The No Entry Sign Challenge

Assignment Report

Rares Bucur(vw19308)

Subtask 1: The Viola-Jones Object Detector

Ground Truth and Visualisation

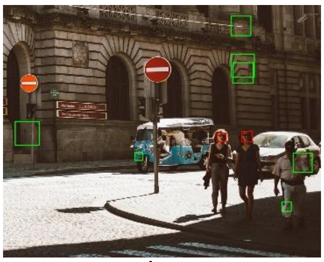
The ground truth is given by all the true values of an object within an image. As shown in the example images below, the red rectangles are displaying the ground truth values for the faces and the green ones show all the faces detected by the Viola-Jones detector.

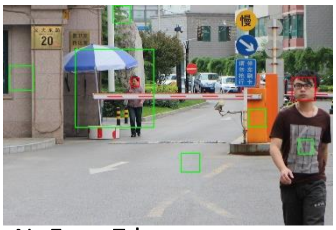

NoEntry1.bmp


NoEntry2.bmp


NoEntry5.bmp


NoEntry4.bmp


NoEntry11.bm


NoEntry7.bmp

NoEntry4.bmp, NoEntry7.bmp show the extent to which I chose the ground truth values for frontal faces. Images saved with faces_truth.cpp

IOU, TPR, F1-SCORE

Intersection over union is an evaluation metric used to measure the accuracy of an object detector on a particular dataset. Computing the IoU can therefore be determined via:

$$IOU(A,B) = \frac{A \cap B}{A \cup B}$$

where A and B are the sets of rectangles, $(A \cap B)$ is the area of overlap or intersection and $(A \cup B)$ is the area of union

The true positives rate or the hit rate (**TPR**) represents the number of correct detections over the number of total true faces found. The set of test images provided contains 9 images which did not include any frontal faces, thus the TPR did not have any relevant results. The TPR is a useful way to test how performant the detector is but it gets difficult to assess how meaningful the TPR is when the test image does not contain the object that you are trying to detect.

There is a possibility to always achieve a TPR of 100% because of the manually set threshold for the intersection over union. Decreasing the threshold rate, naturally increases the positive detection.

The **F1-SCORE** is another way of testing the efficacity of our detector, which is mainly a measurement of the accuracy. It can be calculated via:

$$F1 = \frac{TP}{TP + 0.5(FP + FN)}$$

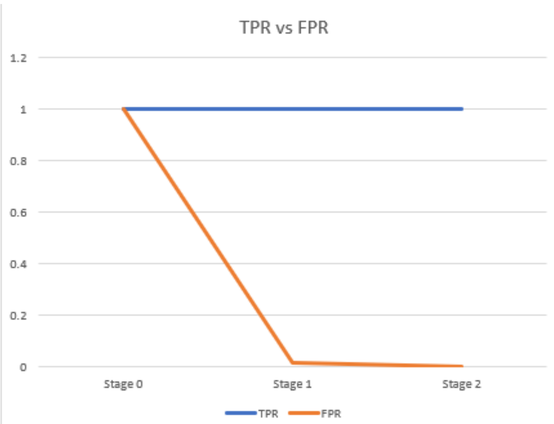
Where TP is the true positives, FP is the false positives and FN is the false negatives

The following results were given by setting the IOU threshold to 0.5:

Test Image	TPR	F1-Score
NoEntry0.bmp	-	-
NoEntry1.bmp	1	0.2
NoEntry2.bmp	1	0.25
NoEntry3.bmp	-	-
NoEntry4.bmp	1	1
NoEntry5.bmp	1	0.33
NoEntry6.bmp	-	-
NoEntry7.bmp	0.5	0.22
NoEntry8.bmp	-	-
NoEntry9.bmp	-	-
NoEntry10.bmp	-	-
NoEntry11.bmp	1	0.36
NoEntry12.bmp	0	0
NoEntry13.bmp	-	-
NoEntry14.bmp	-	-
NoEntry15.bmp	-	-
Average	0.785	0.337

Subtask 2: Building and Testing our own Detector

Training Performance



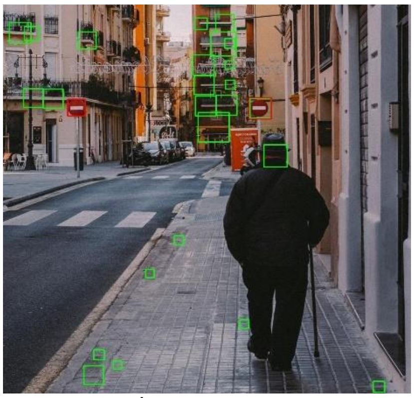
NoEntry2.bmp

The false positive rate (FPR) is another way to test the detector, representing the number of false positives over the number of true negatives.

During the training, the TPR stayed at 500:500(100%) for all stages, as for the FPR, it shows an exponential fall throughout the three stages.

Testing Performance

The example images below were produced using the Viola-Jones detector with the new trained dataset for signs. The green rectangles show all the detected signs and the red show the ground truths.



NoEntry0.bmp

NoEntry4.bmp

Images saved with noentry_viola.cpp

The TPR and F1 score for each test image is shown in the table below:

Test Image	TPR	F1-Score
NoEntry0.bmp	0.5	0.07
NoEntry1.bmp	1	0.12
NoEntry2.bmp	1	0.11
NoEntry3.bmp	1	0.44
NoEntry4.bmp	1	0.12
NoEntry5.bmp	0.6	0.26
NoEntry6.bmp	0	0
NoEntry7.bmp	1	0.18
NoEntry8.bmp	1	0.75
NoEntry9.bmp	1	0.22
NoEntry10.bmp	1	0.5
NoEntry11.bmp	0	0
NoEntry12.bmp	0.57	0.38
NoEntry13.bmp	0	0
NoEntry14.bmp	0	0

NoEntry15.bmp	1	0.66
Average	0.711	0.254

The results show that the detector can identify approximately 71.1% of signs in an image, which is also 9.42% lower than the face detector. This may be due to the fact that faces are easier to recognize, having more distinctive values. Also, the detection of the signs being part of a background can get more difficult.

Different viewing angles were used to train the classifier, but that is not enough to detect all the signs in an image, as the detector mainly identifies the no entry signs that were facing the camera when the photo was taken.

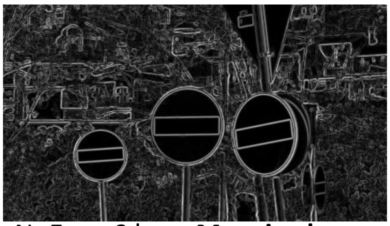
On top of the issues mentioned above, the classifier was trained on sample images taken from a single simple cropped image of a no entry sign. A no entry sign in a real city may not always look the same, it can have different angles or contrast as we can also see in some of the test images.

We can also see a decrease of 24.62% in the F1 score of the no entry sign detector compared to the F1 from the face detector. This results in the detector being less accurate because of having more false positives in the image than the face detector.

Subtask 3: Integrating with shape detectors

Hough Details

The images below show the hough space for circles and how it can be used to filter bounding rectangles:



NoEntry6.bmp **Magnitude**

NoEntry6.bmp **Hough Space Circles**

NoEntry6.bmp

NoEntry8.bmp **Magnitude**

NoEntry8.bmp

NoEntry6.bmp **Hough Space Circles**

Evaluation

The TPR and the F1 Score are shown below for the test images created with the aid of the hough circle transform. I decided not to use it together with hough line transform as I did not implement it in a way so that it can work out the lines from cropped images. Thus, choosing to use it for the entire image would decrease the accuracy of our detector.

Test Image	TPR	F1-Score
NoEntry0.bmp	0.5	0.4
NoEntry1.bmp	1	0.66
NoEntry2.bmp	1	0.66
NoEntry3.bmp	1	0.44
NoEntry4.bmp	1	0.5
NoEntry5.bmp	0.1	0.16
NoEntry6.bmp	0	0
NoEntry7.bmp	0	0
NoEntry8.bmp	1	0.85
NoEntry9.bmp	1	0.66
NoEntry10.bmp	1	0.66
NoEntry11.bmp	0	0
NoEntry12.bmp	0.57	0.38
NoEntry13.bmp	0	0
NoEntry14.bmp	0	0
NoEntry15.bmp	1	0.66
Average	0.573	0.376

The average F1 for the new detector saw an increase of 32.44%, as for the TPR, it saw a slight decrease of 19.4%.

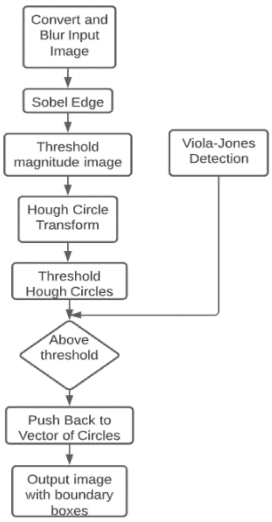
The advantages of the enhanced implementation:

- Lots of false positives are filtered out based on the threshold value we choose, thus increasing the precision of our detector.
- The F1 score is increased, increasing the accuracy of the new detector.

The disadvantages of the enhanced implementation:

- Some false positives still remain, meaning that further filtering would be required.
- Although most false positives would be removed, it also means true positives would also at times be discarded.

Detection Pipeline



The rationale behind this detection method:

- Detected signs from the Viola-Jones detector were used as the starting point, as it had a high TPR.
- I did not use Hough Line Transform as it actually decreases the accuracy of the detector if it is not implemented to process just cropped images of no entry signs.
- Hough Circle Transform was used to detect no entry circles in the image. The threshold is used to count the center pixels and if the counter is above a given threshold, the found signs are pushed into the vector.

Subtask 4: Improving the detector

Idea

Due to time constraints, I did not manage to implement the following approach, but the idea was to improve the overall true positive rate and the F1 score of the detector by combining overlapping rectangles. If we manage to combine multiple boxes like these into a single one, it should be a much better fit for the ground truth.

I would have extended the function in which I add the list of rectangles, by checking if a rectangle overlaps with any other. If an overlap is found, break out of the for loop and remove the box from the vector. After that, if there are intersecting rectangles found, find a new one which should be the average of the two found. In the end, we create a new rectangle and push it back to the vector and if no overlap is found, then just push back the new rectangle into the found vector.

In addition to the idea of grouping the overlapping rectangles, I managed to implement a way of combining the multiple detected circles into a single one by choosing the best radius of them all. This addition did not seem to improve in some way at least the F1 score.

Overall, the final implementation does an acceptable job detecting faces and no entry signs and a slightly better job than the original implementation of the Viola-Jones detector. There are still many problems with this way of implementation and further optimization can be made.