Face Detection in the Wild, Project 3

Keshav murthy Ramachandra
50360333

Part A, Face Detection

- Initially, I used the pre-trained HaarCascade face classifier provided in XML format named frontal_face_default classifier.
- Later, I tried the open cv deep neural network(DNN) module for face detection as the accuracy from the haar cascade was not good enough.
- The model architecture was defined in the .prototxt file and the weights for the model were specified in the .caffemodel file.
- I also used the Multi-Task Cascaded Convolutional Neural Network (MTCNN). Although I received a very good F1 score with this, I couldn't use it as I didn't find a native open cv implementation for it.
- Since the DNN Face detector yielded both a good F1 score and followed the project guidelines, I stuck to the DNN face detection.

Discussion and Results

- The Haar Cascade classifier was not sufficient to detect the side faces.
- The haar cascade classifier also gave many false positives.
- The F1 score I got from it was around 0.79 initially. For this, I used a scale factor* of 1.5 and min neighbors* as 5.
- After changing the scale factor to 1.2 and min Neighbors to 4, I got an F1 score of 0.81. This also reduced the number of false positives.
- When using full-size images for the DNN model, it did not perform well on images where faces were large.
- After resizing the images to (300,300), the F1 score increased to 0.86. Although it detected almost every face in an image, it performed very badly when the face sizes were large.
- Then, after resizing the images to (190,190), the F1 score increased to 0.9. Although it missed out on faces that were too small, overall it performed very well without giving any false positives.

scaleFactor: parameter specifying how much the image size is reduced at each image scale.
minNeighbors: Parameter specifying how many neighbors each candidate rectangle should have to retain it.

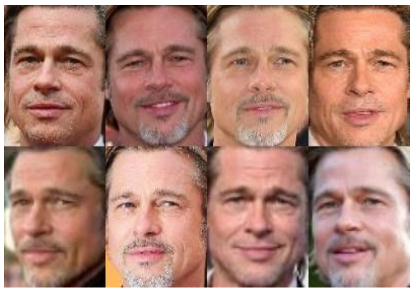
Face Detection in the Wild, Project 3

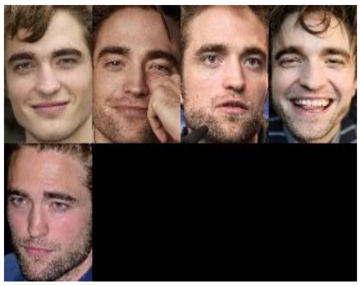
Part B, Face Clustering using KMeans

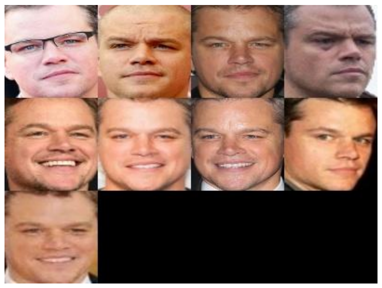
- In the beginning, I used Haar Cascade for detecting faces from the images of the faceCluster_5 folder. With a scale factor of 1.2 and 4 nearest neighbors, I got a perfect result for the provided dataset most of the time but sometimes the results were inaccurate.
- I observed it was because of the random centroid initialization. Sometimes, the initial centroids were not pushed far apart which did not cover the entire space.
- Later, I assigned the first centroid to the location of a randomly selected data point and then chose the subsequent centroids from the remaining points based on a probability proportional to the squared distance away from a given point's nearest existing centroid. This ensured efficient space coverage over the entire dataset.
- But when I tried to cluster a different dataset from the internet, one of the clusters always had wrong results. I observed it was due to the detection of some blurry faces in some of the images. Although the project description stated that there will be only one face in an image, I wanted to solve this.
- Another reason was that Haarcascade gives a lot of false positives which was evident from the first part of the project.
- So, I switched to DNN face detection. This removed the false positives.
- But DNN still detected background blurry faces. For this, I increased the confidence threshold to 0.85 to ensure that the weak detections are removed. Since the project stated that there should be only one face in each image, another possible technique might be is to pick the image with larger confidence to assign to the cluster.

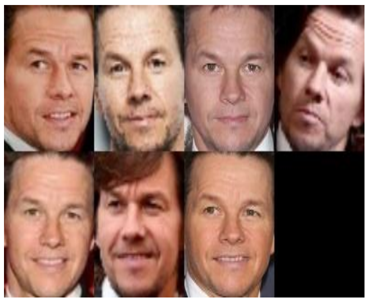
Face Detection in the Wild, Project 3

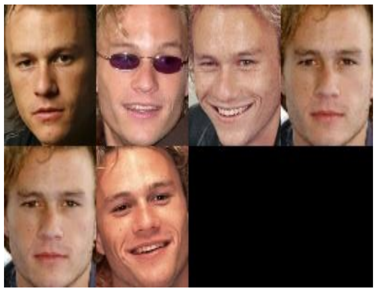
Cluster Results:


Cluster 0


Cluster 1


Cluster 2


Cluster 3


Cluster 4