Towards Optimal Training of Cascaded Detectors

S. Charles Brubaker, Matthew D. Mullin, and James M. Rehg

College of Computing and GVU Center,
Georgia Institute of Technology, Atlanta, GA 30332
{brubaker, mdmullin, rehg}@cc.gatech.edu

Abstract. Cascades of boosted ensembles have become popular in the object detection community following their highly successful introduction in the face detector of Viola and Jones [1]. In this paper, we explore several aspects of this architecture that have not yet received adequate attention: decision points of cascade stages, faster ensemble learning, and stronger weak hypotheses. We present a novel strategy to determine the appropriate balance between false positive and detection rates in the individual stages of the cascade based on a probablistic model of the overall cascade's performance. To improve the training time of individual stages, we explore the use of feature filtering before the application of Adaboost. Finally, we show that the use of stronger weak hypotheses based on CART can significantly improve upon the standard face detection results on the CMU-MIT data set.

1 Introduction

Object detection is one of the classic problems in computer vision, having applications to surveillance, robotics, multimedia processing, and HCI. Developing a generic object detection system is still an open problem, but there have been important successes over the past several years for some visual patterns such as faces [1], pedestrians [2], and cars [3]. Among the most influential systems is the face detector of Viola and Jones [1], which can be credited with the widespread popularity of cascaded detectors. We refer to detectors conforming to this general system architecture as cascades of boosted ensembles, or CoBEs.

The key elements of Viola and Jones' approach are:

- The cascade structure, which enables the detector to be simultaneously fast and accurate.
- The use of Adaboost [4] to combine weak hypotheses into a strong ensemble.
- Thresholding on single feature values to form weak hypotheses (threshold-based hypotheses).
- Feature selection from a large set of features, each of which might be only weakly discriminative in itself.

The large body of literature spawned by this seminal work has tended to focus on alternatives to Adaboost and on alternative feature sets, while other aspects

A. Leonardis, H. Bischof, and A. Pinz (Eds.): ECCV 2006, Part I, LNCS 3951, pp. 325–337, 2006.
© Springer-Verlag Berlin Heidelberg 2006

of the architecture have not received adequate attention. Here we focus on the false positive vs. detection trade-off in the individual stages of the cascade, faster ensemble learning, and the combination of Adaboost with CART [5] to improve detection performance.

The stages of the cascade are trained sequentially, as the output of one stage affects the training examples given to the next. Deciding when to stop training one stage and move on to the next and knowing the appropriate operating point on a stage's ROC curve are critical steps in the training of a cascade. Despite the guidelines provided in [1] and [6], however, obtaining state-of-the-art performance requires that these decisions be made by hand. We present a novel method for cascade learning, which uses a statistical model to predict the final cascade's performance and chooses the detection and false positive rates for the individual stages to meet a performance goal for the entire cascade. We show that the method is robust in the sense that a single set of parameters yields excellent performance over a variety of detection strategies and that it is capable of producing state of the art results.

One of the greatest obstacles to wider use of the CoBE architecture is that the detectors take a long time to train. We explore the use of feature filtering to reduce the feature pool available to Adaboost. Although this idea would seem to hold significant promise for speeding up the training process, we found it to be only moderately effective.

A remarkable aspect of the original Viola-Jones face detector is that it relies so heavily on Adaboost to produce the stage classifiers from such weakly discriminative individual features. We show that although this approach may be computationally efficient, combining Adaboost with CART-based weak learning can significantly improve the final detector's output.

In summary, we

- introduce a new criterion for cascade training, which provides a principled and robust mechanism for choosing stage thresholds and deciding when to stop training one stage and move onto the next,
- evaluate several feature filtering methods as ways to speed up the training process, and
- show that combining Adaboost with slightly stronger CART-based weak classifiers can improve the detector's performance over the standard practice of using threshold-based weak classifiers.

2 Previous Work

To make our discussion of previous work clear, we present a general framework for training a cascade of boosted ensembles in the Learn-CoBE procedure. The subroutines should be understood as placeholders for any number of solutions to the subproblem in question. Although not all changes made to the original Viola and Jones implementation strictly fit into this architecture, we believe it provides a useful abstraction of the CoBE approach.

Let F be the set of features and E the set of examples. We denote the weights for E as W. No more than L iterations of Adaboost are permitted. G refers to the goal cost for the cascade, and $\langle \hat{f}_i, \hat{d}_i \rangle$ denotes the false positive and detection rate pair for the ith stage.

procedure LEARN-COBE()
  $C \leftarrow \emptyset$ {Initialize an empty cascade.}
  for each stage i **do**
    $E \leftarrow$ BOOTSTRAP() {Acquire examples accepted by the current cascade.}
    $F' \leftarrow$ FILTER-FEATURES() {Reduce feature pool available to Adaboost.}
    $s_i \leftarrow \emptyset$ {Initialize current stage.}
    $W \leftarrow$ INITIALIZE-WEIGHTS() {Initialize example weights.}
    repeat
      $h \leftarrow$ WEAK-LEARN() {Learn a new hypothesis based on W.}
      $W \leftarrow$ REWEIGH-EXAMPLES() {Reweigh examples based on h.}
      $s_i \leftarrow s_i \cup h$ {Add the new hypothesis to the ensemble.}
      $\theta_i \leftarrow$ FIND-BEST-THRESHOLD() {Choose a threshold for the ensemble.}
      $\langle \hat{f}_i, \hat{d}_i \rangle \leftarrow$ VALIDATE() {Evaluate current ensemble on validation data.}
    until $|s_i| > L$ or PREDICT-COST() $\leq G$ {Is performance good enough?}
    $C \leftarrow C \cup \langle s_i, \theta_i \rangle$ {Add the stage to the cascade.}
  end for

Despite the critical importance of the FIND-BEST-THRESHOLD and PREDICT-COST functions to the performance of the final detector, they have received little attention. Our earlier work [7], is the only paper that addresses these questions directly in the CoBE context. In comparison, our new method of section 3.2 treats the actual cascade performance as a random variable, which is re-estimated during training stages.

Huitao Luo has recently published a method for adjusting the stage thresholds after the full cascade has been trained [8]. While the success of this method illustrates the importance of the stage thresholds for classification performance, it does not address how the thresholds should be chosen in the cascade training phase (FIND-BEST-THRESHOLD) – something that critically influences the bootstrapped data – or when it is appropriate to begin training a new stage (PREDICT-COST).

Šochman and Matas [9] use Waldboost to build a single boosted ensemble. When applying the detector to an instance, they decide whether to accept, reject, or continue evalutation after each weak hypothesis is calculated. This decision is based on an adaptation of Wald's sequential probability ratio test. Their test does not apply directly to our detectors, because each of our stage decisions is based on a new ensemble. In constrast to their method, we build an explicit probabilistic model of cascade performance based on validation data.

To improve the ensemble training time, we [10] showed how Adaboost with threshold-based weak learners can be replaced with Forward Feature Selection (FFS). Without any loss in detection performance, we were able to improve the training time of an ensemble over the original implementation of Adaboost. The key to the improved training time is that in FFS the best feature thresholds can be precomputed.

Leo Brieman once famously called Adaboost with trees "the best off-the-shelf classifier in the world" (NIPS workshop, 1996). Lienhart et al [11] explored the use of CART as a weak learner in the CoBE framework, but trees only produced moderate improvements over stumps in their experiments and did so only for low false positive rates, where the corresponding detection rate is less than 85%. In contrast, we find that CART trees result in significant improvements to the classification performance at all false positive rates. We hypothesize that this may be due in part to our strategy of adjusting both stage thresholds and post-processing when producing our ROC curves.

Much of the early research on the CoBE architecture focused on the boosting algorithm. In their 2002 paper, Viola and Jones observe that the goal of a stage in the cascade is not to minimize error, but to retain very high detection rates, while accepting modest false positive rates if necessary [6]. They propose Asymmetric Adaboost, which changes the Reweigh-Examples routine to keep most of the weight on the positive examples (instead of treating positive and negative examples equally), ensuring that a high percentage is detected by each weak classifier. The problem of asymmetric learning is also addressed in [12], which introduces the Linear Asymmetric Classifier algorithm, a method to re-weigh hypotheses after they have been selected by other means.

Li and Zhang have applied another alternative boosting algorithm to face detection in their paper on FloatBoost [13], which instead of greedily adding hypotheses to the ensemble allows backtracking to eliminate the less useful or even hurtful hypotheses. In other respects, the algorithm proceeds as RealBoost.

Liu and Shum [14] found that using KL-boost combined with weak classifiers based on histograms of 1D projections in feature space improved detection performance over the original approach. However, it is not clear whether it is the changes to the weighing scheme or the means of forming the weak hypotheses that is critical to the improvement.

A more radical departure from the Learn-CoBE routine is due to Xiao et al [15]. Inspired by the observation that the operating point of a stage may not minimize error, they allow the hypothesis formed by the minimum error threshold of the previous stage to play the role of a weak hypothesis in the next stage of the cascade. Having thus produced a cascaded detector, they convert it to a single weighted voting scheme and train an SVM to relearn the confidence (vote) weights.

Others have changed the feature set while keeping the other key aspects of the CoBE architecture[11, 16]. A more detailed description of our work can be found in our technical report [17]. For a more comprehensive survey of face detection see [18].

3 Cascade Learning

Two of the most important decisions in building a cascade of boosted ensembles are:

1. When to stop training a stage and move on to the next one.
2. How to balance the detection versus false positive trade-off within a stage.

In terms of our LEARN-COBE algorithm, these decisions are determined by the function FIND-BEST-THRESHOLD, which chooses θ_i, fixing the stage's operating point, and by the function PREDICT-COST which determines when to move on to the next stage of the cascade.

3.1 Fixed Stage Goal

The standard approach outlined in [1, 6] is to choose a goal operating point $\langle F_g, D_g \rangle$ (a false positive and detection rate pair) and then take its Lth root to obtain $\langle f_g, d_g \rangle$, where L is the intended number of stages in the cascade. Each stage is constrained to achieve one of f_g or d_g (typically f_g works better) on a set of validation examples that have been accepted by all previous stages of the cascade. The training of the stage terminates when either the other goal criterion is achieved or the maximum number of boosting iterations is exceeded.

This goal-based strategy leaves something to be desired, however. First, it rigidly fixes the number of stages in the cascade before any training is done. Second, it does not permit any trade-off between the detection and false positive rates within the stages. For instance, when selecting the threshold of a stage, one might be able to significantly improve the false positive rate at a small expense to the detection rate, improving the chances of meeting the goal criteria. The extra leeway on the false positive criterion might also be used at a later stage to improve a stage's detection at the expense of the false positive rate. By fixing one element of the operating point, this strategy precludes taking advantage of such trade-offs.

3.2 Cascade Learning with Beta Variables

A key element of our approach is that the algorithm views the performance of the cascade $\langle F, D \rangle$ as a random variable and treats the empirical results on validation data for the individual stages, $\{\hat{f}_i\}$ and $\{\hat{d}_i\}$, as evidence. A statistical model estimates the distribution of full cascade operating points, and each stage is trained to use the minimum number of features that ensure that the probability of meeting the performance goals is sufficiently high.

The key assumption underlying the statistical model is that the results on the validation data for the current stage can be repeated at all subsequent stages. That is, for any $\langle \hat{f}_i, \hat{d}_i \rangle$ pair obtained by varying θ_i, it is possible to train the $(i+1)$th stage and choose θ_{i+1} such $\langle \hat{f}_{i+1}, \hat{d}_{i+1} \rangle = \langle \hat{f}_i, \hat{d}_i \rangle$. We call this the "repeatability assumption". It is important to note that a similar assumption is implied in the fixed stage goal framework, where it is assumed that a particular operating point will be achieved in each stage. Although, the repeatability assumption is not strictly true in practice, it provides a guiding principle for applying our statistical model during training. The advantage of this model is that it affords a principled and practical way to make detection and false positive rate trade-offs in the individual stages.

The inputs to our new method are:

1. A goal operating point for the entire cascade $\langle F_g, D_g \rangle$.
2. A ratio η that reflects the relative importance of the false positive and detection criteria.
3. A maximum number of stages L.

The cascade learner then builds the fastest detector it can while achieving the goal performance with high probability.

Cost Function. Because a reasonable goal might not be known a priori, the algorithm must be robust to unattainable goals and produce results that are as close as possible. Depending on the attainability of the goal, therefore, we adjust our cost function. For simplicity, assume that $\eta > 1.0$, meaning that the false positive criterion is more important. We consider the following cases

1. If $\Pr[D < D_g] < \gamma$ and $\Pr[F > F_g] < \gamma$,
$$\text{cost} = \Pr[D < D_g] + \eta \Pr[F > F_g].$$
2. Else, if $\Pr[F > F_g] < \gamma$, then cost $= 2 + \eta - D$.
3. Otherwise, cost $= 2 + \eta + F$.

The first cost function is suitable when both goals are attainable with some substantial probability γ (0.95 was used our experiments). However, when this is not possible, then the function provides no incentive to trade a small decrease in the false positive rate for a large improvement in the detection rate (an analogous statement holds if $\eta < 1.0$, giving detection greater importance). Therefore, if both criteria cannot be met with probability γ, then we constrain the false positive rate to be met with probability γ and maximize the detection rate. Finally, if the criterion for false positive rate cannot be met with probability γ, we simply minimize the false positive rate. Typically, this means that the false positive rate is reduced to zero, effectively terminating the training process.

Cost Prediction. Minimizing this cost function requires the ability to compute $\Pr[D > D_g]$ and $\Pr[F < F_g]$. We will only treat the detection criterion, because the false positive one is analogous. Consider the likelihood $\Pr[\hat{d}_i | d_i]$, where $\hat{d}_i$ is the measured detection rate over M positive examples. Given the true detection rate d_i, the probability of m out of M examples being detected is just the binomial distribution
$$\binom{M}{m} (1 - d_i)^{M-m} d_i^m.$$
Taking a uniform prior $\Pr[d_i]$ over $[0, 1]$ and applying Bayes rule gives
$$\begin{aligned} \Pr[d_i | m, M] &= \frac{\Pr[m | d_i, M] \Pr[d_i]}{\int_0^1 \Pr[m | p, M] \Pr[p] dp} \\ &= \frac{(1 - d_i)^{M-m} d_i^m}{\int_0^1 (1 - p)^{M-m} p^m dp}, \end{aligned}$$
which is precisely the beta distribution with parameters $m + 1$ and $M - m + 1$.

Assume that the cascade has already been trained through stage i and that we are predicting the cost if the measured operating point of the next stage is $\langle \hat{f}_{i+1}, \hat{d}_{i+1} \rangle$.

PREDICT-COST-SAMPLE *maintains a set of sampled operating points for the currently trained cascade $\{\langle F_i^k, D_i^k \rangle\}_{k=1}^K$. All measurements are made with validation sets of M negative examples and the same number of positive examples.*

procedure PREDICT-COST-SAMPLE()
  for $j = i + 1$ to N **do**
    for $k = 1$ to K **do**
      $F_j^k \leftarrow F_{j-1}^k \cdot \beta_{\hat{f}_i}$, where $\beta_{\hat{f}_i}$ is a random beta deviate with parameters $\hat{f}_i M + 1$ and $(1 - \hat{f}_i)M + 1$.
      $D_j^k \leftarrow D_{j-1}^k \cdot \beta_{\hat{d}_i}$, where $\beta_{\hat{d}_i}$ is a random beta deviate with parameters $\hat{d}_i M + 1$ and $(1 - \hat{d}_i)M + 1$.
    end for
    $G_f \leftarrow |\{k : F_j^k > F_g\}|/M$
    $G_d \leftarrow |\{k : D_j^k < D_g\}|/M$
    $\text{cost}_j \leftarrow \text{COST}(G_f, G_d)$.
  end for
  return $\min_j \text{cost}_j$.

Therefore, conditioned on the validation measurements, D is the product of beta variables. The exact distribution only admits a clean analytic form in a few specialized cases [19], but it can easily be approximated. One strategy is to sample from the distribution for D by taking a sample from the distribution d_i for each stage and taking their product. The quantity $\Pr[D > D_g]$ can be estimated by counting the fraction of samples greater than D_g. This method is used in the PREDICT-COST-SAMPLE procedure. A final set of samples for a fully trained cascade is shown in Fig. 1.

Given the ability to estimate the cost for a (partially) trained cascade, we now describe its use in stage training. It is here that we apply the repeatability assumption; i.e., if we can achieve $\langle \hat{f}_i, \hat{d}_i \rangle$ on a validation set for the current stage, then we assume that we can achieve the same result for all subsequent stages. Therefore, as we are training the ith stage, we use the results on the validation set to estimate $\langle \hat{f}_j, \hat{d}_j \rangle$ for all previous stages ($j < i$), but we use the results for the ith stage on validation data for any subsequent stages ($j > i$). The operating point having the lowest cost according to this estimate is chosen for each stage, as shown in the FIND-BEST-THRESHOLD procedure.

3.3 Discussion

The main advantage of our new approach over the fixed stage goal approach is that it allows subtle tradeoffs between detection and false positive rates in the stages. Moreover, it can "remember" past trade-offs to help decide whether a new trade-off will improve the chances of achieving the cascade's goal operating point. Note that though we specify a maximum number of stages, we do not

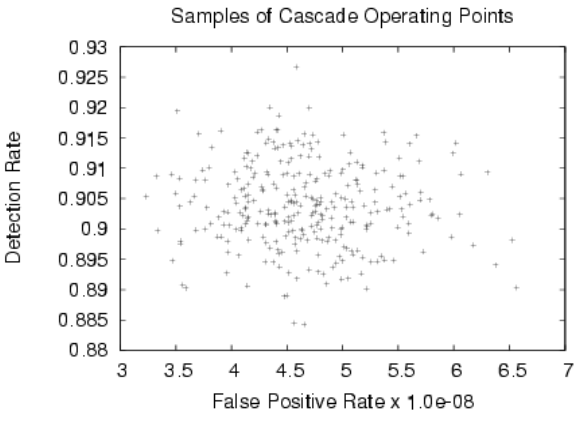

Fig. 1. Samples generated by PREDICT-COST-SAMPLE of the operating point for a fully trained twenty-stage cascade. The accumulation of error is significant even though a validation set of 1000 examples was used for both the positive and negative classes.

specify a minimum. If the learner predicts better performance with fewer stages, then it will plan for fewer stages.

As shown in Fig. 1, the variances in the distributions of F and D are significant for a twenty-stage cascade, even when one thousand examples are used at every stage. It is possible to account for this effect in the fixed stage goal approach simply by setting more ambitious goals than are necessary, so that even if the validation results are too optimistic, the desired performance may nevertheless be achieved. If forty stages are used instead of twenty, however, then the additional accumulation of error will change the distribution, and a new set of more ambitious goals may be required. Because we explicitly model the accumulation of errors, no such parameter retuning is necessary in our approach, making it well-suited for comparative studies.

To demonstrate the effectiveness and robustness of our improved cascade learning algorithm, we have conducted a set of experiments in which we automatically trained 35 detectors using a single set of parameters. This set of experiments ranges from a cascade using four level deep CART trees that achieves state of the art performance (see Sect. 5) to a cascade where the feature pool was reduced to 200 randomly selected features (see Sect. 4). Results for all of our experiments can be found in [17].[1]

4 Feature Selection

The primary computational cost in training the stage classifiers is that in every round of boosting the WEAK-LEARN routine examines every example for every

[1] Complete results and code are also available at http://www.cc.gatech.edu/cpl/cobe.

feature. Since reducing the example corpus weakens the generalization, the alternative of reducing the feature pool via the FILTER-FEATURES routine is an attractive option.

To actually improve the training time, however, the filtering algorithm itself must be faster than Adaboost. Unfortunately, few filtering algorithms offer an asymptotic improvement in training time. Nevertheless, asymptotically equivalent methods often admit implementation speed-ups, which make the actual run-time faster than the worst-case analysis time would indicate. Moreover, because Adaboost's greedy selection of features is not optimal, limiting the feature pool available to Adaboost may actually improve the results. The idea is that Adaboost may produce a better classifier when it is presented with a small set of features, all of which are good, rather than a large set containing these same good features in addition to many spurious ones.

For purposes of this discussion, therefore, we divide filtering techniques into two broad categories:

Fast Filters: This category consists primarily of ranking schemes which examine each feature once and sort according to some measure of the feature's discriminative power. These filters are typically much faster than Adaboost and run in $O(|F| \log |F|)$ time. From this category, we test random selection and ranking by mutual information. For the latter, we choose a feature threshold that maximizes the mutual information between the resulting binarized feature and the class label, and then select the features that have the most mutual information with the class label as individual features.

Slow Filters: This category includes methods that examine each feature in F before choosing the next feature to add to the selected pool F'. These filters run in $O(|F'||F||E|)$ time and are about as fast as Adaboost with a thresholding weak learner. From this category, we use the Conditional Mutual Information Maximization method of [20, 21] and Forward Feature Selection [10].

Notice that the running times given above assume that the examples have been sorted by their feature value for every feature in a precomputation step.[2] With this strategy, the evaluation of a feature, either for selection or for use in a weak classifier, can be performed in $O(|E|)$ time, where E is the set of examples. It is also important to realize that although these filtering methods sometimes choose a threshold value for the feature during the selection, the original feature values are retained for the boosting or ensemble learning phase of the training process.

In this context, our hope would be that filters from the first category would improve the training time significantly without diminishing the quality of the results and that filters from the second category would improve the quality of the results and offer a modest improvement in training time.

[2] This pre-sorting strategy has been previously noted in [22] and is explained in more detail in [23] and [17].

4.1 Analysis

We evaluate these methods by training a full cascade using the learning algorithm of Sect. 3.2 with a fixed set of parameters. To evaluate the classification performance, we apply the detector to the CMU-MIT data set and average the detection rate over a range of 0-130 false positives.[3] This roughly corresponds to the area under curve measure used for traditional ROC curves.

Fast Filters. Each of the fast methods was used to reduce the feature pool by 90% and 99% during the training of several detectors. As shown in table 1, in both cases random selection (RND) gives comparable performance to the ranking method (RANK). At first, this may seem counter-intuitive. The ranking method does, after all, include the most discriminative features. How can a random selection of features produce detectors that perform just as well or better? The answer is the well known redundancy problem [24]. The "best" features tend to misclassify the same examples, making it difficult for Adaboost to learn an ensemble of hypotheses that classifies these examples correctly. We discuss this phenomenon in greater detail in [17].

Table 1. Feature filtering results grouped by the number of features made available to the weak learner (Final Pool). Notice that the random filtering outperforms the ranking filter. Although CMIM and FFS are better than random filtering, they do not outperform the inherent feature selection strategy of Adaboost.

Filter	Initial Pool	Final Pool	Avg. Detection rate for [0-130] False Positives on CMU-MIT
RND	134736	13473	0.889
RANK	134736	13473	0.872
RND	134736	1347	0.874
RANK	134736	1347	0.834
RND	13473	200	0.829
CMIM	13473	200	0.870
FFS	13473	200	0.860

Slow Filters. To assess the asymptotically slower methods, Conditional Mutual Information Maximization (CMIM) and Forward Feature Selection (FFS), we first randomly selected 10% of the features and then used the methods to filter down to 200 features. For a baseline comparison we also trained a detector with 200 randomly selected features. Both the FFS and CMIM cascades produce ROC curves comparable to the one produced by a random 10% selection of features. That is, the detectors perform as well as they would if no filtering had been applied at all. Thus, although these methods offer a modest (factors of 2 or 3) improvement in training time, they do not outperform the greedy selection naturally employed by Adaboost.

[3] This upper bound of 130 false positives represents an average of one false positive per image.

5 Weak Learning

Although thresholding on a single feature has been the dominant practice in CoBEs for object detection, Adaboost does not restrict how the weak learning takes place. The thresholding strategy may be efficient in terms of training or execution time, but it seems doubtful that such a simple weak learner would give the best results. We therefore explore the use of CART-based weak hypotheses, which we found to significantly improve the cascade performance.

Our experiments show that CART-based detectors offer improved detection rates with only small drops in speed. The ROC curve of Fig. 2 shows the improvement coming from using CART trees of depth 2, 4, and 6, as opposed to stumps (i.e. threshold-based hypotheses) when discrete Adaboost is used. A more comprehensive set of results for RealBoost and GentleBoost can be found in [17]. These results are consistent with our findings for discrete Adaboost.

Table 2 gives a comparison to several other published cascade training methods. While a comprehensive comparison would include testing speed as well as classification performance, these numbers suggest that the current method produces results which are comparable to published work that is based on substantial mod-

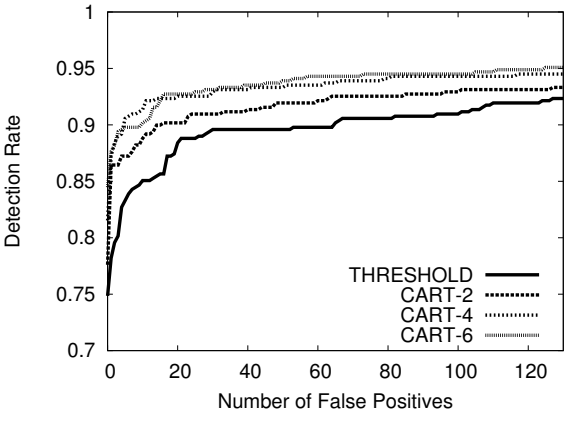

Fig. 2. CART depths up to 4 significantly and consistently improve performance

Table 2. A comparison of detection rates on the CMU-MIT data set for several standard detectors

	False Positives							
Detector	6	10	31	46	50	65	78	95
Viola-Jones [1]	–	0.761	0.884	–	0.914	0.920	0.921	0.929
Viola-Jones [1] (voting)	–	0.811	0.897	–	0.921	0.931	0.931	0.932
Luo [8]	0.866	0.874	0.903	–	0.911	–	–	–
Li-Zhang [13]	–	0.836	0.902	–	–	–	–	–
Schneiderman [25]	0.897	–	–	0.957	–	–	–	–
CART-4 w/ Realboost	0.891	0.905	0.931	0.935	0.935	0.943	0.948	0.951

ifications to the basic Adaboost learning method. Our results show that the basic method can yield excellent performance if stronger weak hypotheses are employed. Moreover these results can be obtained without hand-tweaking cascade parameters during training, as a consequence of our automatic global training method. Promising directions for future studies include an evaluation of these methods from the standpoint of testing speed and the use of our global training method of Sect. 3.2 in conjunction with previously-published stage learning algorithms.

6 Conclusion

We have described a novel algorithm for fully-automatic cascade training based on a probabilistic prediction of cascade performance. This method can take advantage of favorable trade-offs of detection and false positive rates for the individual stage and removes much of the guess-work associated with training cascades of boosted ensembles in the past. Because it takes into account the accumulation of error in the estimates of the overall cascade performance, it is well-suited for controlled experiments comparing cascaded detectors which are trained using a wide variety of stage learning algorithms.

A major barrier to the wider use of cascades of boosted ensembles is that they take a long time to train. We explore feature filters which can produce a moderate speed-up by reducing the set of features available to the ensemble learner.

Finally, we show that although thresholding on single features to form weak hypotheses may reduce training time and produce a faster detector, combining Adaboost with CART-based weak learning can significantly improve the detector's performance.

Acknowledgements

This material is based upon work which was supported in part by the National Science Foundation under NSF Award IIS-0133779.

References

1. Viola, P., Jones, M.J.: Robust real-time face detection. Int. J. Comput. Vision **57**(2) (2004) 137–154
2. Viola, P.A., Jones, M.J., Snow, D.: Detecting pedestrians using patterns of motion and appearance. In: Proc. ICCV. Volume 2. (2003) 734–741
3. Schneiderman, H., Kanade, T.: Object detection using the statistics of parts. Int. J. Comput. Vision **56**(3) (2004) 151–177
4. Freund, Y., Schapire, R.E.: A decision-theoretic generalization of on-line learning and an application to boosting. J. Comput. Syst. Sci. **55**(1) (1997) 119–139
5. Breiman, L., Friedman, J., Olshen, R., Stone, C.: Classification and Regression Trees. Wadsworth and Brooks, Monterey, CA (1984)
6. Viola, P., Jones, M.: Fast and robust classification using asymmetric AdaBoost and a detector cascade. In: NIPS 14. (2002) 1311–1318

7. Sun, J., Rehg, J.M., Bobick, A.F.: Automatic cascade training with perturbation bias. In: CVPR (2). (2004) 276–283
8. Luo, H.: Optimization design of cascaded classifiers. In: CVPR (1). (2005) 480–485
9. Sochman, J., Matas, J.: Waldboost-learning for time constrained sequential detection. In: CVPR (2). (2005) 150–157
10. Wu, J., Rehg, J.M., Mullin, M.D.: Learning a rare event detection cascade by direct feature selection. In: NIPS 16. (2004) 1523–1530
11. Lienhart, R., Kuranov, A., Pisarevsky, V.: Empirical analysis of detection cascades of boosted classifiers for rapid object detection. In: Pattern Recognition LNCS 2781. (2003) 297–304
12. Wu, J., Mullin, M., Rehg, J.: Linear asymmetric classifier for cascade detectors. In: Proc. 22nd International Conference on Machine Learning. (2005) 993–1000
13. Li, S.Z., Zhang, Z.Q.: Floatboost learning and statistical face detection. IEEE Trans. on PAMI **26**(9) (2004) 1112–1123
14. Liu, C., Shum, H.Y.: Kullback-leibler boosting. In: CVPR (1). (2003) 587–594
15. Xiao, R., Zhu, L., Zhang, H.J.: Boosting chain learning for object detection. In: Proc. ICCV. Volume 1. (2003) 709–715
16. Levi, K., Weiss, Y.: Learning object detection from a small number of examples: The importance of good features. In: CVPR (2). (2004) 53–60
17. Brubaker, S.C., Wu, J., Sun, J., Mullin, M.D., Rehg, J.M.: On the design of cascades of boosted ensembles for face detection. Technical Report GIT-GVU-05-28, Georgia Institute of Technology (2005)
18. Yang, M.H., Kriegman, D.J., Ahuja, N.: Detecting faces in images: A survey. IEEE Trans. on PAMI **24**(1) (2002) 34–58
19. Gupta, A.K., Nadarajah, S., eds.: Handbook of Beta Distribution and its applications. Marcel Dekker, Inc. (2004)
20. Vidal-Naquet, M., Ullman, S.: Object recognition with informative features and linear classification. In: Proc. ICCV. (2003) 281–288
21. Fleuret, F.: Fast binary feature selection with conditional mutual information. Journal of Machine Learning Research **5** (2004) 1531–1555
22. Opelt, A., Fussenegger, M., Pinz, A., Auer, P.: Weak hypotheses and boosting for generic object detection and recognition. In: ECCV (2). (2004) 71–84
23. Grossmann, E., Kale, A., Jaynes, C.: Towards interactive generation of "ground-truth" in background subtraction from partially labeled examples. In: Proc. ICCV VS-PETS workshop. (2005)
24. Guyon, I., Elisseeff, A.: An introduction to variable and feature selection. Journal of Machine Learning Research **3** (2003) 1157–1182
25. Schneiderman, H.: Feature-centric evaluation for efficient cascaded object detection. In: CVPR (2). (2004) 29–36