REMOVAL OF FALSE POSITIVE IN OBJECT DETECTION WITH CONTOUR-BASED CLASSIFIERS

Hongyu Li
School of Software Engineering
Tongji University, Shanghai, China
hyli@tongji.edu.cn

Lei Chen
School of Software Engineering
Tongji University, Shanghai, China
enjiushi@gmail.com

ABSTRACT
This paper proposes a method of constructing a contour-based classifier to remove the false positive objects after Haar-based detection. The classifier is learned with the discrete AdaBoost. During the training, the oriented chamfer is introduced to construct strong learners. Experimental results have demonstrated that the proposed method is feasible and promising in the removal of the false positive.

Index Terms— boosting, contour, classifier, false positive

1. INTRODUCTION
Object detection has attracted the attention of researchers for years, and a lot of methods such as [1, 2, 3, 4] have been proposed. Haar-based detectors are quite popular in this field since they can effectively find the intensity difference between areas. On the basis of these detectors, a cascade scheme was proposed in [3] to reduce the computation cost. In this scheme, each step during detection produces binary results. Negative results are immediately rejected while positive ones trigger next step. A cascade detector is actually a series of sub-window filters. However, the shortcoming of such detectors is that they often bring too many false positive areas, although the true positive is seldom missed.

In this paper, we propose an approach to remove the false positive through combining Haar-based detectors with contour-based classifiers. Fig. 1 presents the workflow of the proposed method. Firstly, the Haar-based detector is adopted to search for possible positions of an object. Secondly, the contour-based classifier is used to remove the false positive from candidates. On one hand, the Haar-based detector can avoid the omission of possible objects and run rapidly. On the other hand, unlike Haar-based features, contour-based features put more emphasis on global information of objects. This makes possible successfully removing the false positive candidates.

The remainder of this paper is organized as follows: Section 2 briefly presents the oriented chamfer. Section 3 discusses the contour-base classifier. Experimental results are shown in Section 4. This paper ends with conclusions in Section 5.

2. ORIENTED CHAMFER
The oriented chamfer [4] is helpful and powerful in contour match since it is sensitive to edge and shape and good at dealing with rigid area. It consists of two parts, chamfer distance and edge orientation. The former calculates the mean distance between pixels in contour fragment images and edge images. The chamfer distance $d_{cham}$ at position $x$ in a classifier edge can be formulated as follows,

$$d_{cham}(x) = \frac{1}{N} \sum_{t \in T} \min_{e \in E} ||(t + x) - e||_2.$$ \hspace{1cm} (1)

Here $E$ represents the edge pixel set extracted from the target image. $N$ is the number of pixels in the classifier edge pixel set $T$. $t$ and $e$ represent elements from $T$ and $E$ respectively.

The edge orientation part calculates the mean orientation difference between pixels in contour fragment images and
3. CONTOUR-BASED CLASSIFIER

3.1. Building a Training Set

In this study, a training set is composed of a certain amount of images with potential objects. To build such a training set, Haar-based detector is first used on some original images to extract sub-windows with candidate objects, as shown in Fig. 2(a). The corresponding images to those extracted windows still need to be processed to obtain more cues of objects. Here, three features are computed, Canny edge, distance, and angle, as shown in Fig. 2(b). The edges are extracted with the Canny operator. The distance transform algorithm in [5] is performed on each candidate image to generate the distance image. We mean the orientation of edge pixels by angle. In general, the angle \( \theta_P \) is normalized in the following form,

\[
\theta_P = (\theta_A + \pi/2) \times 255/\pi,
\]

where \( \theta_A \) is the original angle value ranging from \(-\pi/2\) to \(\pi/2\). The angle value \( \theta_P \) after normalization ranges from 0 to 255. In Fig. 2(b), the bottom three rows respectively represent the Canny edge, distance, and angle.

Finally, the Canny edge, distance and angle images build a training set together with the intensity images in the extracted sub-windows. In this training set, images are classified into two groups. One is true positive, where the object in the window is true or correct. The other is false positive, where the object in the window is false or fake.

In addition, a condition must be satisfied during building the training set in our experiments. That is, the resolution of training images must be more than \(300 \times 300\), which assures that the generated Canny edges are enough.

3.2. Training Step

The training step in the proposed method will introduce the idea of the boosting algorithm. The boosting algorithm was originally proposed in the literature [6, 7, 8] of machine learning. One of often used versions is the discrete AdaBoost [9]. The discrete AdaBoost is able to combine a set of weak learners into a strong learner, which can perform well in classification.

In our training procedure, a training sample \( x_i \) is constructed with candidate images as well as the corresponding edge, distance, and angle images. The label value \( y_i \) is manually assigned according to the property of candidate images. \( y_i \) is equal to 1 for true positive, \(-1\) for false positive. For each training sample, there is a weak learner \( h \) to generate.

The way of computing the oriented chamfer is slightly different here. For the candidate image, position \( x \) always stays at the top left corner, therefore there are no changes for \( t \). Formulas (1) and (2) are thus modified as:

\[
d_{\text{cham}}(x_i) = \frac{1}{N} \sum_{t \in T} \min_{e \in E} \| t - e \|_2; \tag{6}
\]

\[
d_{\text{orient}}(x_i) = \frac{1}{N} \sum_{t \in T} | o(t) - o(\vartheta_E(t)) |; \tag{7}
\]

Formula (4) is changed into:

\[
d(x_i) = d_{\text{cham}}(x_i) + \lambda_i d_{\text{orient}}(x_i). \tag{8}
\]
Given $m$ training samples, $\{x_1, \ldots, x_m\}$, the corresponding labels $\{y_1, \ldots, y_m\}$, and initial weights $D_1(i) = 1/m, i = 1, \ldots, m$, the whole training procedure is described as follows:

For $j = 1, \ldots, t$;
For $k = 1, \ldots, n$;
Use weak learner $h_k$ to calculate the oriented chamfer $d$ for each training sample $x_i$;
Find best parameters $(\lambda_k, \theta_k)$ that minimize the error rate $\epsilon_k$;
Store the best learner with the lowest error rate and the corresponding parameters as $h'_j, \lambda'_j, \theta'_j$, and $\epsilon'_j$;
If $\epsilon'_j \geq 0.5$ then stop; else continue;
Set $\alpha_j = \frac{1}{2} \log(\frac{1-\epsilon'_j}{\epsilon'_j})$ and update $D_{j+1}(i)$ with Eq. 9;
End
End

Here, the error rate $\epsilon_k$ is the weighted sum of the logical expression $h_k(x_i) \neq y_i$.

$$\epsilon_k = \sum_{i=1}^{m} D_j(i) [h_k(x_i) \neq y_i].$$

The weak learner $h_k(x_i)$ is obtained from the $\delta$ function,

$$h_k(x_i) = \delta(d(x_i) < \theta_k).$$

$\delta(\cdot)$ returns 1, when the oriented chamfer $d(x_i)$ is less than the threshold $\theta_k$; otherwise -1. The weights are updated in the following way,

$$D_{j+1}(i) = \frac{D_j(i) \exp(-\alpha_j y_i h'_j(x_i))}{Z_j},$$  \hspace{1cm} (9)

where $Z_j$ is a normalization factor. The purpose of updating weights is to increase the influence of those wrongly classified training samples while decreasing the others.

In the end, the obtained $\alpha'_j, \lambda'_j$, and $\theta'_j$ from the training step are used to construct the strong classifier.

3.3. Test Step

In the test step, we will make use of the classifier generated in the training step to determine the label of test samples. Before testing, the Haar-based detector needs to run on the input image to search for candidate windows. At the same time, the Canny edge, distance, and angle images are also extracted for each candidate image. Therefore, the oriented chamfer value $d$ is easily calculated with the trained weak learner. Then the classifier can be described as follows,

$$C(x) = \sum_{j=1}^{t} h'_j(x) = \sum_{j=1}^{t} \alpha'_j \delta(d(x) < \theta'_j),$$  \hspace{1cm} (10)

where $\alpha'_j$ and $\theta'_j$ are both from the training step. The classification strategy is that the test window is true positive when the value $C$ is more than zero, otherwise false positive.

4. EXPERIMENTAL RESULTS

In this section, we will check the feasibility of the proposed method in removing the false positive. In our experiments, the used dataset is the Celtech face database. A 17-stage Haar-based cascade detector was first executed on this dataset, and 694 potential faces were totally found, including 418 true positive and 276 false positive. Among these candidate faces, 212 true positive face images and 138 false positive face images were used for training. The training procedure spent 3 hours in generating a classifier with 9 weak learners. The left candidates are used for test. The classification accuracy is presented in Table 1 and some examples of experimental results are shown in Fig. 3.

<table>
<thead>
<tr>
<th>Table 1. Classification accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>test samples</td>
</tr>
<tr>
<td>false positive</td>
</tr>
<tr>
<td>true positive</td>
</tr>
</tbody>
</table>

From Fig. 3, it is easy to find that the proposed method can work well in removing the false positive that successfully cheats Haar-based detectors. In the Caltech dataset, the classification accuracy is high enough, 92.029% and 96.698% respectively for correctly distinguishing the false positive and true positive images. Besides, we also tested some images with multiple faces obtained from internet in our experiments.
An example is shown in Fig. 4. In this case, if only Haar-based detector was used, eight false positive faces were obtained. After postprocessing with the proposed method, only two false positive faces were left and the others were successfully removed.

In addition, observing the misclassified samples, we notice that most of these images are very small, with the size of around 35 × 35. That results from the information loss small images suffer during extracting edges. Edges play an important role in the proposed method. Therefore, for small images with disordered edges, the classification strategy can hardly perform well, while large images with clear edges can be generally classified in a correct way.

For an image with the size of 60 × 60, the whole computation procedure needs about half a second. The computation time will increase proportionally with the image size and the number of edges in images. In fact, the time cost mainly comes from the computation of the distance and angle since the oriented chamfer distance only needs to be calculated once for each window.

5. CONCLUSIONS

This paper proposes a postprocessing method of Haar-based detectors to remove the false positive objects. The idea of the discrete AdaBoost is introduced in this approach to bipartite the detected results into true positive and false positive. During the training, the oriented chamfer is adopted to construct a strong classifier. Compared to contour-based detectors, the proposed method has greatly reduced the computation cost. Of course, there are still chances to improve the computation speed with the help of parallel computing or GPU. Experimental results have proven that the proposed method is feasible and promising in the removal of the false positive.

6. ACKNOWLEDGMENTS

This research was partially supported by NSFC Grant 6090-3120, 863 Project 2009AA043001 and Shanghai Natural Science Foundation Grant 09ZR1434400.

7. REFERENCES


