Assignment 5 Boosting

Yonglin ZHUO: yonzh457
Andreea IOANA: andbo713

1 PLOT THE MOST IMPORTANT HAAR FEATURES

Plots of the most important Haar features chosen by AdaBoost and some thoughts why these might be important for the classification.

At first, when we construct the training set and testing, we need to use the Haar features to extract the features of the original data. It should be some particular features are more important.
Secondly,as it known, at the output step of the AdaBoost algorithm, we need to use the formula(1.1) to get the final estimate. In our lab, we use each Haar Feature to get each weak learner. The learning rule is so simple. For example, 4 Haar features have chosen and the learning rule for the first weak leaner. Then if the first coordinate of one feature vector is less than a specific threshold, then the output should be 1,otherwise, -1. As a consequence, the weight α_t of each weak leaner stands for the importance of each Haar Feature. The higher the weight of weak learner is, the more important the corresponding feature is. *Figure1.1* shows the top 10 important Haar features when we pick up 100 Haar features for one implementation with 6000 training and 3000 testing examples. In this particular experiment, we use all of the 100 features to learn 100 weak learners. Actually it has considerately low error rate of 0.0623 on testing set
Figure 1.1 shows us 10 Haar features whcih have the top 10 weights. And their weights are (0.6807 0.3746 0.3369 0.2816 0.2754 0.2658 0.2617 0.2573 0.2492 0.2431).

$$H(x) = sign\{\sum_{i=1}^{T} \alpha_t H_t(x)\} \tag{1.1}$$

Figure 1.1: Top 10 Haar Features

2 Plot the misclassified

Plots of some misclassified faces and non-faces that seem hard to classify correctly.

Figure 2.1 shows the pictures that cannot be classified in the experiment which has 100 features, 100 weak classifiers, 6000 training points, and 3000 testing points. In this case, there are 187 photos misclassified on the testing data set.

Figure 2.1 shows the pictures that cannot be classified in the experiment which has 100 features, 100 weak classifiers, 2000 training points, and 1000 testing points. In this case, there are 77 photos misclassified on the testing data set.

As we can see that the faces photos which are hard to distinguish seem to have lower contrast ratio. They tend to be too bright or too dark, which might be difficult for extracting the features. At the same time, it seems not to be so easy to recognize the nonface pictures which has better contrast ratio.

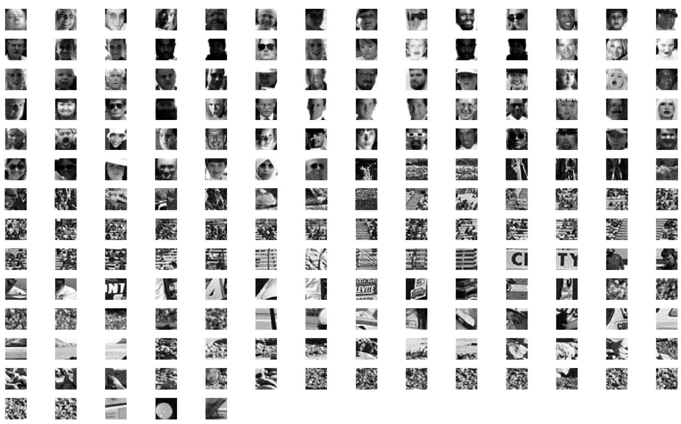

Figure 2.1: All the misclassified photos of the experiment for 6000 training and 3000 testing samples

Figure 2.2: All the misclassified photos of the experiment for 2000 training and 1000 testing samples

3 Plot of the classification accuracy

Plot of how the classification accuracy on training and test data depends on the number of weak classifiers. What was the best accuracy you could achieve and how?

Table 3.1 and *Figure 3.1* show that the misclassification rate of model which has 75 features with the increasing of weak learners. On both testing and training set, you can see that the error rate tends to decrease with the number of weak learners going up. But it seems that the error rate on testing set goes down when the number of weak learners is less than 65. I think it might because of the overfitting. The model with 65 weak learners has the lowest testing error rate of 0.094

Table 3.2 and *Figure 3.2* show that the misclassification rate of model which has 100 features with the increasing of weak learners. On both testing and training set, you can see that the error rate tends to declines with the number of weak learners increasing. And the overall leaner with 100 weak learners has the best testing performance with the error rate of 0.077. Compared with these tables, you can find that if we expect a lower error rate, we could make efforts via two methods.

1. Increase the number of weak learner

2. Increase the number of features. For equal number of weak learners.

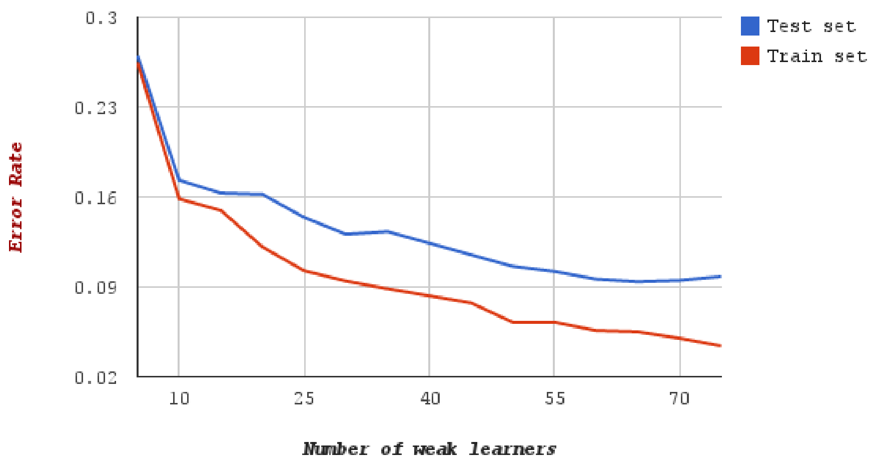

Figure 3.1: Error plot for model of 75 features

Table 3.1: Error rate table for model of 75 features

nbr of weak learners	*Test set*	*Train set*
5	*0.27*	*0.2645*
10	*0.173*	*0.1585*
15	*0.163*	*0.1495*
20	*0.162*	*0.121*
25	*0.144*	*0.1025*
30	*0.131*	*0.0945*
35	*0.133*	*0.0885*
40	*0.124*	*0.083*
45	*0.115*	*0.0775*
50	*0.106*	*0.0625*
55	*0.102*	*0.0625*
60	*0.096*	*0.056*
65	*0.094*	*0.055*
70	*0.095*	*0.05*
75	*0.098*	*0.044*

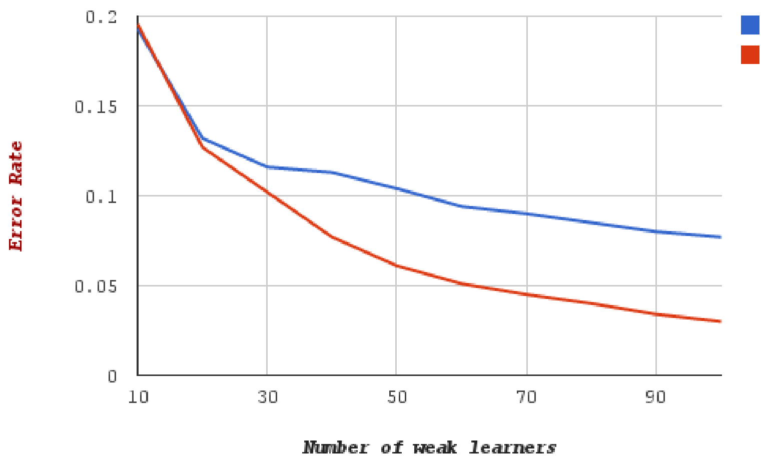

Figure 3.2: Error plot for model of 100 features

Table 3.2: Error rate table for model of 100 features

nbr of weak learners	*Test set*	*Train set*
10	*0.193*	*0.1955*
20	*0.132*	*0.127*
30	*0.116*	*0.102*
40	*0.113*	*0.077*
50	*0.104*	*0.061*
60	*0.094*	*0.051*
70	*0.09*	*0.045*
80	*0.085*	*0.04*
90	*0.08*	*0.034*
100	*0.077*	*0.03*

4 Defend your results

Normally, the adaBoost algorithm has a good performance on face recognition in our experiment. You can find that models with more features and more weak learners tends to have higher accuracy. It is reasonable becuase more features means that you can extract more inforamtion of the original pictutes. And more weak learners will help you overcome more aspects of the feature space. Unfortunately, it cost almost unacceptable computing time when I tried to get a model with 100 features and 100 weak learners on a training set with 6000 observations. In the feature, we might improve the computation efficiency by two tricks

1. Instead of Brutal Force method, we need to use more effient technique to find a best weak learner

2. Use some Dimensionality approaches like PAC to compress our feature space