Introduction

Boosting technique is the effective method of producing an accurate classifier by combining rough and inaccurate rules. To begin boosting, defining a weak learner is the first step. The boosting algorithm repeatedly call the weak learner, each time feeding it a different distribution over the training data. Each iteration generates a weak classifier and combine all of these single classifiers with

Adaptive Boosting is one of the practical boosting algorithms focusing on classification problems and improve a set of weak classifiers into a strong one. It is named AdaBoost algorithm. Before the implementation of the AdaBoost algorithm, selecting weak classifier is the first step. The weak leaner performance is expected to be a little better than random, so it is desired to get error rate less than 1/2.

Methodology

A. Dataset preparation

The image resolution 20 by 20, normalized gray scale [0,1] images are used. 1000 face and 1000 nonface images stored in the dataset. 9% of them split into test set. For face detection, the Haar-like feature extraction is used. There are different Haar-like features (Figure. 1) are available for implementation. Most likely ‘type-2-y’ and ‘type-3-x’ are particularly having high performance which is mentioned in the reference [1]. To reduce time consumption for testing all available Haar-like features, minimized into ‘type-2-y’ and ‘type-3-x’.

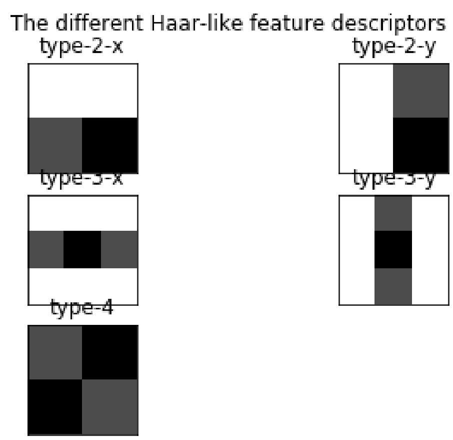

Figure 1. the example of Haar-like feature

B. Haar-like feature extraction

Viola-Jones algorithm uses sub-window of images and tries to detect relevant features i.e., Haar-Features (let’s call Haar features as patterns to avoid confusion with feature values extracted from the images). One of pattern could have more than 20,900 sub-window of an images which harr-like feature could be. The two patterns have been used in this project, ‘type-2-y’ and type-3-x’, which includes 34,067.

[1] p.Viola, M.Jones, “Rapid Object Detection using a Boosted Cascade of Simple Features”. CVPR 2001

C. Adaptive boosting

To select the best single feature that best separates face and nonface, for each features the weak learner determines the threshold classification function, such that the minimum number of training sets are misclassified. The number of features (in case of the setting, 34,067) are extremely high cost of computation, it requires to adapt an extra boosting technique to get optimum threshold for weak learner. The objective function is $h_t = argmin(\epsilon_t(h))$. For a decision stump, project is used a decision tree before boosting. Based on the decision prediction, weak learner is selected, and the weight is updated through the iteration of boosting algorithm. The number of iterations was tested from 3 to 10. Above the certain number of iterations, the errors became as low as 0.

Result

Table 1. 3 iteration error values

# of iteration	Error	Alpha
1	0.291	0.44
2	0.208	0.67
3	0.12	1.0

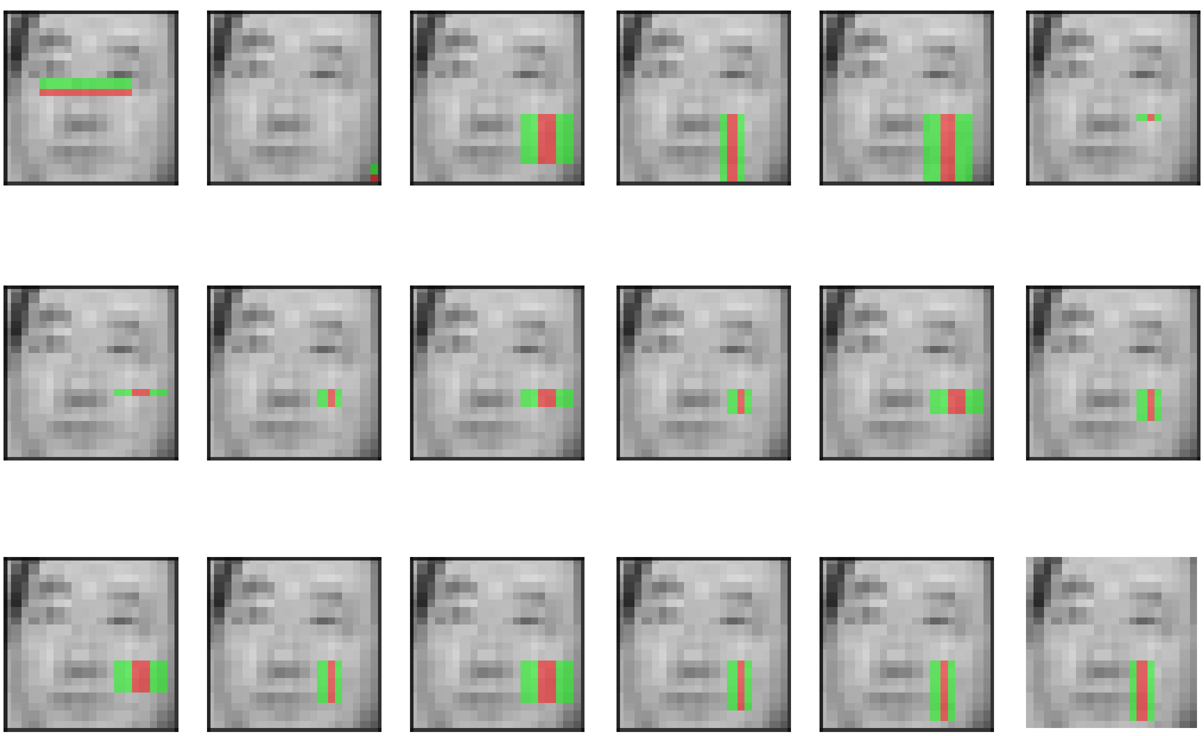

Figure 2. 3- iteration the most important features

[1] p.Viola, M.Jones, "Rapid Object Detection using a Boosted Cascade of Simple Features". CVPR 2001

Table 2. 5 iteration error values

# of iteration	Error	Alpha
1	0.296	0.43
2	0.214	0.65
3	0.125	0.97
4	0.051	1.46
5	0.012	2.20

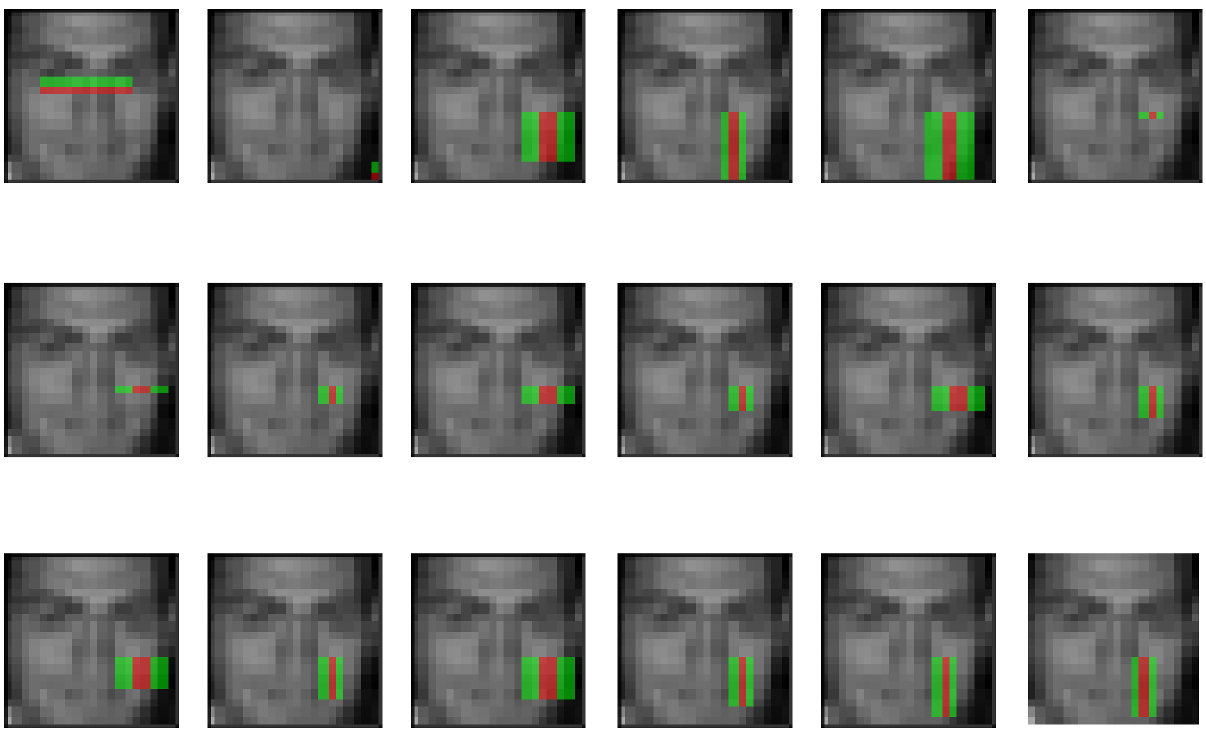

Figure 3. 5- iteration the most important features

[1] p.Viola, M.Jones, "Rapid Object Detection using a Boosted Cascade of Simple Features". CVPR 2001

Table 3. 10 iteration error values

# of iteration	Error	Alpha
1	0.391	0.443
2	0.209	0.665
3	0.11	0.99
4	0.047	1.496
5	0.0011	2.24
6	0.00004	5.05
7	2.6e-07	7.57
8	1.34e-10	11.3
9	1.56e-15	17.04

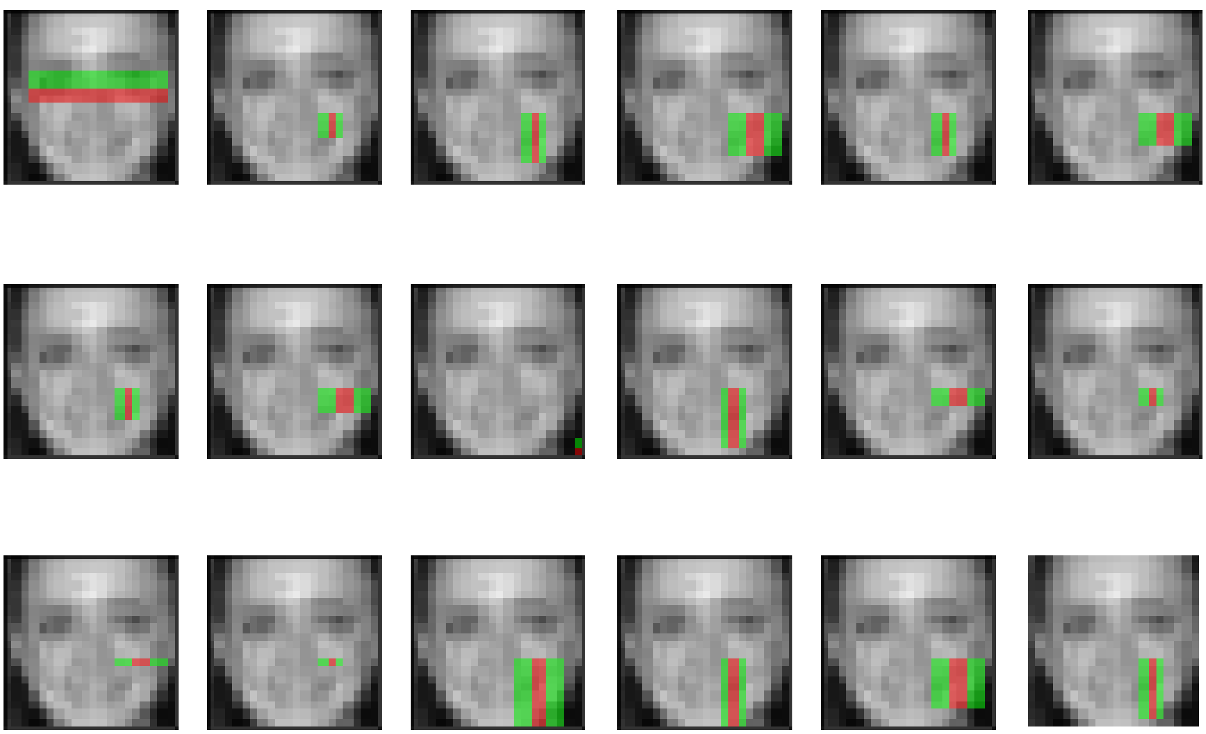

Figure 4. 10- iteration the most important features

[1] p.Viola, M.Jones, "Rapid Object Detection using a Boosted Cascade of Simple Features". CVPR 2001

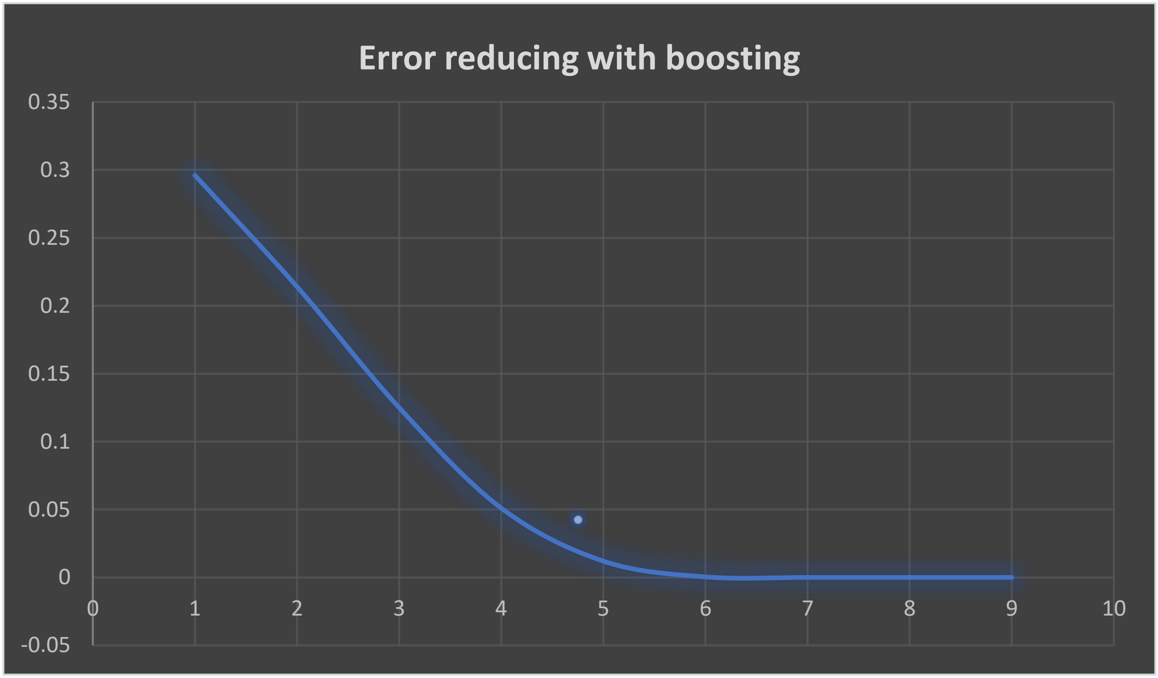

Figure 5. Error value trend based on the number of iterations

Conclusion

As we have seen in the result section, the most importance feature in face detection is selected the one in below. The features around eyes with haar-like pattern 'type-2-y'.

Figure 6. The most important features

The results from my code is slightly different from Viola. The nose seems detecting better with type-2-x as shown in figure 7. In addition to that, the face detection features, not only located on the face, but also feature detected the edge of the part of the face as well as the face edge. The reason is that it has big contraction on the boundary between those. Based on this founding, we could use Harr-like feature also for edge.

[1] p.Viola, M.Jones, "Rapid Object Detection using a Boosted Cascade of Simple Features". CVPR 2001

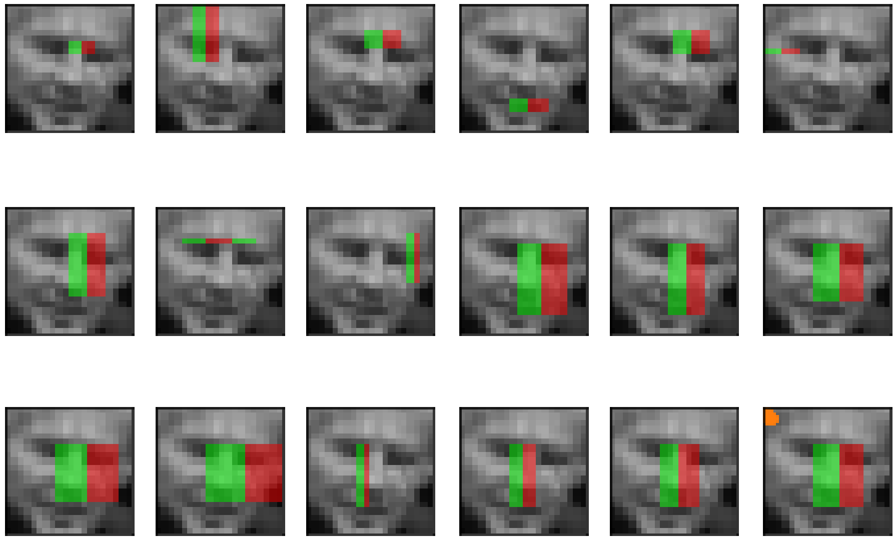

Figure 7. 'type-2-x' feature detection

The evaluation of the classifier is used Receiver Operating Characteristic (ROC) metric. ROC curve typically true positive rate on Y- axis and false positive rate on the X axis. The top left corner of the plot is the best which has zero false positive and true positive rate as one. The Adaboost algorithm results of face detection score is around 69 to 70% and ROC curve is as shown below.

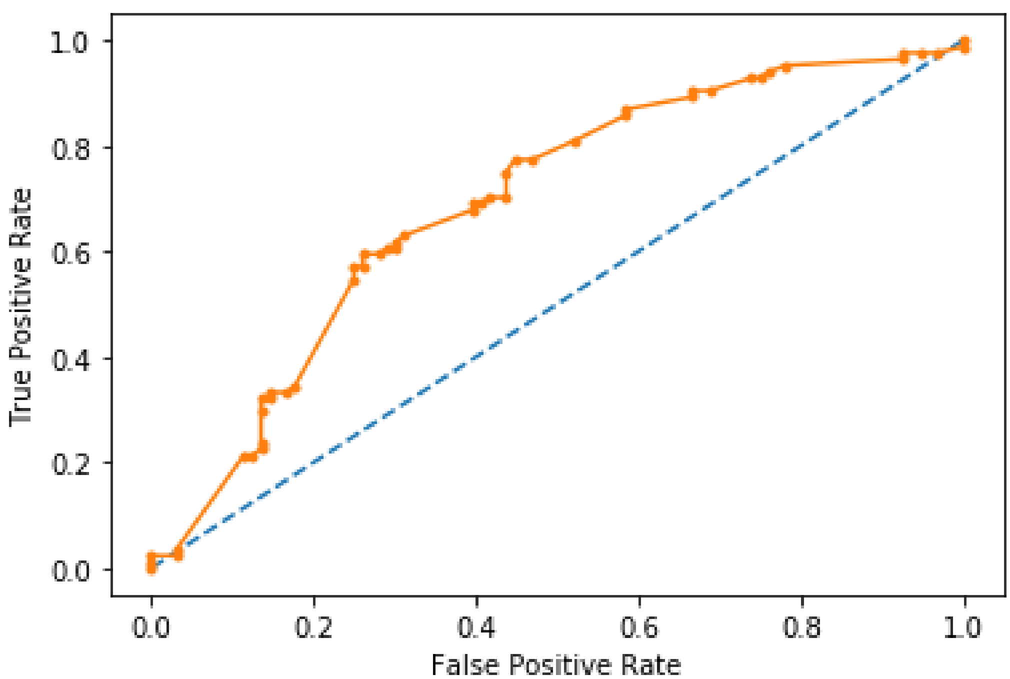

Figure 8. ROC curve with iteration 10

[1] p.Viola, M.Jones, "Rapid Object Detection using a Boosted Cascade of Simple Features". CVPR 2001