ECEN 649

PATTERN RECOGNITION

FALL 2019

TEXAS A&M UNIVERSITY

PROJECT REPORT ON
IMPLEMENTING VIOLA-JONES ALGORITHM

Name: AKSHAY HIREGOUDAR

UIN: 128003713

Date: December 9, 2019

This report is submitted as part of the final project of ECEN 649 – Pattern Recognition course at Texas A&M University. The main objective of the project is to implement the Viola-Jones algorithm which is based on the idea of AdaBoost. It is mainly used in face detection applications. The original paper can be found here.

The CMU dataset which contains both face and non-face figures is used to train and test our algorithm. The training set contains 500 face and 2000 non-face images whereas the testing set contains 472 face and 2000 non-face images. The dataset can be found here.

The algorithm has been implemented in a Jupyter notebook using Python 3. The notebook can be found here which contains the entire code along with the results, plots, and images. The Readme file containing information about the required packages and implementation procedure can be found here.

GitHub repository link: https://github.com/akshayhiregoudar/Face-Detector

2.1 Extract Haar Features

- The total number of Haar Features is **63960**.
- There are **17100** type 1 (two horizontal) features.
- There are **17100** type 2 (two vertical) features.
- There are **10830** type 3 (three horizontal) features.
- There are **10830** type 4 (three vertical) features.
- There are **8100** type 5 (four) features.

2.2 Build your AdaBoost detector

Best Features obtained by implementing AdaBoost

- **Detector with 1 round of AdaBoost**
  - Type: 2h (Two Horizontal)
  - Position: (x,y) = (8,3)
  - Width: 2
  - Length/Height: 10
  - Threshold: -1.602218
  - Precision/Accuracy: 0.34

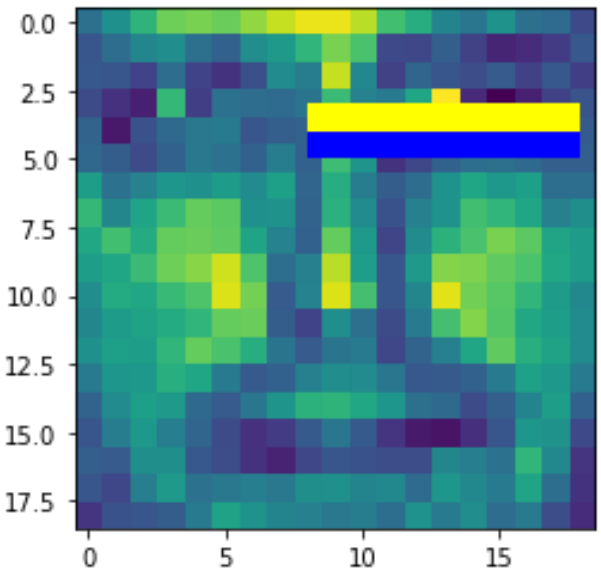

Figure 1: Top 1 feature for 1 round of AdaBoost

- **Detector with 3 rounds of AdaBoost**
  - Type: 2h (Two Horizontal)
  - Position: (x,y) = (11,10)
  - Width: 6
  - Length/Height: 1
  - Threshold: --0.846878
  - Precision/Accuracy: 0.57

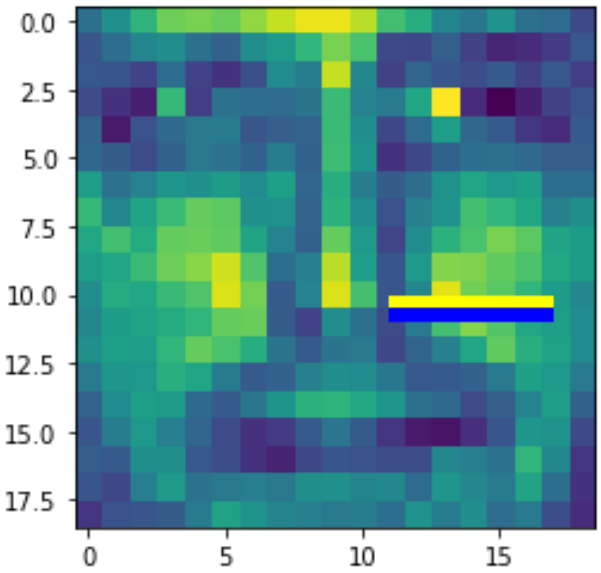

Figure 2: Top 1 feature for 3 rounds of AdaBoost

- **Detector with 5 rounds of AdaBoost**
  - Type: 2v (Two Vertical)
  - Position: (x,y) = (12,0)
  - Width: 4
  - Length/Height: 2
  - Threshold: -1.785327
  - Precision/Accuracy: 0.66

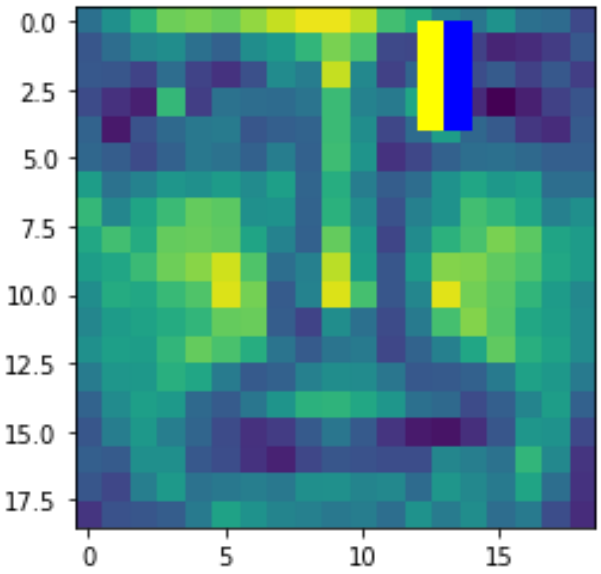

Figure 3: Top 1 feature for 5 rounds of AdaBoost

- **Detector with 10 rounds of AdaBoost**
  - Type: 2v (Two Vertical)
  - Position: (x,y) = (8,11)
  - Width: 3
  - Length/Height: 2
  - Threshold: 0.892658
  - Precision/Accuracy: 0.80

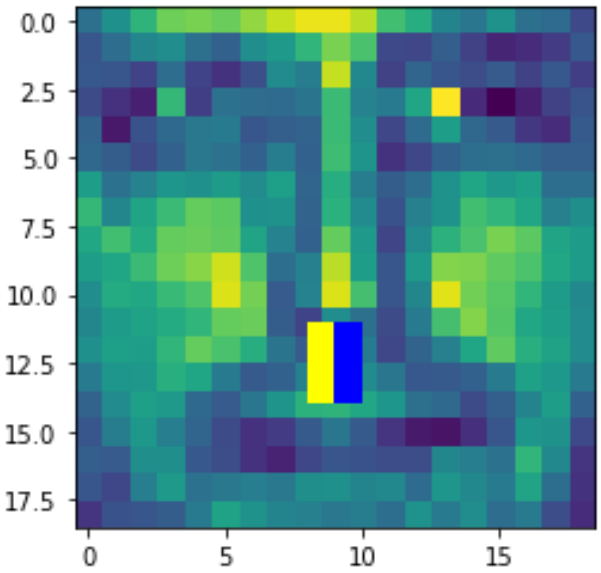

Figure 4: Top 1 feature for 10 rounds of AdaBoost

Change in rate of accuracy, false-positive rate, and false-negative change

- AdaBoost rounds: **1**
  - Total accuracy: 0.34
  - False Positive: 0.14
  - False Negative: 0.70

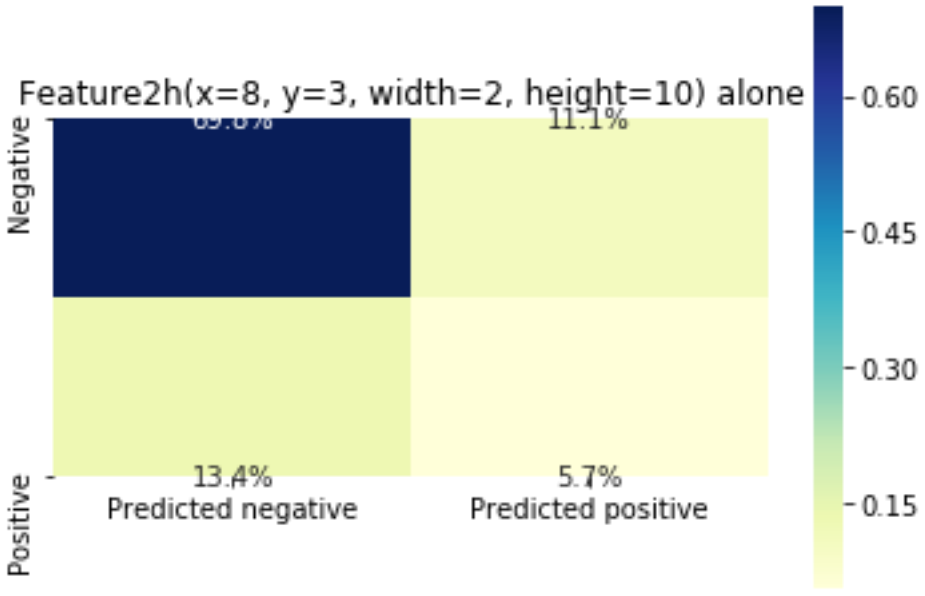

Figure 5: Heatmap showing the accuracy parameters of 1 round of AdaBoost

- AdaBoost rounds: **3**
  - Total accuracy: 0.57
  - False Positive: 0.06
  - False Negative: 0.68

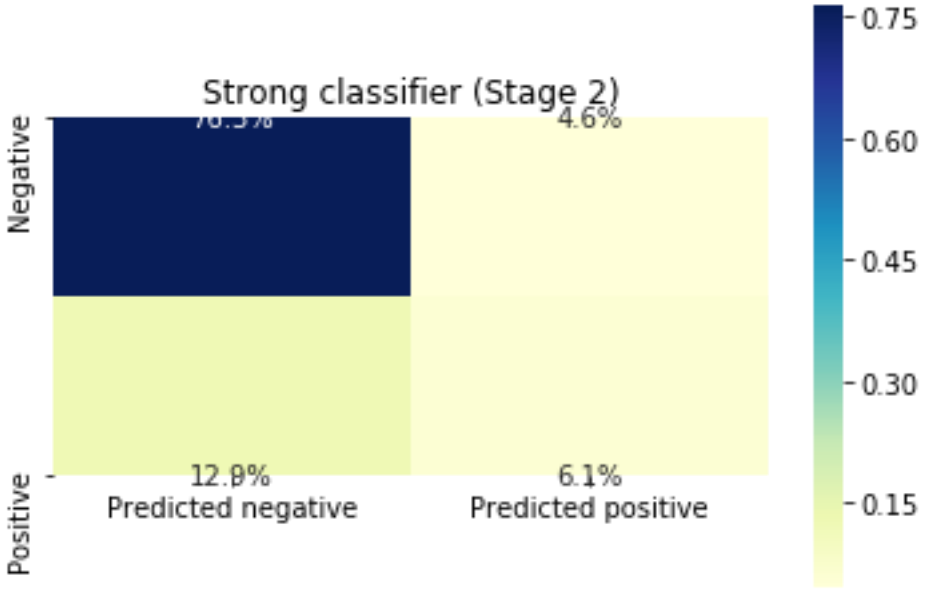

Figure 6: Heatmap showing the accuracy parameters of 3 rounds of AdaBoost

- AdaBoost rounds: **5**
  - Total accuracy: 0.66
  - False Positive: 0.02
  - False Negative: 0.86

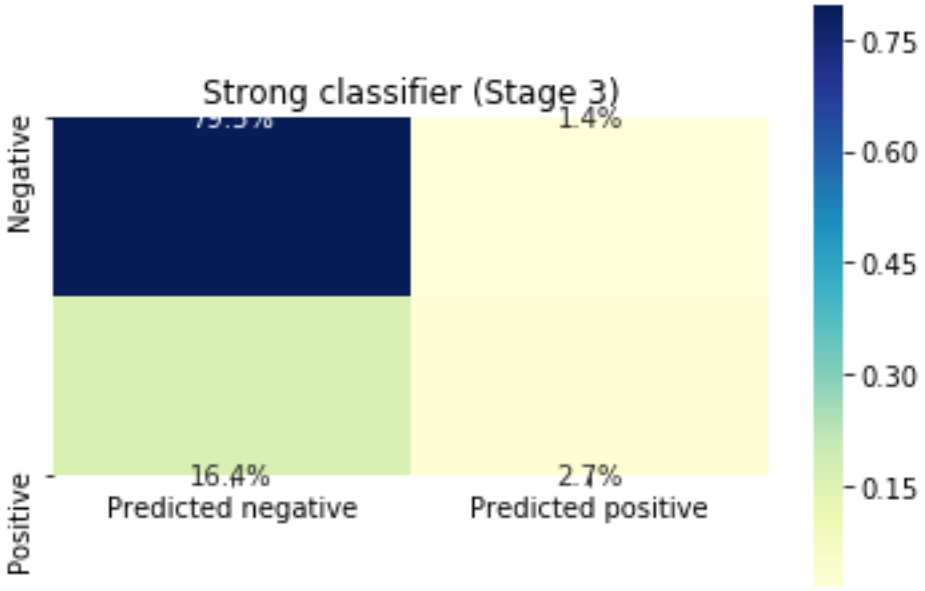

Figure 7: Heatmap showing the accuracy parameters of 5 rounds of AdaBoost

- AdaBoost rounds: **10**
  - Total accuracy: 0.80
  - False Positive: 0.01
  - False Negative: 0.88

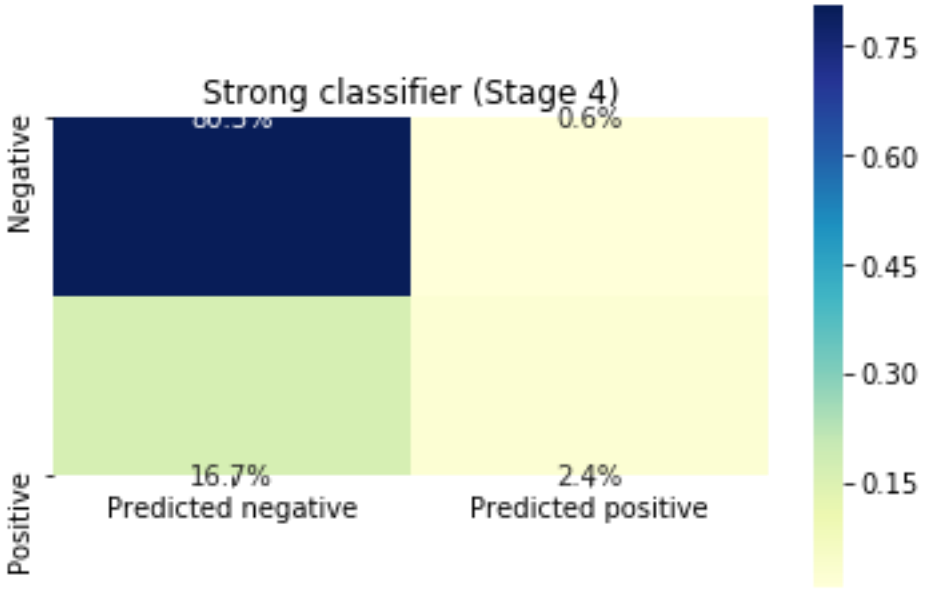

Figure 8: Heatmap showing the accuracy parameters of 10 rounds of AdaBoost

Observation:

As the number of AdaBoost rounds increases, the value of:

- accuracy increase
- false-positive rate decreases
- false-negative rate increases

2.3 Adjust the threshold

Criterion	Total Accuracy	False Positive	False Negative
Empirical Error**	**0.66**	**0.02**	**0.86**
False Positive	**0.37**	**0.13**	**0.68**
False Negative	**0.36**	**0.10**	**0.76**

Table 1: Adjusting the threshold

**** The corresponding values for this criterion are taken from the AdaBoost 5 rounds test run which was done in the previous section.

2.4 Additional Part

2.4.2 Detecting in real-world photos

A four-stage cascade has been used to build a classifier to perform face detection on images. Given below are the images which have undergone 4-layer cascade training with 10 rounds of AdaBoost. The performance of the classifier increases with the number of AdaBoost rounds. The maximum number of rounds implemented in this project is 10 and the classifier gave a mediocre result. However, the classifier will give a better output as we increase the number of rounds which will be computationally expensive and more time-consuming.

- Baelish

Figure 9:Processed image of baelish

- Tesla

Figure 10: Processed image of tesla

- Land

Figure 11: Processed image of land (NO FACE DETECTED)

- Groupe2

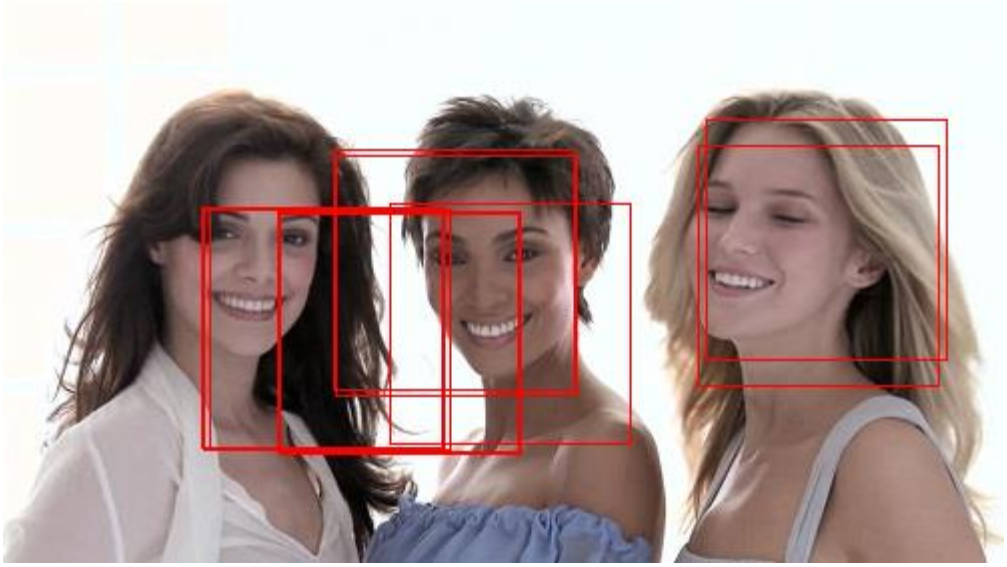

Figure 12: Processed image of groupe2

- Visages

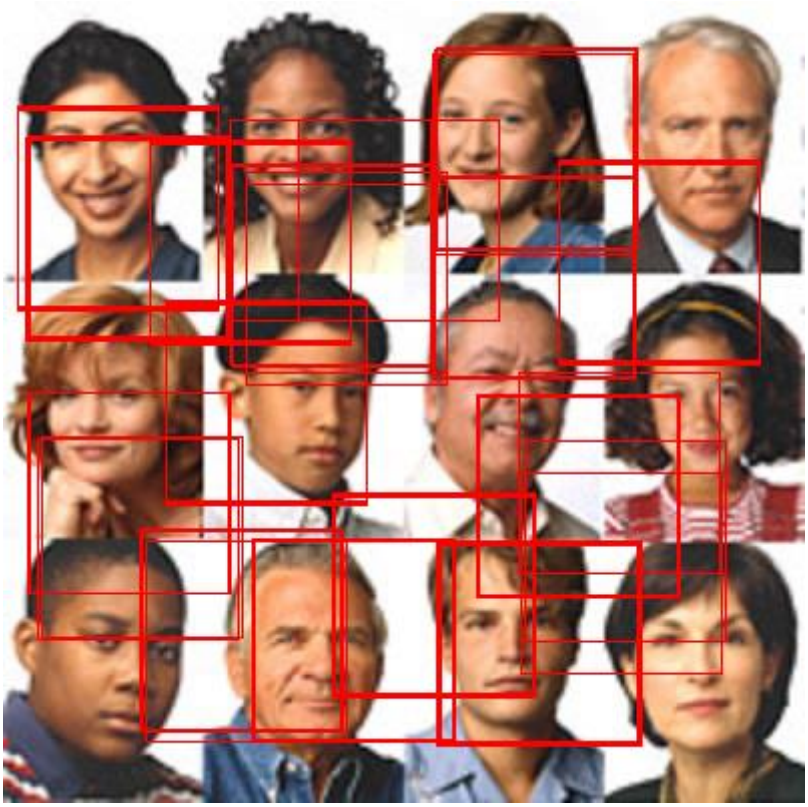

Figure 13: Processed image of visages

Observation:

In the original paper, a 38-layer cascaded classifier was trained as compared to a 4-layer cascaded classifier we used in our implementation. The classifier performed okay for images with single and no faces but it performed slightly poor on images with multiple faces. From the Jupyter notebook (face-detector.ipynb), we can also observe that the trained cascade performs better as we increase the number of layers.