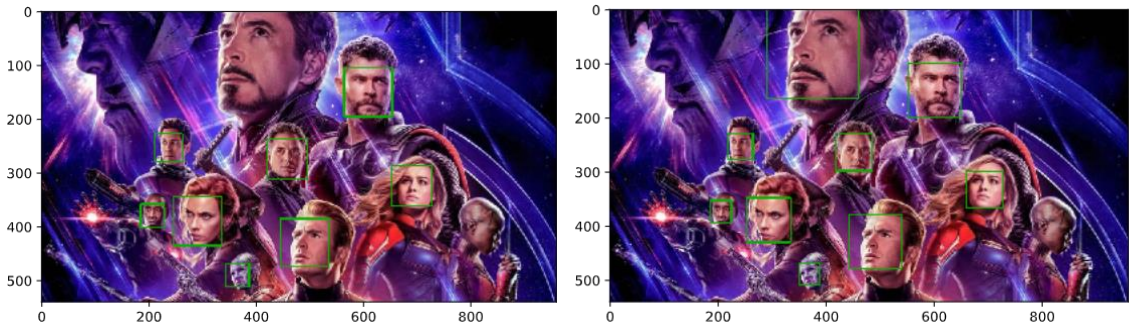
HOG is a linear SVM learning algorithm used to perform face detection. Its basic idea is to divide the image into small connected cells, compute histograms for each cell, convert histograms into feature vectors, and look for specific labels that specifies a facial structure.

CNN uses convolution methods and dense layers found in neural network to create weights and predictions to be able to classify a facial feature accurately. This method is more accurate but takes significantly longer to run in multiple regions.

Given a sample image below:

HOG took 0.318 seconds but was only able to detect 8 out of 11 faces. [Left Image]

CNN took 42 seconds but was able to detect 9 out of 11 faces. Without GPU [Right Image]



[HOG] [CNN]

HOG is not good at detecting faces in angle or if they look odd.

CNN also struggles in detecting faces in angle due to it is training sets. This can be fixed by retraining the model with customized training sets.

A third method consist of a more computationally expensive method called **Felzenszwalb super pixel generation method**. This method consists of four mathematical equations to create regions of proposals and creating layers based on the input image. With these mathematical equations, we can create layers based on the object's the color similarity, texture similarity, size similarity, and fill similarity. We can then iterate over these layers to produce multiple regions of interest. Once these regions of interest are created, a model or weight must be used along with a non-maximum suppression to clean up the detection regions.

Facial Detection Conclusion

For our application, the HOG method will be used due to its speed. Although CNN performs more accurately, almost 60 seconds in images with multiple faces will not suffice for live performance of our project. To use the Felzenszwalb super pixel generation method, either the device or the webserver must include a powerful GPU to compute and train these models.

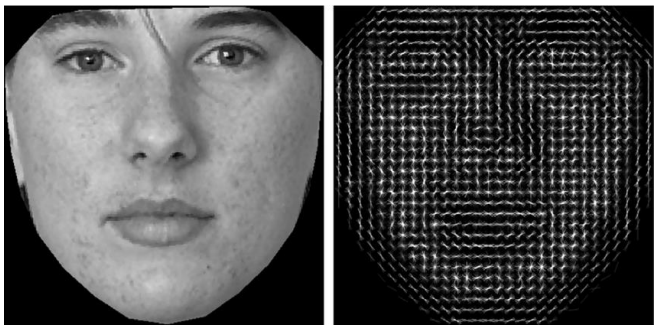
Facial Recognition

The facial recognition occurs after the given the face has been detected. But to do so, the pipeline must include pretrained weights of the faces being identified. Therefore, to have a fully functioning pipeline, the facial recognition pipeline must be able to load a pretrained weight from the model trainings.

1. Model Training

Model training includes the facial detection pipeline. After a face has been located with it's coordinates. Either HOG or CNN must be applied to create weights from the given identities.

- HOG: Once a face is detected, it is then transformed into a 128 d-embedded key. The more data that is included in one's facial identity, the more key can be related to this one person. Once 128 d-embedded keys have been developed, these are then saved as the pretrained weight used for classification.



- CNN: A ResNet34 architecture will be used in this project which includes 34 dense layers. This neural network will be trained from the given training set to be compared to the new untrained images. This convolutional neural network is capable of creating it's own neuron weights that can predict what it's classifying. With experience, this model can be divided into multiple batches for GPU training using the CUDA architecture which speeds up the training process and identification process. The model can also be retrained with new images which allows the model to increase in accuracy in more time and data.

```
├── dataset
│   ├── alan_grant [22 entries]
│   ├── claire_dearing [53 entries]
│   ├── ellie_sattler [31 entries]
│   ├── ian_malcolm [41 entries]
│   ├── john_hammond [36 entries]
│   └── owen_grady [35 entries]
```

For this case, the models were trained with 218 images from the cast of the Jurassic Park.

2. Facial Recognition Pipeline

Once the weight has been created by either HOG or CNN methods, the pipeline can now identify facial features and compare new faces in the untrained image.

- First, the pipeline will consist of a facial detection algorithm to locate the new face in the new image that's never been trained on
- Then, the pipeline will simple iterate through the loaded weight to compare the new face to the labeled weights from either HOG or CNN
- Finally, if a match is found, it'll return the label it matches. If not, it will return an unknown data.

The following images are outputs from both methods.

- The HOG method takes just about 1 second.
- The CNN Method takes just about 2-3 seconds with a GTX 1080 GPU, but takes about 42 seconds without a dedicated GPU and only utilizing a CPU

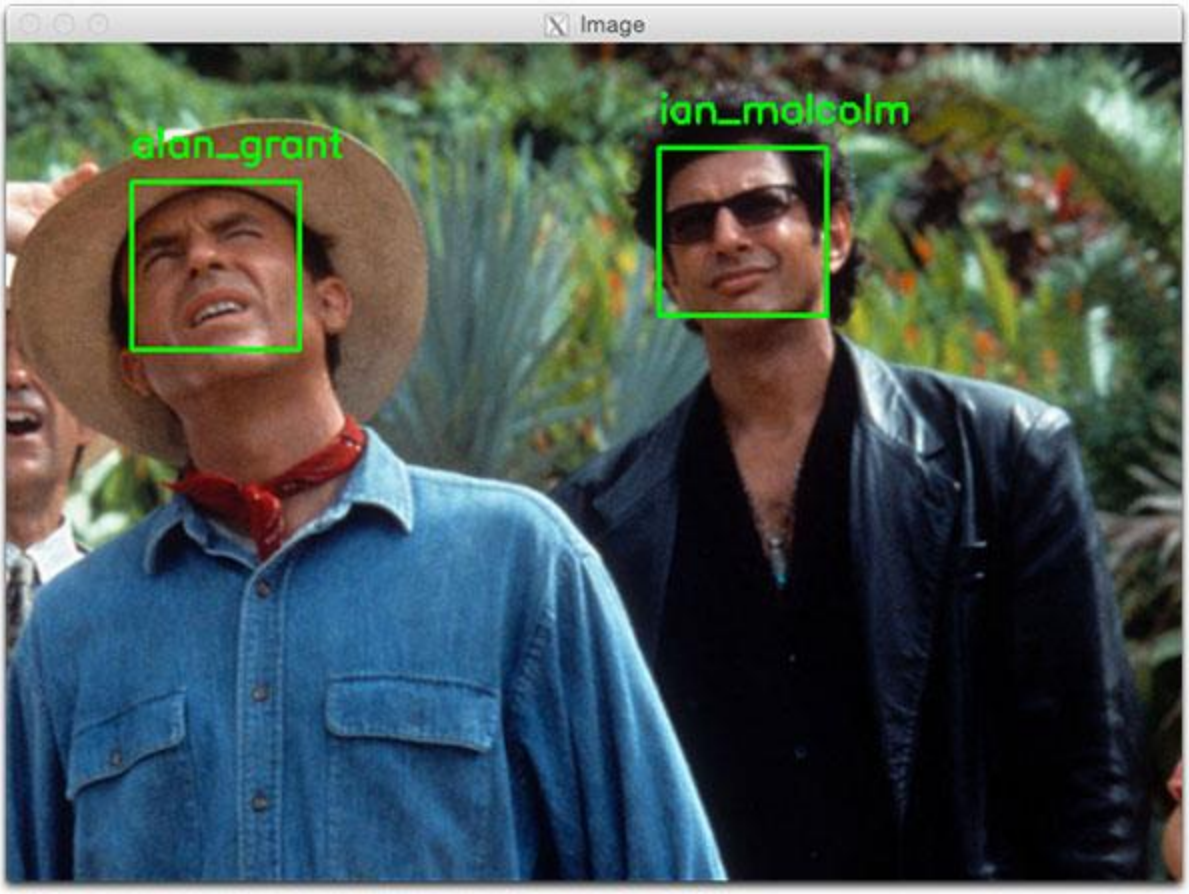


Image
alan_grant
ian_malcolm

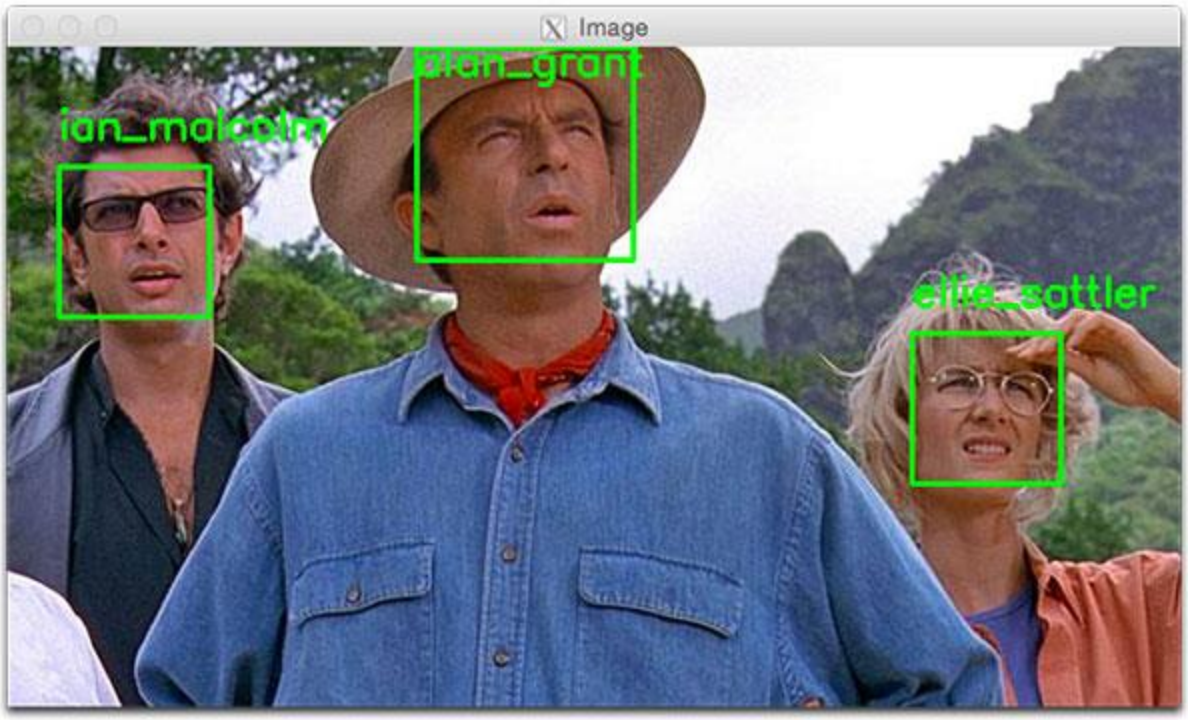


Image
ian_malcolm
alan_grant
ellie_sattler

Facial Recognition Conclusion

Speed and computation again affect the method of choosing. If this pipeline is being used in smaller devices with no dedicated GPU's, the HOG must be used otherwise it will take too much time to identify even 1 person. Also, the HOG method will never be as accurate as the CNN. If using HOG, a lot of misclassification will occur.

Data Collection

Data collection will consist of two individual methods. Manual & Face Enrollment using camera.

- **Manual Face** Enrollment will consist of individually or automatically labeling image data from database.

- **Face Enrollment using camera** will consist of a library OpenCV and a camera that can capture image data to update and add on to the database to make the model more accurate. This will consist of a loop that utilizes a certain the confidence threshold from the facial recognition. If the confidence is high enough, these facial data will populate with the current labeled dataset to increase model accuracy.

This dataset collected will be use to train the models and create callable weights used in the facial recognition pipeline. Then, new dataset will be collected for model retraining and increasing the accuracy of the model to reflect different scenarios.

Final Conclusion

So far, I was able to create the working pipeline to run a pretty basic facial recognition. The problem present is when individuals are wearing face mask especially during the pandemic. My hypothesis is that the simple linear vector analysis found in the HOG method will simply not work enough. Therefore a convolutional neural network focused on identifying an individual based solely on separate features is a must. This means that a bigger computation architecture will be required for our project. Since one of the main goal of the project is portability, the computation must occur in the webserver where the data will either be computed locally or by using cloud base methods.