Face Detection using Haar Cascade and OpenCv DNN Module

Rugved Hattekar: 50320920

Introduction:

Face detection is basically a technology widely used in the modern world in variety of applications which mainly focuses on detecting faces from digital images. There are various algorithms such as Haar Classifier, Viola Jones algorithm which are widely used in this domain. In this project I will be using a deep neural network trained using Haar Cascade Classifier to detect the image at any given angle also the side faces and blurry background faces.

Goal:

The goal of the task 1 is to implement face detection algorithm using Opencv and test it on the test image dataset given in the assignment.

What is Haar cascade classifier?

Haar cascade is a machine learning based object detection algorithm which was proposed by Viola and Jones in their paper of “Rapid Object Detection using a Boosted Cascade of Simple Features". It is a machine learning based approach where a cascade function is trained from a lot of positive and negative images. It is then used to detect objects in other images. The Algorithm has following four stages:

1. Haar Feature Selection
2. Creating Integral Images
3. Adaboost Training
4. Cascading Function

Haar Feature Selection:

A Haar Feature is consists of adjacent rectangular regions describing the regular dark and light regions. The best Haar features are based on the Adaboost algorithm which selects best Haar Features out of more than 16000+ features. During the detection phase, a window of the target size is moved over the input image, and for each subsection of the image and Haar features are calculated. I the computation is below the learned threshold; the image is assumed to have a facial feature.

Implementation of Haar Cascade Classifier and Limitations:

Opencv has a pretrained Haar cascade classifier class. I Initially implemented the Opencv’s haar cascade classifier and detectMultiScale method. After implementation the Fbeta score I observed was below 0.5. While investigating the reasons behind such a poor Fbeta, I realized following points:

1. The Haar cascade classifier is variant to the rotation. If the image to be detected is rotated at certain angles, it is difficult for the Haar cascade to recognize it.
2. Haar cascade classifier is heavily dependent on the images it is trained on. Any other image not resembling the training set, tends to have poor accuracy.
3. One-sided faces are not detected in the Haar classifier.

Implementation of DNN:

In the project I have tried to improvise on the Haar Cascade by using Opencv DNN module. The DNN module is trained using CNN which has kernels very similar to haar features. However, because Haar Features have to be determined manually, there is a certain limit to the types of things it can detect. If you give classifier (a network, or any algorithm that detects faces) edge and line features, then it will only be able to detect objects with clear edges and lines. Even as a face detector, if we manipulate the face a bit (say, cover up the eyes with sunglasses, or tilt the head to a side), a Haar-based classifier may not be able to recognize the face. A convolutional kernel, on the other hand, has a higher degree of freedom (since it's determined by training), and could be able to recognize partially covered faces (depending on the quality of the training data). This Dnn is using Resnet architecture.

Steps of Implementation:

1. Read all the images using "glob" Python library.
2. Read the images in sorted order using OpenCv
3. Import the dnn module from opencv using the pretext and pretrained weights.
4. This weights are used for the prediction of the face.
5. Run the image through the model.
6. Interpret the output of the model to select best detection with more than 50% confidence and to obtain the [x,y,h,w] coordinates of the detected face.
7. Draw a rectangle around the region of interest using the above obtained coordinates.
8. Save the image and the dump the coordinates in the Json file.

References:

https://docs.opencv.org/master/db/d28/tutorial_cascade_classifier.html

https://towardsdatascience.com/face-detection-models-which-to-use-and-why-d263e82c302c

https://towardsdatascience.com/extracting-coefficients-of-opencv-face-detection-dnn-model-7f3d944898b9

https://towardsdatascience.com/extracting-coefficients-of-opencv-face-detection-dnn-model-7f3d944898b9

https://www.pyimagesearch.com/2018/02/26/face-detection-with-opencv-and-deep-learning/