



Prof. B. Schiele
<schiele AT mis dot tu-darmstadt dot de>

Mario Fritz
<fritz AT mis dot tu-darmstadt dot de>

Übungsblatt 4: Boosting for Detection and Evaluation of Detectors (40 points)

Ausgabe: 16.06.2007

Abgabe: 29.06.2007

As motivated in the last exercise we will now use and evaluate boosting in a detection framework. For a more efficient implementation, we switch over to a GentleBoost implementation available on the internet: <http://web.mit.edu/torralba/www/>. The task we will evaluate the method in is car detection in the UIUC single scale database: <http://l2r.cs.uiuc.edu/~cogcomp/Data/Car/>. The necessary files and data is provided on the course homepage.

1. What is different when using GentleBoost instead of standard AdaBoost?
2. As we want to use our boosted classifier for detection, we use a sliding window technique. Therefore we scan the whole image for the object of interest by sliding a fixed bounding box over it and deciding for each position if the object is present or not. Finally the output of the classifier is used as a confidence to sort the obtained object hypothesis. Download the matlab framework and data for this exercise from the course homepage.
3. Familiarize yourself with the code. `Detector.m` is the main routine and should be runnable as a script in matlab. The training set consists of bounding boxes with cars and bounding boxes without cars. The feature extraction basically average over pixel-values in 6×6 blocks and then computes derivatives between these blocks in x and y . These 214 derivatives make up the feature vector. Each dimension is afterwards used as the basis for a weakclassifier. Why do you think that these features might be a reasonable idea? In how far do you think they are superior to simply taking the pixel values. Do you think they are too weak or too strong? In what kind of problems could we run with these kind of features?
4. In order to evaluate the detector, the - so called - groundtruth for the dataset is provided in the file `groundtruht.mat`. (we use a small subset of the whole UIUC test database). Each line in the file specifies for each image where the car is located (there exactly one car per image). The 4 numbers denote: x and y coordinates of upper-left corner and x and y coordinates of lower right corner. In the following we want to evaluate and improve the detector. Therefore we have to decide which predictions of the detector were right (true positives) and which were wrong (false positives). Write a function that compares the groundtruth boundingboxes with the predicted boundingboxes of a hypothesis of the detector. In our evaluation a bounding is considered to be a true positive if the area of the intersection of the two bounding boxes divided by the area of the union of the two bounding boxes is larger than 0.5:

$$\frac{A_{\text{hypothesis}} \cap A_{\text{groundtruth}}}{A_{\text{hypothesis}} \cup A_{\text{groundtruth}}} > 0.5 \quad (1)$$

In addition, only the first detection of a particular car is considered to be a true positive. All following detections will be counted as false positive. Keep that in mind.

5. Plot the results in two fashions ROC and precision/recall. The curve in both visualization of the detector performance is parameterised by the score. That means, that first the hypotheses with the highest score are considered. As we lower the threshold step by step, we hopefully gain recall or true-positive rate (curve moves up), but might also lose precision/increase false positive rate (curve moves to the right).
 - ROC-plot: The horizontal axis shows the false positive rate, which is the number of false positives divided by total number of negatives in the test set (windows containing no car) **all**

~~windows that were test~~. The vertical axis shows the true positive rate respectively, which is the number of true positives divided by the total number of positives in the test set. This implies, that this curves will always end in the upper right corner at (1,1), because when you lower the acceptance threshold to a very small number, you will accept all hypothesis as cars. Then you surely detect all positives, but will also get all false positives. -

- Precision-Recall-plot: The horizontal axis shows the 1-precision. Precision is the number of true false-positives divided by the number of predictions (notice that the number of predictions changes in this case, as we make more predictions when we lower the threshold = walk along the curve). Recall is the number of true positives divided by the total number of positives in the test data set.
6. Evaluate the effect of the number of boosting rounds on the performance. Plot the ROC and precision-recall curves for 10, 50, 100, 200, 300, 400, 500 iterations
 7. Inspection your detections again. They might look better than what the curves look like. Why? (think of the problem of multiple detections) Propose and implement a simple method method to improve the results (remember that we have one instance per image).
 8. A very successful method to improve performance of a detector in practice is too bootstrap hard examples. This means running the detector on database different from training and test and gathering false positives. Then the detector is retrained with the original training set plus the false positives. The idea is to sample problematic examples and retrain afterwards to improve on them. As no additional data is provided, we use the test set for generating false positives for the bootstrapping. Detect on the test set and identify 100 false positives with the highest score. Add those to the training set and see if you can observe an improvement in the ROC and precision-recall plot.