Boosted Classifier for Car Detection

David C. Lee

ECE Department
Carnegie Mellon University
U.S.A

Abstract

Recently, Viola and Jones [1] have proposed a detector using Adaboost to select and combine weak classifiers from a very large pool of weak classifiers, and it has been proven to be very successful for detecting faces. We have followed their approach and applied it to detect rear views of cars. The detector was carefully examined and was expanded in a number of ways, such as varying the type and complexity of weak learners, using Real Adaboost, fitting parametric functions to the probability distributions, aligning training images at different positions, and exploiting a tendency in the classifier to speed up the running time.

1 Introduction

We adopted the methods from face detection and applied it for car rear-view detection. We believe that face detection and car detection are similar in nature and applying face detection methods for car detection is a natural choice. The success of face detection may have been due to the fact that face images have rich internal features. Some image categories, such as pedestrian, can only be characterized by the contour of the object since there is not a characteristic pattern that stands out inside the region of the object. On the other hand, faces have very distinctive pattern, caused by eyes, nose, mouth, etc, that can be well captured by simple filters encoding the intensity difference as in Viola and Jones [1]. Similarly, rear views of cars also have distinctive patterns, such as the dark shadow region right below the car, and dark tire region, as shown in section 3.1. Therefore, we decided to apply the methods developed for face detection to car detection.

2 Previous work

2.1 Adaboost / Viola Jones face detector

Adaboost [3] is a meta algorithm that is designed to boost the performance of any existing classifier. A popular choice of weak learner is decision tree of depth 1, which is simply a classifier that depends only on a single feature. When the number of features is very large, as in the case of rectangular features in [1], Adaboost can be viewed as a feature selection procedure. More than 45,000 features were tested in [1] and more than 140,000 features were tested in our work, but the discriminative features are only a very small fraction of it. Finding and computing only the discriminative ones is far more efficient than attempting to evaluate them all.

The biggest contribution of Viola et al. [1] was in improving the speed of the detector to real time. This was possible due to some clever observations: evaluating only the necessary features chosen by Adaboost, using cascading structure to quickly reject negative samples, and using very simple features that are fast to evaluate using integral image.

Reducing the time spent on classifying negative samples is the very important in reducing the time to evaluate an image, since a typical image has around 1~10 cars and 130,000 sub-windows of non-car. Viola et al. have used a cascade style detector that could quickly reject negative samples by evaluating only a few features. It is mentioned in their paper that cascading improves the running time by 10 times while slightly decreasing the accuracy. A less known benefit of cascading is that it effectively uses more training data (more negative samples in particular) than one would normally be limited by the time and computational resource needed for training. Each stage of cascade only trains on the samples that passed the previous stages, and since only a very small fraction of the negative samples pass through the previous stages, each stage can train on a small sample while having the effect of training on a much larger set.

(a)

Figure 1 (a) Images collected while driving around Pittsburgh (b) cropped regions of cars (c) cropped regions of non-cars

2.2 Extensions on Viola-Jones

After the success of Viola et al. [1], there has been a lot of extensions to their method, such as extension of the feature set by Lienhart et al. [4], extensions on the weak classifier and boosting method by Wu et al. [2], a tree structured cascade for multi view face detection by Huang et al. [5], and so on. In this paper, we have also examined and expanded the original work of Viola et al. [1]

3 Own work

3.1 Dataset

The images of cars were collected inside a car from the passenger seat while driving around the Pittsburgh area. Figure 1 (a) shows some of the images that were collected, figure 1 (b) shows cropped region of cars, and figure 1 (c) shows cropped regions used for negative training images. A total of 621 images of cars were collected and 80 images were set aside for the final testing. Of the remaining 561 images, 629 cropped regions of cars were extracted and they were randomly divided into 300 for training and 329 for testing for use in comparing different methods and plotting the Receiver Operating Characteristic (ROC) curve.

Figure 2 shows the first 3 filters chosen by the boosting process. The shadow below the car is the most discriminative feature, and the left and right tire region are the second and third most discriminative feature.

The quality of the detector was measured on the 329 images reserved for testing. On the 640x480 images, a scanning window of size 50x50 was evaluated with sliding stride of 3 pixels and repeating the process by resizing the image with scale 0.9, leading to 128918 sub windows. Performance was measured by comparing the ROC curve.

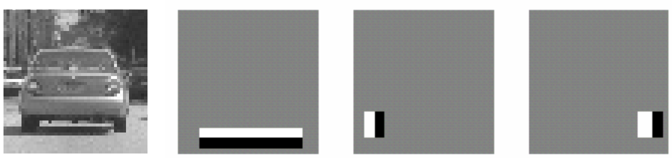

Figure 2 First three filters selected by Adaboost

3.2 Real Adaboost

- Given example images $(x_1, y_1), \ldots, (x_n, y_n)$ where $y_i = -1, 1$ for negative and positive examples respectively.
- Initialize weights $w_i = \frac{1}{2m}, \frac{1}{2l}$ for $y_i = -1, 1$ respectively, where m and l are the number of negative and positives respectively
- for $t = 1, \ldots, T$
  - For each feature j
    - a real valued weak learner is defined as $h_j = \frac{1}{2}\log\left(\frac{p_{t,j}}{n_{t,j}}\right)$ where $p_{t,j}$ and $n_{t,j}$ is the weighted histogram of feature j for positive and negative respectively weighted with w_i
  - Choose the classifier h_t with the lowest error ε_t where error is computed by thresholding the real-valued confidence at zero and classifying as positive when greater than zero and classifying as negative when less than zero.
  - Update and normalize the weights

$$w_i \leftarrow w_i \exp(-y_i h_t(x_i))$$

$$w_i \leftarrow \frac{w_i}{\sum_{j=1}^{n} w_j}$$

- The final strong classifier is

$$h(x) = \sum_{t=1}^{T} h_t(x)$$

Figure 3 Real Adaboost Algorithm

Viola et al. [1] have used discrete Adaboost and decision stump (single level decision tree) as the weak classifier. We have compared two versions of discrete Adaboost and also compared Real adaboost. The three settings were (1) a decision stump, (2) a slightly more complex binary classifier, and (3) Real Adaboost and they are depicted in Figure 4. The first two methods with binary classifier are trained with discrete Adaboost and the third method was trained with real Adaboost. The algorithm for method (3) is given in Figure 3. Experiments show (Figure 5) that method (3) gives the best performance and method (2) and (1) gives slightly lower performance. Even in the second and third case, no compromise in speed needs to be made if the weak classifier is stored as a look up table [2].

We have also experimented with weak classifier using 2 features to create a 2 dimensional histogram. This yielded in better performance in training set but gave slightly worse performance in testing set, which suggests the classifier have overfitted the training set given the power of a more complex classifier.

3.3 Fitting Parameters

Simoncelli [6] have noticed empirically that histograms of the features used in our setting follow the form:

$$h(x) \propto e^{-|x/s|^p}$$

We observed (Figure 6) that the histograms of features for negative samples follow the above form, but the histograms of positives do not (and should not, for them to be discriminative!). We can try to fit the above function when estimating $n_{t,j}$ but then we can no longer weight the histograms with w_i since the weighted histogram would no longer follow the above form. We have first experimented where the histograms $p_{t,j}$ and $n_{t,j}$ were unweighted, and then tried fitting the unweighted negative histogram. Both yielded a slightly worse performance than using weighted histograms.

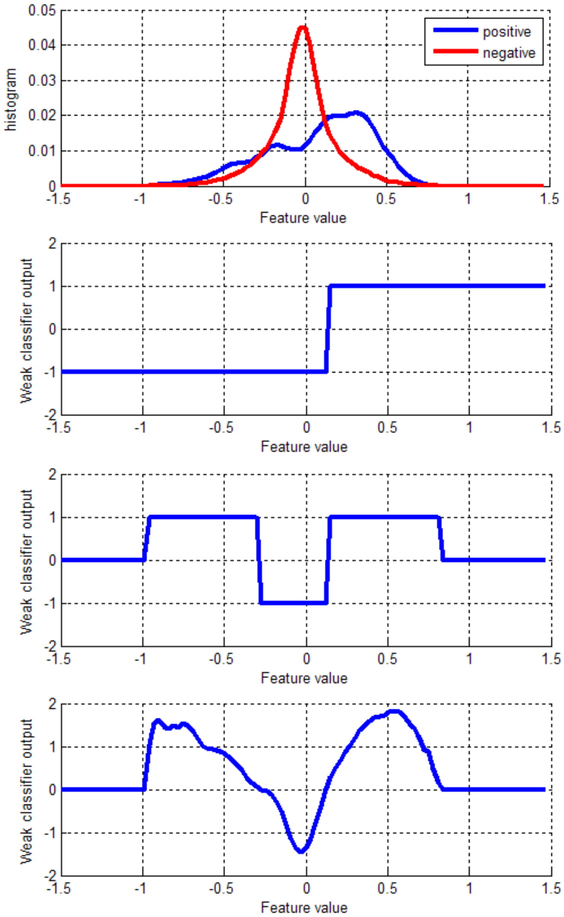

Figure 4 First row: histogram of postive and negative samples, second, third, fourth row: weak classifier for method (1), (2), (3)

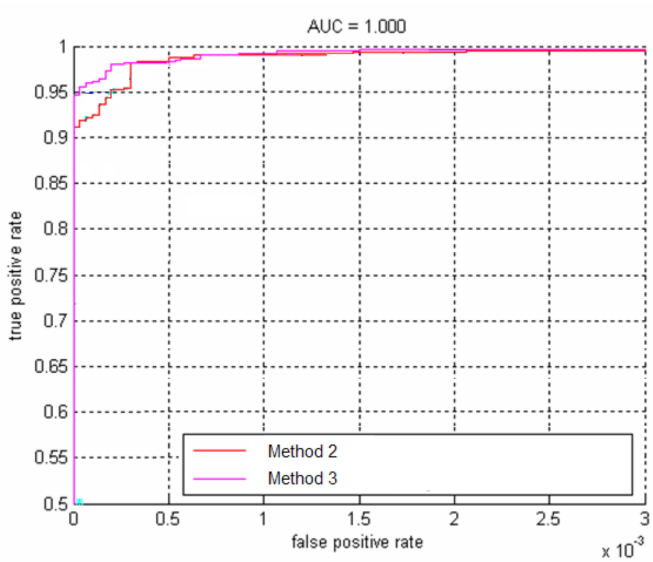

Figure 5 Performance comparison of two different weak learners

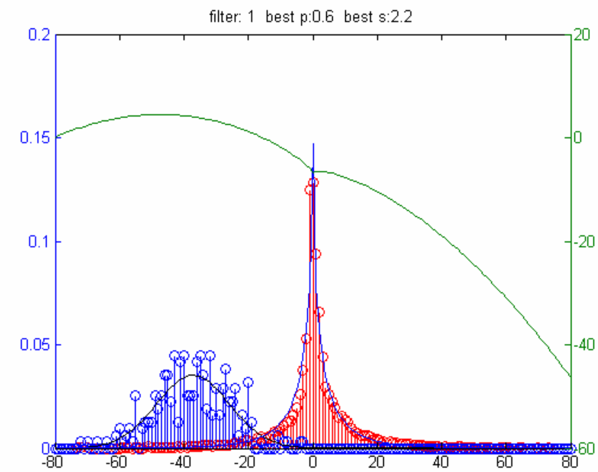

Figure 6 Fitted histogram with equation given by Simoncelli [6]

3.4 Speed up

Although we did not use the cascading structure for quick rejection of negative samples, we did observe a tendency in the classifier that could increase the speed drastically. Figure 7 shows the score of the final strong classifier plotted against the number of weak learners. As the number of weak learner increases, the score for both positive and negative samples increase or decrease almost linearly. Samples that get low score early on in the evaluation could be rejected to save time. By adjusting the threshold for rejection, we could reduce the average number of evaluated features from 500 to 7 features achieving a running time of about 100ms for a 640 x 480 image, without compromising the accuracy.

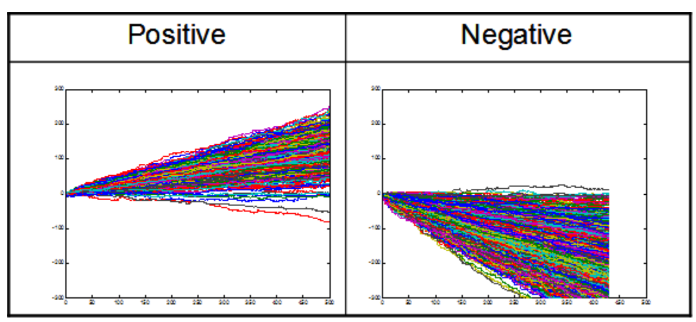

Figure 7 Tendency in the score of the final classifier

3.5 Image Alignment

As briefly mentioned in section 3.1, we have noticed that the most discriminative feature for detecting cars is the shadow region. There may be two reasons for this; the region may actually be the most characteristic part in detecting cars, or it might have been caused by the fact that the training images of cars were aligned at the tips of the two tires, thus making the bottom region more consistent across car images.

This brings us to the question of how we should align the car images. So we have compared two different alignments, one aligned with the two tips of the tires and the other aligned the top two corners of the car. Figure 8 compares the first 10 features chosen by the process, and it shows that more features

Figure 10 The first row and the left most image of second row shows results with no errors. The three rightmost images on the second row show examples where there are either false detections or missed detection.

Figure 8 Two ways to align images of cars

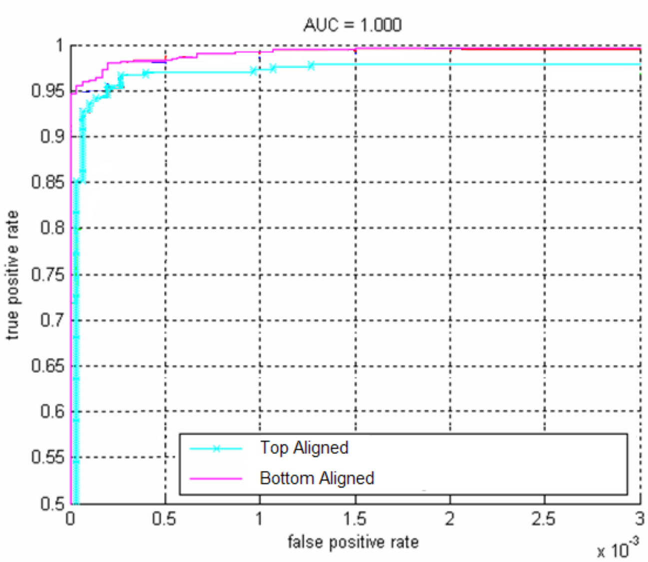

Figure 9 Comparing the performance of two alignment methods

were chosen from the upper region when the images are aligned at the top. Figure 9 compares the accuracy of the two alignments and shows that aligning at the bottom gives a better detector. This shows that careful alignment does affect the performance and

4 Conclusion

We have carefully studied and observed the face detection method developed by Viola et al. [1] and applied it to car detection. The final results with the best combination of explored methods are shown in Figure 10. On the 80 images set aside for the final testing, there were a total of 149 cars and 143 of them were correctly detected (96.0%) and there were 24 false positives (0.3 fp/image).

References

[1] Paul Viola and Michael Jones, "Robust Real-time Object Detection" *International Journal of Computer Vision* (IJCV), 2004.

[2] Bo Wu, Haizhou Ai, Chang Huang, "Fast Rotation Invariant Multi-View Face Detection based on Real Adaboost", *International Conference on Automatic Face and Gesture Recognition*, 2004

[3] Y. Freund and R. E. Schapire, "A short introduction to boosting", *Journal of Japanese Society for Artificial Intelligence*, 14(5):771-780, September, 1999

[4] Rainer Leinhart and Jochen Maydt, "An Extended Set of Haar-like Features for Rapid Object Detection," *IEEE ICIP 2002*, Vol. 1, pp. 900-903, Sep. 2002.

[5] Chang Huang, Haizhou Ai, Yuan Li, and Shihong Lao, "Vector Boosting for Rotation Invariant Multi-view Face Detection," *International Conference on Computer Vision (ICCV)*, 2005.

[6] Simoncelli, "Statistical Models for Image Compression, Restoration and Synthesis", *ACSSC*, 1997