CSE 573
Project III

Ateendra Ramesh
ateendra@buffalo.edu
Person # - 50290770

May 17, 2019

1 Objective

This project focuses on detecting faces using the Viola Jones algorithm.

2 Viola Jones Algorithm

This algorithm is one of the most powerful face detection algorithm and give extremely good performance, despite recent development by deep learning algorithms. With large amounts of data, the algorithm was trained extensively.

This uses the feature extraction and the AdaBoost algorithm to create cascades of classifiers to reduce false positives. This algorithm further ensures that the training error goes to zero quickly as the model's weights are iteratively re-weighted to ensure that the next classifier is able to make more accurate predictions. The AdaBoost mechanism can be seen in Figure 1 and the cascade algorithm can be seen in Figure 2.

3 Algorithms

This section briefly elaborates the algorithms used for the face detection task framework in this project.

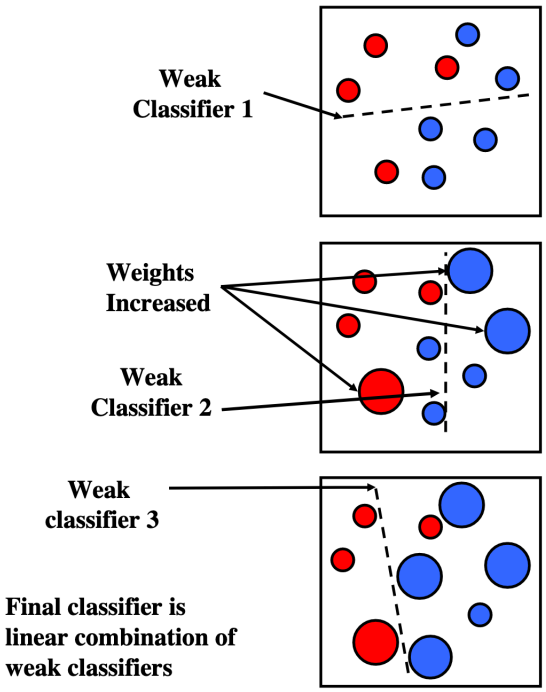

Figure 1: AdaBoost Algorithm - Iterative Reweighting

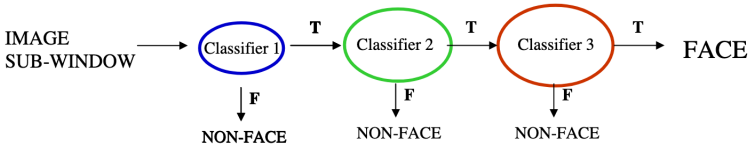

Figure 2: Cascade Algorithm

3.1 Integral Image

The integral image is computed using the cumulative sum function in the `numpy` package. This is very similar to summed area tables.

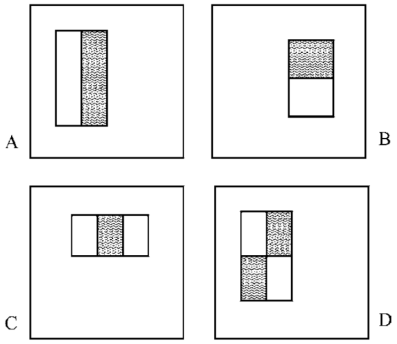

Figure 3: Haar Feature Visualization.

3.2 Haar Features

The features are built upon the integral image features, where in, the regions are proportioned into 2, 3, and 4 rectangle features. 5 types of boxes were extracted for each of the image patches. For a single 24×24 image patch, it results close to 160k features. It is shown in Figure 3, except, the vertical 3-rectangle features were also used.

3.3 Feature Selection - AdaBoost

In this algorithm, the AdaBoost algorithm mentioned in the paper is followed. The steps of that process are shown below.

1. After features are extracted, a weight is assigned to each of the training examples, i.e., $\frac{1}{2l}$ and $\frac{1}{2m}$ for the the positive and negative samples respectively. Here, l is the number of postive examples, and n is the number of negative examples.

$$w_i = \begin{cases} \frac{1}{2l} & if x = 1 \\ \frac{1}{2m} & if x = 0 \end{cases} \tag{1}$$

2. For each feature f, the algorithm checks whether this feature makes the best split and registers the error caused by every feature.

3. When choosing the feature f, the threshold (θ) and the polarity value (p). This is done by going through all the examples, and choosing a

value such that this error is minimized in Equation 2.

$$e = \min\left(S^+ + T^- - S^-, S^- + T^+ - S^+\right) \tag{2}$$

Here the T refers to the total sum of positive and negative weights. The resulting minimal error would be chosen and the corresponding feature value will be set as threshold and the polarity will be set to -1 if the majority class is face and +1 if otherwise.

4. Using the value of θ and p, the classifier is formed with the following equation.

$$h(x, f, p, \theta) = \begin{cases} 1 & if pf(x) < p\theta \\ 0 & otherwise \end{cases} \tag{3}$$

5. Upon choosing a feature, its importance α was computed by first determining β based on the error (e) the feature had made using the equation below.

$$\beta = \frac{e}{1-e} \tag{4}$$

$$\alpha = ln(\frac{1}{\beta}) \tag{5}$$

6. Upon building the above steps T classifiers would be built to get a strong classifier.

For making a prediction with a strong classifier, the following equation was used.

$$C(x) = \begin{cases} 1 & if \sum_{t=1}^{T} \alpha_t h_t(x) > \frac{1}{2} \sum_{t=1}^{T} \alpha_t \\ 0 & otherwise \end{cases} \tag{6}$$

4 Methodology

1. Read an image from the dataset.
2. Extract Haar features using the integral image from the read image.
3. Using AdaBoost, compute the T classifiers needed and assemble a strong classifier from Equation 6.
4. Build multiple strong classifiers for cascades.

5 Data

In this project, the CBCL dataset and parts of the FDDB was used for training. And the results can be seen below. 2 strong classifiers were trained and used for classification of faces.

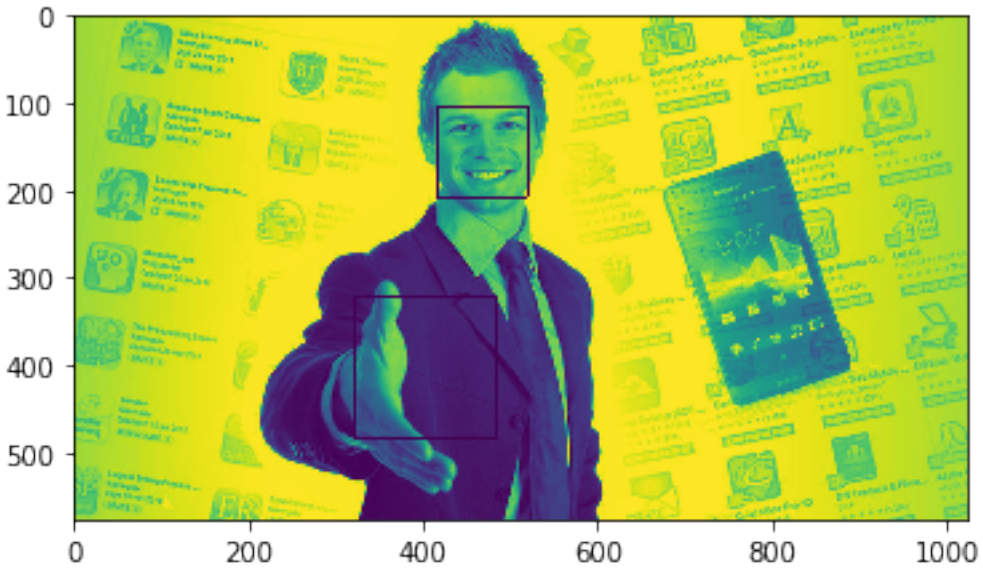

Figure 4: Detected Faces in Test Data

6 Testing

While testing, the feature were re-sized into 15 scales, 1.5 factors apart. And for non-faces are penalized more to further reduce false positives.

$$h_{test}(x, f, p, \theta) = \begin{cases} 1 & if pf(x) < p\theta \\ -0.8 & otherwise \end{cases} \tag{7}$$

7 Result Analysis

There were many false positives and the penalization in testing caused faces to be filtered. The scaling was not sufficient to detect large faces such as

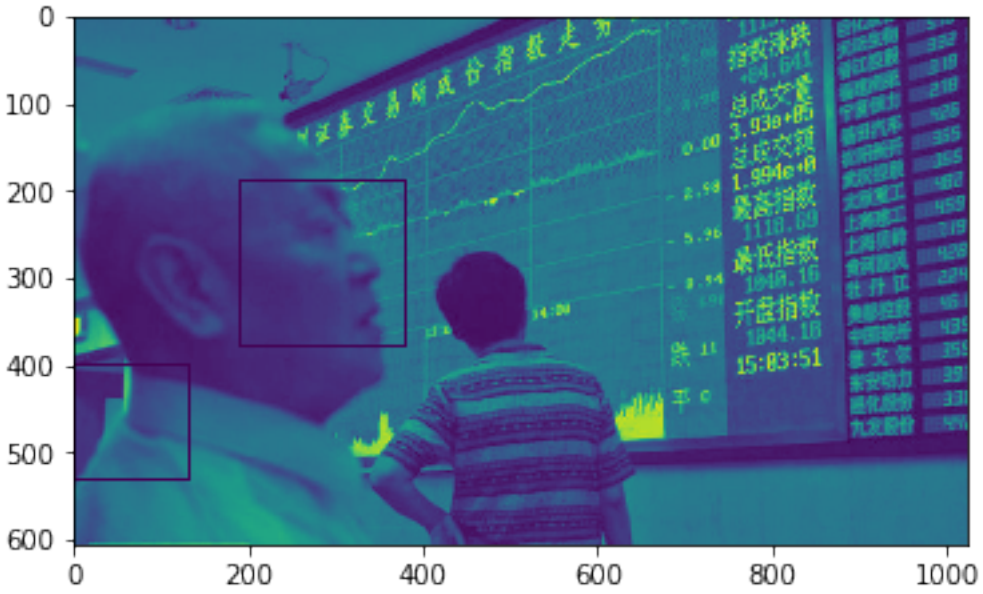

Figure 5: Detected Faces in Test Data

the one in Figure 4. But, the time taken for computation was extremely large due to lack of parallelization of the code, which is scope for further improvement.

8 Scope for Improvement

1. More training data and parallelized code for faster execution as training the above models took 10 hours.

2. More layers of cascade would be helpful in removing false positives.

References

[1] Medium Article by DataDrivenInvestor on the Viola Jones Algorithm - `https://medium.com/datadriveninvestor/understanding-and-implementing-the-viola-jones-image-classification-algorithm-`

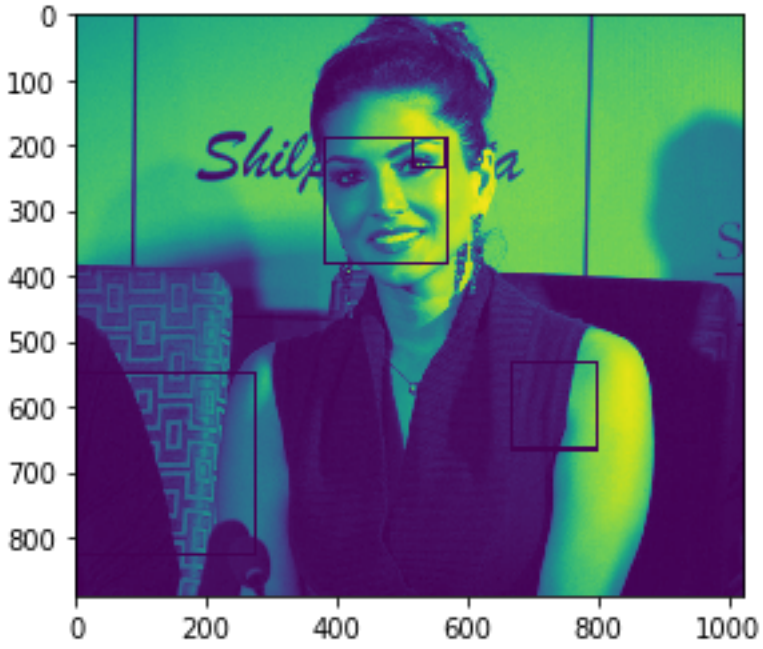

Figure 6: Detected Faces in Test Data

[2] Viola, Paul, and Michael Jones. "Rapid object detection using a boosted cascade of simple features." CVPR (1) 1 (2001): 511-518.

[3] Viola, Paul, and Michael J. Jones. "Robust real-time face detection." International journal of computer vision 57.2 (2004): 137-154.

[4] Viola–Jones object detection framework - `https://en.wikipedia.org/wiki/Viola%E2%80%93Jones_object_detection_framework`

[5] The Viola/Jones Face Detector - `https://www.cs.ubc.ca/~lowe/425/slides/13-ViolaJones.pdf`

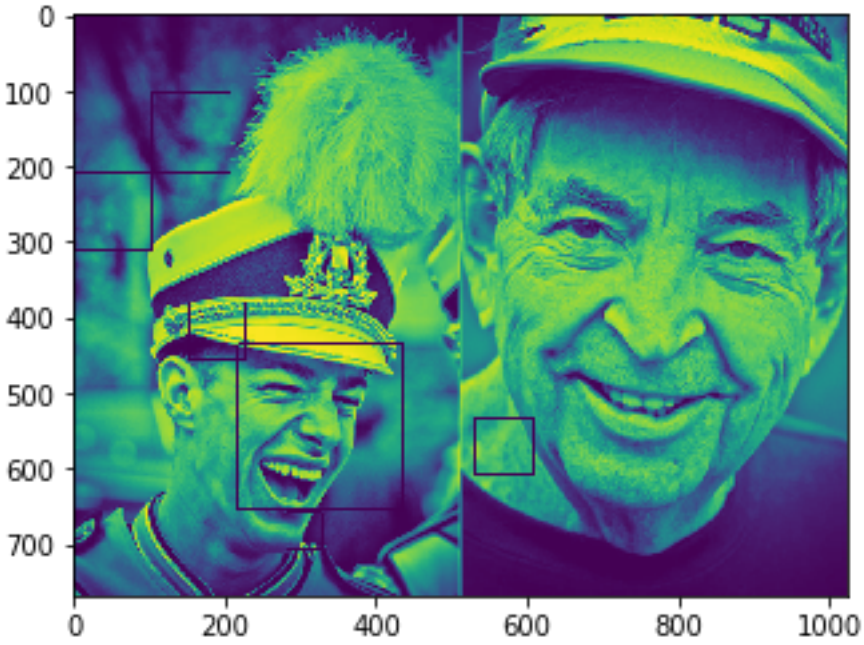

Figure 7: Detected Faces in Test Data

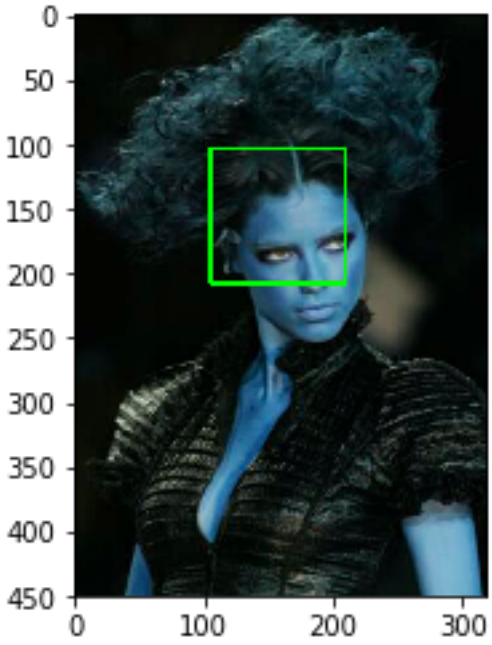

Figure 8: Detected faces in FDDB data

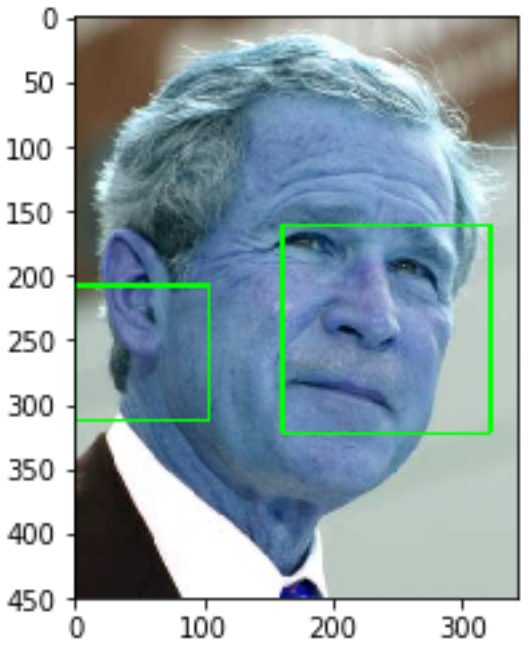

Figure 9: Detected faces in FDDB data