Cascade AdaBoost Classifiers with Stage Optimization for Face Detection

Zongying Ou, Xusheng Tang, Tieming Su, and Pengfei Zhao

Key Laboratory for Precision and Non-traditional Machining Technology
of Ministry of Education, Dalian University of Technology, Dalian 116024, P.R. China
ouzyg@dlut.edu.cn

Abstract. In this paper, we propose a novel feature optimization method to build a cascade Adaboost face detector for real-time applications, such as teleconferencing, user interfaces, and security access control. AdaBoost algorithm selects a set of weak classifiers and combines them into a final strong classifier. However, conventional AdaBoost is a sequential forward search procedure using the greedy selection strategy, the weights of weak classifiers may not be optimized. To address this issue, we proposed a novel Genetic Algorithm post optimization procedure for a given boosted classifier, which yields better generalization performance.

1 Introduction

Many commercial applications demand a fast face detector, such as teleconferencing, user interfaces, and security access control [1]. Several face detection techniques have been developed in recent years [2], [3], [4], [5]. Due to the variation of poses, facial expressions, occlusion, environment lighting conditions etc., fast and robust face detection is still a challenging task.

Recently, Viola [3] introduced an boosted cascade of simple classifiers using Haar-like features capable of detecting faces in real-time with both high detection rate and very low false positive rates, which is considered to be one of the fastest systems. Central part of this method is a feature selection and combination algorithm based on AdaBoost [6]. Some of the recent works on face detection following Viola-Jones approach also explore alternative-boosting algorithms such as Float-Boost [7], GentleBoost [8], and Asymmetric AdaBoost [8]. In essence, Adaboost is a sequential learning approach based on one-step greedy strategy. It is reasonably expected that a post global optimization processing will further upgrade the performance of Adaboost. This paper investigates performance improvement of cascade Adaboost classifier by post stage optimization using Genetic Algorithm.

The remainder of this paper is organized as follows. In section 2 the Adaboost learning procedure proposed in [3] is introduced. The stage Optimization procedure based on Genetic Algorithms is presented in section 3. Section 4 provides the experimental results and conclusion is drawn in section 5.

D. Zhang and A.K. Jain (Eds.): ICB 2006, LNCS 3832, pp. 121 – 128, 2005.

2 Cascade of AdaBoost Classifiers and Performance Evaluation

There are three elements in the Viola-Jones framework: the cascade architecture, a set of Haar-like features, and AdaBoost algorithm for constructing classifier.

A cascade of face classifiers is a decision tree where at each stage a classifier is trained and formed to detect almost all frontal faces while rejecting a certain fraction of non-face patterns. Those image-windows that are not rejected by a stage classifier in the cascade sequence will be processed by the succeed stage classifiers. The cascade architecture can dramatically increases the speed of the detector by focusing attention on promising regions of the images.

Each stage classifier was trained using the Adaboost algorithm [6]. The idea of boosting is selecting and ensemble a set of weak learners to form a strong classifier by repeatedly learning processing over the training examples. In i stage, T numbers of weak classifiers h_{ij} and ensemble weights α_{ij} are yielded by learning. Then a stage strong classifier H_i (x) is:

$$H_i(x) = \begin{cases} 1 & \sum_{j=1}^{T} \alpha_{ij} h_{ij}(x) \geq \theta_i \\ -1 & otherwise \end{cases} . \tag{1}$$

The stage threshold θ_i is adjusted to meet the detection rate goal.

As conventional AdaBoost is a sequential forward search procedure based on the greedy selection strategy, the coefficients may not be optimal globally. Ideally, given $\{h_1,\ldots h_T\}$, one solves the optimization problem for all weak classifier coefficients $\{\alpha_1,\ldots \alpha_T\}$. The task becomes to construct a learning function that minimizes misclassification error.

3 Genetic Algorithms for Stage Optimization

To achieve high detection performance, the false rejection rate (FRR) and the false acceptance rate (FAR) should be both as low as possible. We take the minimum FAR as optimal object function, and take the FRR within an allowance magnitude as constraint condition. The weight α_{ij} and threshold θ. are the optimal parameters in optimization processing. For a given sets of positive and negative samples $\{(x_1,y_1)\ldots(x_k,y_k)\}$ where $y_i=\pm 1$, given the FRR f, the optimization model can be written as:

$$\begin{aligned} &\arg\min_{\alpha_i,\theta}(num(y_i^n \neq H(x_i^n \mid \alpha_i,\theta)) / num(x_i^n)) \\ &s.t.\ \ num(y_i^p \neq H(x_i^n \mid \alpha_i,\theta)) / num(x_i^n \mid) \leq f \end{aligned} . \tag{2}$$

The function num $(\cdot)$ means the numbers of samples and the superscript p and n denote the positive and negative samples respectively. A true gradient decent cannot be implemented since the H (x) is not continuous. To address this issue, we use the Genetic algorithms to optimize the parameter.

3.1 Individual Representation and Fitness Function

In order to apply genetic search a mapping must be established between concept descriptions and individual in the search population. Assume that the stage classifier contains T weak classifiers (h_i) with T weight values α_t and threshold b. This information is encoded in a string as Fig.1.

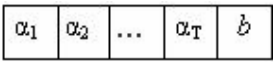

Fig. 1. The representational structure of individual

The fitness function concerns accuracy measures-high hit rate (*hit*) and low false acceptance rate (*f*), and is defined as follow:

$$F = \begin{cases} 1 - n^- / N^- + m^+ / M^+ & if \quad m^+ / M^+ \geq hit \\ m^+ / M^+ & if \quad m^+ / M^+ < hit \end{cases} . \tag{3}$$

where: m^+ is the number of labeled positive samples correctly predicted,
M^+ is the total number of labeled positives samples in the training set,
n^- is the number of labeled negative samples wrongly predicted,
N^- is the total number of labeled negative samples in the training set,
hit is the hit rate of the original stage classifier in the training set.

3.2 Cascade Face Classifiers GA Post Optimization Learning Framework

We adapted "bootstrap" method [10] to reduce the size of the training set needed. The negative images are collected during training, in the following manner, instead of collecting the images before training is started.

1. Create an initial set of nonface images by collecting *m* numbers of random images. Create an initial set of face images by selecting *l* numbers of representative face images. Given a total stage number T_S, the final cumulatively false acceptance rate *f*.
2. Set stage number S =1
3. Train a stage face classifier using these $m+l$ numbers of samples by Discrete Adaboost [3].
4. Using **GA algorithm [11]** to optimize the stage classifier.
5. Add this stage face classifier to ensemble a cascade face classifier system. Run the system on an image of scenery that contains no faces and filter out *m* numbers of negative images that the system incorrectly identifies as face to update the negative samples.
6. S=S+1;
7. If ($S < T_S$ and (m/the numbers of detected image)>*f*) Go to step 3.
8. Else Exit.

4 Experimental Results

The training face image set is provided by P.Carbonetto [12], which contains 4916 face images of size of 24×24. The non-faces samples are collected from various sources using the "bootstrap" method as mentioned above. Each stage 9000 non-face samples are used.

Two cascade face detection systems consisting of 30 stages were trained: One is with conventional AdaBoost [3] and the other is with our novel post-optimization procedure for each stage classifier. A Harr-like candidate feature set as used in [3] is adopted for Adaboost processing, and the selected weak classifiers is combined to form a stage classifier.

Parameters used for evolution were: 70% of all individuals undergo crossover, 0.5% of all individuals were mutated. The GA terminated if the population was converged to a good solution so that no better individual was found within the next 2000 generations. If convergence did not occur within 10000 generations, the GA was stopped as well.

We tested our systems on the CMU dataset [2] and the non-faces test set of CBCL face database [13]. The CMU dataset has been widely used for comparison of face detectors [2,3,7,8]. It consists of 130 images with 507-labeled frontal faces. The non-faces test set of CBCL face database contains 23,573 non-faces images, which resize to 24×24 pixel.

The criterion [8] is used to evaluate the precision of face localization. A hit was declared if and only if

- The Euclidian distance between the center of a detected and actual face was less than 30% of the width of the actual face as well as
- The width of the detected face was within ±50% of the actual face width.

During detection, a sliding window was moved pixel by pixel over the picture at each scale. Starting with the original scale, the features were enlarged by 20% until exceeding the size of the picture in at least one dimension.

Often multiple faces are detected at near by location and scale at an actual face location. Therefore, multiple nearby detection results were merged. Receiver Operating Curves (ROC) was constructed by varying the required number of detected faces per actual face before merging into a single detection result.

Fig.2 shows changes of weights of composed weak classifiers in the first stage in the process of GA optimization. There are total 14 weak classifiers in this stage. In training process, two methods are used to generate initial individual. One initializes the weight individual near the original weight yielded by conventional Adaboost. The other randomly initializes the weight individual. As can be seen in Fig.3 the first method reach the optimizations object (FAR=0.394) very quickly with about 66 iterations. Both methods can reach same optimization level, though the randomly initialized weights method takes much more iteration before convergence after GA post-optimization. The false acceptance rate on training set was about 15% lower than before, while keeping the hit rate constant at 99.95% as shown in Fig.3. In Fig.2 we can see the weight of the 12^{th} weak classifier of the first stage is close to zero. The small weight implies the less important in discrimination the weak classifier will be. With

this heuristic, the weak classifier whose weight closes to zero can be removed. This will lead to fewer weak classifiers and consequently decrease the total processing work in classifying. Just as shown in Fig.3 after deleting the 12^{th} weak classifier and re-post optimization, the false acceptance rate will be change to 0.41,which is about 3.9% higher than without post optimal processing.

Table 1. A comparison of the false acceptance rate of total 16 stages in a cascade Adaboost processing with and without post GA optimization on the non-face test set of CBCL Database

Stage NO.	False acceptance rate		Stage NO.	False acceptance rate	
	conventional AdaBoost	With GA-post-optimization		conventional AdaBoost	With GA-post-optimization
1	0.7572	0.6440	9	0.1243	0.1118
2	0.6637	0.5500	10	0.1614	0.1453
3	0.4817	0.4045	11	0.0706	0.0607
4	0.4221	0.3413	12	0.1240	0.1066
5	0.6774	0.5758	13	0.2027	0.1724
6	0.3157	0.2715	14	0.2257	0.1918
7	0. 3560	0.3100	15	0.2468	0.2087
8	0.3349	0.2947	16	0.3052	0.2503
Final Cascade system FAR				0.0013	0.00067

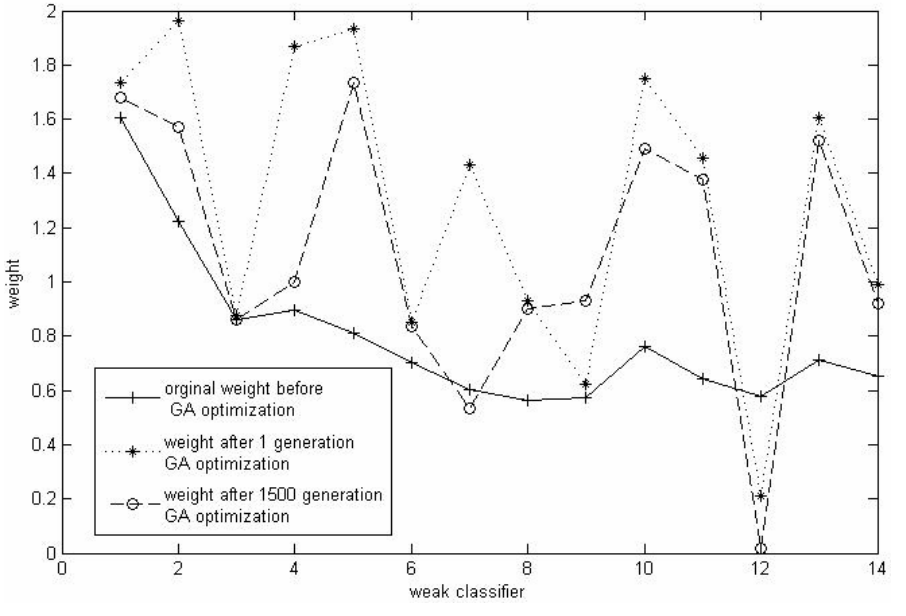

Fig. 2. The weight values of weak classifier in stage 1 with and without GA post optimization

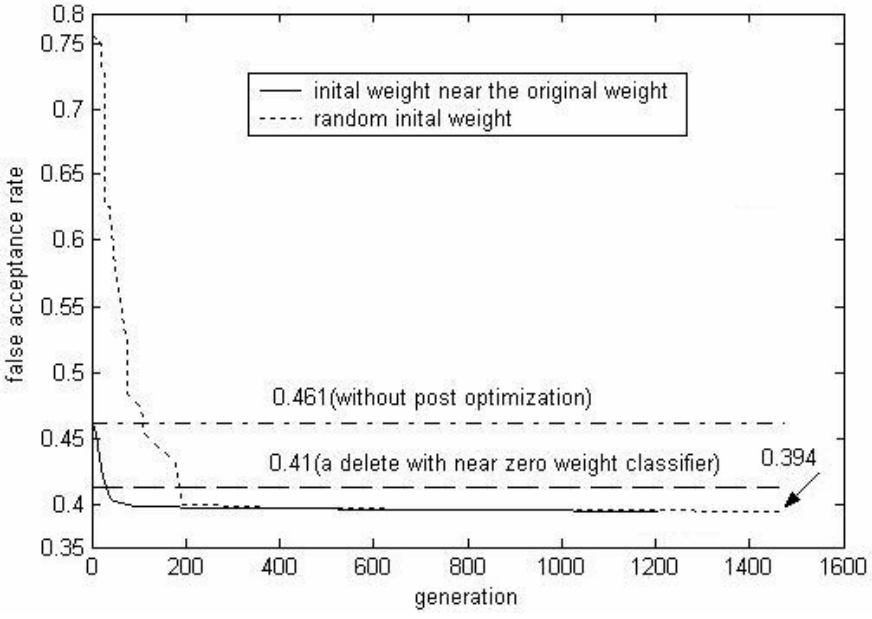

Fig. 3. The changes of false acceptance rate of stage 1 in cascade Adaboost with post GA optimization on training set (keeping hit rate constant)

Table 2. A Comparison of detection rate for various face detectors on the MIT+CMU test set

Detector \ False Acceptance number	10	31	50	95	167
With GA-post optimization(our)	81.3%	89.9%	92.4%	93.5%	94.1%
Without GA-post optimization(our)	80.9%	89.3%	91.5%	92.9%	93.5%
Viola-Jones(voting Adaboost) [3]	81.1%	89.7%	92.1%	93.2%	93.7%
Viola-Jones(Discrete Adaboost) [3]	79.1%	88.4%	91.4%	92.9%	93.9%
Rowley-Baluja-Kanade [3]	83.2%	86.0%	-	-	90.1%

We tested two face detection systems on the non-faces test set of CBCL face database. As cascade structure adopt the more non-face sub-window discard in early stage, the quicker detection speed will be achieved. From Table 1 we also can see that the face detector by GA post optimization discards more non-face image with same number of stage. This means GA post optimization can upgrade effectively the detection speed and accuracy. The average decrease of false acceptance rate is about 14.5%. Table 1 also shows that the final FAR of the classifier with post optimization was about 50% (0.00067 vs. 0.0013) lower than the classifier without post optimization.

Table 2 lists the detection rates corresponding to specified false acceptance numbers for our two systems (with and without post optimization) as well as other pub-

lished systems (the data is adopted from Ref.[3]). The test database is MIT+CMU test set. As shown from Table.2, GA-post-optimization boosting outperformed the conventional Adaboost.

5 Conclusion

Adaboost is an excellent machine-learning algorithm, which provides an effective approach in selecting the discriminating features and combining them to form a strong discriminating classifier. Based on above framework, many face detection algorithms have got much success in practice. However, in essence Adaboost is a sequentially one-step forward greedy algorithm. It is expected that a global optimization will further improve the performance of Adaboost. A stage post GA optimization schema for cascade Adaboost face detector is presented in this paper. The experiment example shows that the false acceptance rate can be decrease 15% (from 0.461% to 0.39%) in one stage while the hit rate of stage keeps the same level on train set. The decrease rates of false acceptance rate in different stage on test set are about the similar value as shown in table 1, which means the classifier with GA post optimization achieves higher detection rate than the conventional Adaboost classifier. A total average decrease rate of false acceptance rate is about 50%, which implies that the cascade detector will decrease a similar percentage of processing work in repeating treating the non-face image regions, which will lead to increase the detection speed. The experiment also shows that the hit rate and the false acceptance rate can be both simultaneously upgrading with stage post optimization.

Reference

1. Yang, M. H., Kriegman, D. J., and Ahuja, N.: Detecting Faces in Images: A Survey. IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 24 (2002) 34-58
2. Rowly, H., Baluja, S., and Kanade, T.: Neural network-based face detection. PAMI, Vol. 20 (1998) 23-38
3. Viola, P., Jones, M.: Rapid object detection using a boosted cascade of simple features. IEEE CVPR, (2001) 511~518
4. Romdhani, S., Torr, P., Schoelkopf, B., and Blake, A.: Computationally efficient face detection. In Proc. Intl. Conf. Computer Vision, (2001) 695–700
5. Henry, S., Takeo, K.: A statistical model for 3d object detection applied to faces and cars. In IEEE Conference on Computer Vision and Pattern Recognition. (2000)
6. Freund, Y., Schapire, R.: A diction-theoretic generalization of on-line learning and an application to boosting. Journal of Computer and System Sciences, Vol. 55 (1997) 119-139
7. Li, S.Z., Zhang, Z.Q., Harry, S., and Zhang, H.J.: FloatBoost learning for classification. In Proc.CVPR, (2001) 511-518
8. Lienhart, R., Kuranov, A., and Pisarevsky, V.: Empirical analysis of detection cascades of boosted classifiers for rapid object detection. Technical report, MRL, Intel Labs, (2002)
9. Viola, P., Jones, M.: Fast and robust classification using asymmetric AdaBoost and a detector cascade. In NIPS 14, (2002)

10. Sung, K.K.: Learning and Example Selection for Object and Pattern Detection. PhD thesis, MIT AI Lab, January (1996)
11. Goldberg, D.E.: Genetic algorithms in search, optimization, and machine learning, Addison-Wesley, Reading, A (1989)
12. Carbonetto, P.: Viola training data (Database). URL http://www.cs.ubc.ca/~pcarbo
13. http://cbcl.mit.edu/projects/cbcl/software-datasets/FaceData1Readme.html