ROBUST REAL-TIME FACE DETECTION BASED ON COST-SENSITIVE ADABOOST
METHOD

Yong Ma, Xiaoqing Ding

State Key Laboratory of Intelligent Technology and System,
Dept. of Electronic Engineering, Tsinghua University, Beijing 100084, P.R.China
my@ocrserv.ee.tsinghua.edu.cn

ABSTRACT
This paper presents a method of detecting faces based on Cost-Sensitive AdaBoost (CS-AdaBoost) algorithm. The two main differences between CS-AdaBoost algorithm and the naive AdaBoost are that (1) unequal initial weights are given to each training sample according to its misclassification cost, and (2) the weights are updated separately for positives and negatives at each boosting step. Due to these two variations, every stage of the face detector trained by CS-AdaBoost algorithm can more effectively focus on face samples than by the naive AdaBoost to achieve robust and high detection rate with modest false alarm rate, so that the final face detector can yield high detection rates, very low false positive rates, and robust performance. Experiments also demonstrate the effectiveness of our method.

1. INTRODUCTION
In these years, the problem of face detection has attracted much attention due to its wide applications in face recognition, human-computer interaction, etc. In images with cluttered background, appearance-based face detection methods such as view-based learning clustering [1], probabilistic estimation [2], and neural networks [3], relying on techniques of statistical analysis and machine learning to find the relevant characteristics of face and non-face patterns, generally achieve good detection performance. But most of these methods are computationally rather expensive and not appropriate for real-time applications.

Recently, Viola etc [5] propose a fast face detection approach. In their method, simple Haar-like features are extracted; face/non-face classification is done by using a cascade of successively more complex classifiers which are trained by naive AdaBoost learning algorithm. Although the performance of their face detector is pretty good, one of its problems is that the complete face detection cascade has 38 stages, which makes the structure of the detector complex and the detection speed a bit slow. This is mainly due to that the two types of misclassification errors (having a face undetected and having a false alarm) are treated equally in training the classifiers using naive AdaBoost method. So in every stage, the classifier is designed to only yield a low error rate on the whole training data, and adjustment of the threshold of the boosted classifiers can only yield high detection rate and cannot reject non-face patterns as much as possible at the same time.

In this paper we propose a new variant of AdaBoost, called Cost-Sensitive AdaBoost (CS-AdaBoost), to construct a strong face/non-face classifier from weak classifiers. In CS-AdaBoost algorithm, false negative and false positive are treated differently so that the classifier seeks to minimize the number of high cost errors and total misclassification cost. Experiments demonstrate that our face detector trained by CS-AdaBoost needs fewer features to achieve higher detection rate and more robust performance than Viola’s method, and our application of the face detector to real-time face recognition shows that our face detector can achieve 18 frames per second with the size of 386 × 288 pixels on common Pentium III 1G computer.

The rest of the paper is organized as follow: Section 2 introduces the CS-AdaBoost algorithm. Section 3 describes the flowchart of face detector based on CS-AdaBoost. Section 4 and 5 give experimental results and conclusions.

2. COST-SENSITIVE ADABOOST
AdaBoost algorithm was mainly developed by Freund and Schapire [4]. It is a general method for improving the classification performance of any given learning algorithm. In its naive form, AdaBoost uses a base learner to induce multiple individual classifiers in sequential trials, and a weight is assigned to each training example. At the end of each trial, the weight is adjusted to reflect the importance of each training example for the next induction trial. This adjustment effectively increases the weights of misclassified examples and decreases the
weights of the correctly classified examples. These weights cause the learner to concentrate on different examples in each trial and so lead to different classifiers. Finally, the weak classifiers are combined to form a strong classifier. A detailed description of AdaBoost can be referred to [4].

Given training set \( L = \{(x_i, y_i)\}, i = 1, \ldots, N \), 
\( y_i \in \{+1, -1\} \) is the label of pattern \( x_i \) (in the following parts, +1 stands for face pattern, and -1 stands for non-face);

- Given cost parameter \( c > 0 \);
- Define \( C_i \) for each training sample

\[
C_i = \begin{cases} 
\frac{2c}{c+1}, & \text{if } y_i = +1 \\
\frac{c}{c+1}, & \text{if } y_i = -1 
\end{cases}
\]

- Initialize training set sample weight

\[
D_0(i) = \frac{C_i}{\sum_j C_j};
\]

- For \( t = 1, \ldots, T \):
  1. Train base classifier \( h_t \) using distribution \( D_t \) and the training set;
  2. Compute the model error of the base classifier

\[
\varepsilon_t = \sum_i D_t(i) \frac{(1 - h_t(x_i) y_i)}{2};
\]
  3. Choose \( \alpha_t = \frac{1}{2} \ln \frac{1 - \varepsilon_t}{\varepsilon_t} \);
  4. Update

\[
D_{t+1}(i) = \begin{cases} 
\frac{D_t(i)e^{-\alpha_t(2-C_i)}}{Z_t}, & \text{if } y_i h_t(x_i) = 1 \\
\frac{D_t(i)e^{\alpha_t C_i}}{Z_t}, & \text{if } y_i h_t(x_i) = -1 
\end{cases}
\]

where \( Z_t \) is a normalization factor so that

\[
\sum_i D_{t+1}(i) = 1
\]

- Output the final hypothesis:

\[
H(x) = \text{sign} \left( \sum_{t=1}^{T} \alpha_t h_t(x) \right).
\]

Fig. 1. The Cost-Sensitive AdaBoost learning procedures

In its naïve form, the initial weights of samples are equal, and the update of the weights does not differentiate the misclassification for samples from different classes. The classifier is designed to only yield a low error rate on the whole training data. In Fig.1, we introduce CS-AdaBoost, which takes into account misclassifications from different classes, especially the number of high cost errors, at two different levels.

Firstly, different initial weights are given to each sample according to its misclassification cost. When we select the cost parameter \( c > 1 \), the initial weight of positive class will be greater than that of the negative. Then the classifier will focus on the positive examples, and the classifier will get high detection rate after a few iterations. This is important in the first several stages of the face detector because of the demand of high speed.

On a local scale, CS-AdaBoost introduces a second variation to AdaBoost in the weight updating procedure. Due to the naïve AdaBoost’s greedy balanced weight-updating scheme, the initially unequal sample weights are immediately lost if we do not change its updating procedure. In CS-AdaBoost, for \( c > 1 \), the weights of the misclassified positive samples will be increased more than those of misclassified negatives, and the weights of the correctly classified negative samples will be decreased more than those positive correctly classified. The procedure therefore pays more attention to the hardest positive examples. In terms of margin analysis, the result of the procedure is to increase the margin of the positive samples more than for the negative ones.

Fig. 2. Cumulative margin distribution of three classifiers on face samples trained by CS-AdaBoost algorithm with different misclassification costs

In order to show the differences between CS-AdaBoost and the Naïve AdaBoost, three classifiers are trained on 11,580 face samples and 11,000 non-face samples by CS-AdaBoost with different misclassification costs, in which CS-AdaBoost with \( c = 1 \) is equivalent to the Naïve AdaBoost algorithm. The cumulative margin distributions of three classifiers on face samples are given in Fig.2. All these classifiers are constructed by 20 weak learners (the weak learner will be described in section 3.2),
and thresholds of them are adjusted to achieve 100% detection rate on face samples. The classifiers trained with $c = 1.5$ and $c = 2.5$ can exclude about 36% non-faces, while the classifier trained with $c = 1$ can only reject 30% non-faces. From Fig. 2, we also can see that the margins of faces of the classifiers trained with $c = 2.5$ and $c = 1.5$ are more concentrated in large values than the classifier trained with $c = 1$. According to Schapire’s analysis [7], the generalization of an AdaBoosting classifier can be characterized by its margin. So the classifier trained by CS-AdaBoost method not only excludes more non-face samples at the same detection rate, but also generalizes with more robust performance.

3. FLOWCHART OF FACE DETECTOR BASED ON CS-ADABOOST METHOD

Our face detector also has cascade structure similar to others [3][5]. To detect faces, the input image is repeatedly sub-sampled for several iterations and scanned with a $20 \times 20$ rectangular sub-window in which the decision of there being a face pattern is made, as Fig. 2 illustrates. First, the sub-window is preprocessed by gray-scale normalization. The preprocessing steps can lighten the influence of illumination and reduce the diversity of face patterns. Then, the features of the sub-window are extracted, and an individual strong classifier trained by using CS-AdaBoost algorithm is applied to classify the sub-window. A positive result from the previous stage triggers more features extraction and the evaluation of the strong classifier with more complex structure of the next stage. A negative outcome at any point leads to the immediate rejection of the sub-window. At last, the postprocessing method is used to merge the overlapping face candidates and output the final result.

3.1. Features of face detector

The high speed and detection rate of the algorithm depend not only on the cascade architecture but also on individual detector. Here each weak classifier is constructed based on a simple feature. Five types of simple features, which are block differences similar to steerable filters, are computed as shown in Fig. 4. Each feature can be computed very efficiently from the integral image [6].

Fig. 4. The five types of simple Harr wavelet like features defined on a sub-window: the sums of the pixels which lie within the white rectangles are subtracted from the sums of pixels in the gray rectangles.

Among them, the first four types of feature are used in [5]. We also find that the last type of feature is pretty effective to capture the characteristics of faces around the area of eyes and mouth. There are a total number of 92,267 features for $20 \times 20$ pixel sub-windows (with some restrictions to block size).

3.2. Design of the individual strong classifier

As Fig. 3 illustrates, the whole face detector is a cascade of face/non-face strong classifiers. An individual face/non-face strong classifier is constructed based on a number of weak classifiers where a weak classifier uses a single feature by thresholding the value of the feature according to the face/non-face histograms of the feature [5]. Here we use previous presented CS-AdaBoost learning algorithm to select features and combine these weak classifiers to a strong classifier.

When training the whole cascade, every strong classifier is trained by the non-faces passing through the previous strong classifier via bootstrapping method [3]. In the early stage of the cascade, in order to reject non-faces as much as possible and miss faces as few as possible on training set, the cost parameter $c$ should be much higher than 1. With the false alarms on training set fewer and fewer, $c$ should decrease gradually. In the last stages, because the remaining non-faces are much similar to face patterns, $c$ should be close to 1.

3.3. Post-Processing stage

Since the final detector is insensitive to small changes in translation and scale, multiple detection will usually occur around each face in a scanned image, while false detections often occur with less consistency. Here we adopt a simple heuristic method similar to that of [3] to merge overlapping detections. For each location, the number of child detections within a specified neighborhood of that location, and the maximum confidence of these detections can be counted. If both of them are above specific thresholds, then that location is classified as a face.
Table 1: Detection rates for various numbers of false alarms on the CMU+MIT test set containing 507 faces

<table>
<thead>
<tr>
<th>False Alarms</th>
<th>10</th>
<th>31</th>
<th>50</th>
<th>65</th>
<th>78</th>
<th>95</th>
<th>167</th>
</tr>
</thead>
<tbody>
<tr>
<td>Detector</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Our method</td>
<td>90.1%</td>
<td>91.3%</td>
<td>92.5%</td>
<td>93.1%</td>
<td>93.3%</td>
<td>93.5%</td>
<td>94.1%</td>
</tr>
<tr>
<td>Viola-Jones</td>
<td>76.1%</td>
<td>88.4%</td>
<td>91.4%</td>
<td>92.0%</td>
<td>92.1%</td>
<td>92.9%</td>
<td>93.9%</td>
</tr>
<tr>
<td>Viola-Jones(voting)</td>
<td>83.1%</td>
<td>89.7%</td>
<td>92.1%</td>
<td>93.1%</td>
<td>93.1%</td>
<td>93.2%</td>
<td>93.7%</td>
</tr>
<tr>
<td>H.Rowley</td>
<td>83.6%</td>
<td>86.2%</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>89.2%</td>
<td>90.1%</td>
</tr>
</tbody>
</table>

4. EXPERIMENTS

Face training samples are collected from different face databases and Internet to cover wide variations in facial expression and lighting condition. Each face sample is manually cropped and normalized to the size of 20 × 20 pixels. In order to make the classifier less sensitive to rotation and size variation, several face samples are generated from each original sample by rotating 10 degrees left and right, and scaling between 0.8 and 1.2. Approximately 11,580 face samples are collected to train all the classifiers. All these samples are preprocessed by gray-scale normalization.

Fig. 5. Detection rate versus false alarms of our detector on MIT+CMU’s test set.

Performance of the face detector is evaluated on the MIT+CMU’s test set [3], which consists of 130 images with 507 faces. By adjusting the threshold of the number of child detections and maximum confidence belonging to a face, the ROC curve of our detector with search step 1 pixel on this test set is shown in Fig 5. As a comparison with Viola’s method [5] and H.Rowley’s NN [3] on the test set, Table 1 lists the detection rate for various numbers of false detections for our system as well as these published systems. From the Table 1, it can be seen that our face detector has better detection performance. Especially when the number of false alarms is 10, the detection rate of our detector without any voting strategy is about 7%~14% higher than others. From the ROC curve, we also can see that our detector is more robust when changing the thresholds. Besides these, our detector only has 20 layers and totally about 3000 features (weak learners), while the Viola’s detector has 38 layers and 6000 features.

We also apply our face detector to video-based face recognition. The face detector can process the video stream at the rate of 18 frames of size 386 × 288 pixels per second to detect faces from 30 × 30 pixels to 240 × 240 pixels on common Intel Pentium III 1G computer.

5. CONCLUSION

A face detection method based on cost-sensitive AdaBoost is presented in this paper. In our method, we give different cost to the misclassification of having a face missed and having a false alarm to train the cost-sensitive AdaBoost classifiers to achieve more robust performance and higher speed over conventional AdaBoost-based methods. Comparative results on test sets demonstrate the effectiveness of the algorithm.

6. REFERENCES