High Detection-rate Cascades for Real-Time Object Detection

Hamed Masnadi-Shirazi, Nuno Vasconcelos
Department of Electrical and Computer Engineering,University of California San Diego
San Diego, CA 92093
hmasnadi@ucsd.edu, nuno@ece.ucsd.edu

Abstract

A new strategy is proposed for the design of cascaded object detectors of high detection-rate. The problem of jointly minimizing the false-positive rate and classification complexity of a cascade, given a constraint on its detection rate, is considered. It is shown that it reduces to the problem of minimizing false-positive rate given detection-rate and is, therefore, an instance of the classic problem of cost-sensitive learning. A cost-sensitive extension of boosting, denoted by asymmetric boosting, *is introduced. It maintains a high detection-rate across the boosting iterations, and allows the design of cascaded detectors of high overall detection-rate. Experimental evaluation shows that, when compared to previous cascade design algorithms, the cascades produced by asymmetric boosting achieve significantly higher detection-rates, at the cost of a marginal increase in computation.*

1. Introduction

Significant attention has been devoted to the problem of real-time object detection in recent years. The seminal contribution was the cascaded object detector of Viola and Jones (VJ) [16]. It achieves real time detection with performance comparable to that of previous state of the art methods. Although its success is due to a combination of innovative ideas, such as using a large set of very simple features, efficient computation through integral images, and the recourse to weak classifiers and boosting, one of the greatest assets of the VJ detector is the adoption of a cascaded classifier architecture[16]. In particular, the detector is implemented as a sequence of classifiers of high detection and low false positive rate. Because most image locations can easily be identified as not containing the target object, this allows the rejection of non-targets with a very small number of feature evaluations, enabling extreme computational efficiency. In the original VJ algorithm, a standard implementation of AdaBoost [5] is used to identify the set of discriminant features which, along with a threshold, make up each stage of the cascaded detector. The overall goal is to meet pre-specified detection and false-positive rates for the whole cascade, while maximizing detection speed.

While cascades designed with the procedure of [16] are extremely fast, the lack of an optimal method for cascade design is problematic in two ways. First, it makes this design dependent on manual supervision. Because the stages of the cascade are thresholded combinations of weak learners, there is a need to search for the configuration (number of learners and threshold) of each stage. This usually requires trial and error and can be very time consuming, compromising the viability of cascaded detectors as a practical solution for generic object detection. Second, the resulting cascades are inevitably sub-optimal. This is particularly problematic because the objects to detect occur rarely in the images where detection is performed, and the cost of loosing each occurrence is usually large. The problem is compounded by the fact that the missed detections of the various stages accumulate, and it is very difficult to design cascades of high overall detection rate.

The limitations of the original VJ design have motivated a number of enhancements in the recent years. Most of these contributions, while improving various aspects of the original design - e.g. by proposing improved boosting algorithms (or other feature selection procedures) [8, 2], embedded cascade structures [19, 18], cost-sensitive extensions for the design of cascade stages [10, 7], or threshold post-processing [1, 9] - have not addressed the problem of how to optimally design the whole cascade. This problem has only recently started to receive some attention [2, 13], and interesting theoretical formulations have been proposed for its solution, but their translation into practical algorithms requires assumptions or approximations that may not always hold. Perhaps due to this, these formulations have not yet proved effective at producing high detection-rate cascades.

In this work, we propose a new cascade design algorithm that addresses this problem. We introduce a novel formulation for cascade design, which poses the problem as one of jointly minimizing computational cost and false positive rate for a given detection rate. We then show that this reduces to the problem of minimizing false positive given detection rate, i.e. the classic problem of cost-sensitive learning. We next introduce a cost-sensitive extension of boosting that maintains a high detection rate across the boosting iterations, and allows the design of single-feature embedded cascades with high overall detection rate. Experimental evaluation shows that, when compared to the state-of-the-art, these cascades do achieve significantly higher detection rates, at the cost of a marginal increase in computation.

2. Relation to previous work

A number of extensions to the VJ algorithm have been recently proposed in the literature. Some, e.g. using floating instead of sequential feature search [8], optimizing thresholds after cascade designed [1, 9], or using the outputs of the cascade stages as input space to the design of a final classifier [19], could be combined with what is discussed here. A number of researchers proposed the idea of embedded cascades [19, 18, 13]. These are cascades where each stage is part of the subsequent one, enabling the use of computationally efficient stages without compromise of detection performance. While, like these methods, our solution produces embedded cascades, it is based on the optimal design of the overall cascade, rather than individual stage design.

It has also been realized that one of the major problems of the original VJ design is the reliance on the AdaBoost algorithm. Because the latter is not cost-sensitive, it forces the manipulation of thresholds of the individual cascade stages, which are thus forced to operate at detection and false-positive rates for which they have not been optimized. This has motivated various authors to adopt cost-sensitive extensions of AdaBoost [10, 7]. These are, however, mostly heuristic, and have only been applied to the design of individual stages. In result, there are no guarantees of 1) optimality, in a cost-sensitive sense, for the whole cascade, or 2) ability to achieve a high overall detection rate. We propose a principled cost-sensitive extension of AdaBoost, which designs the whole cascade in a single step, under an explicit constraint on the detection rate.

Recently, some proposals have started to emerge in the area of optimal global cascade design. Waldboost [13] is based on an extension of the classic work by Wald on optimal sequential decision making [13]. Although Wald's underlying framework is principled, the optimal solution is impossible to compute, and approximations are needed. Due to this, the resulting algorithms give guarantees on the sum of false-positive and miss rate, but not on each of these terms individually (see, e.g., Theorem 2 of [13]). This compromises its ability to design high-detection rate cascades. Furthermore, Wald's procedure is only optimal for independent observations. Waldboost is a boosting-based extension that addresses this limitation, but requires a number of assumptions that only hold in the asymptotic regime of infinite length cascades. It is, therefore, not clear how the intermediate decisions are related to Wald's theory, and the difficulty to maintain high detection rates is compounded. The net result is a rather non-intuitive design procedure which, for example, relies on the specification of a zero false-positive rate for cascade design, when the goal is to produce cascades with a constraint on the detection rate [13].

In [2], the globally optimal design of detector cascades is addressed in the VJ framework, i.e. by designing each stage individually. False positive and miss rates are treated as random variables, and a sampling based procedure is designed to predict the best operating point for each stage, given the operating points of the previous stages. This relies on an assumption of repeatability, i.e. that by simple observation of previous points it is possible to 1) predict the best operating point for the next, and 2) actually design it so as to achieve that point. It is unclear how closely this assumption holds in practice. Overall, none of the above methods has demonstrated the ability to produce cascades of high detection-rate.

3. Optimal cascade design

To address this problem, we propose a new framework for cascade design. In this section, we define optimality in a sense close to that of VJ and show that optimal design of embedded cascades reduces to cost-sensitive learning.

3.1. Definitions

A cascaded detector is a decision function of the form

$$h(\mathbf{x}) = s_1(\mathbf{x}) \wedge \ldots \wedge s_K(\mathbf{x}) \tag{1}$$

where $s_i(\mathbf{x})$ are detectors, denoted as *cascade stages*, and the $\wedge$ operator implements an early rejection of the examples which are classified as negatives. The false-positive rate of the i^{th} detector is denoted by f_i, its detection rate by d_i, the false-positive rate of the whole cascade by F and its detection rate by D. An assumption, underlying all previous cascade design algorithms, is that errors committed by successive stages are independent. While not necessarily realistic, this assumption makes the problem significantly more tractable, and is also adopted in this work. It follows that

$$F = \prod_{i=1}^{K} f_i \tag{2}$$

$$D = \prod_{i=1}^{K} d_i, \tag{3}$$

and it is possible to show that the expected complexity of the whole cascade is

$$\begin{aligned} E[C] &= (1-\pi)\sum_{i=1}^{K} C_i(1-f_i)\sum_{j=1}^{i-1} f_j \\ &\quad +\pi\sum_{i=1}^{K} C_i(1-d_i)\sum_{j=1}^{i-1} d_j, \end{aligned}$$

where

$$C_i = \sum_{j=1}^{i} c_i \tag{4}$$

and c_i is the cost of evaluating the i^{th} stage and π the probability of occurrence of the object of interest.

3.2. Optimal cascades

We formulate the problem of optimal cascade design as the joint minimization of the false-positive rate and expected cascade complexity, given a target detection rate D^*,

$$h^* = \arg\min_{h\in\mathcal{H}} \mathcal{F}(F, E[C]) \tag{5}$$

$$D = D^* \tag{6}$$

where $\mathcal{F}$ is some function monotonically increasing in F and $E[C]$, and $\mathcal{H}$ is the set of cascades under consideration. Similarly to VJ, we consider the set of constant detection-rate cascades

$$\mathcal{H} = \{h | d_i = (D^*)^{1/K}\}, \tag{7}$$

i.e. whose stages share a common detection rate. It is possible to show that, for any $h \in \mathcal{H}$ and $k \in \{1, \dots, K\}$,

$$\frac{\partial E[C]}{\partial f_k} \geq 0 \ \forall f_k \geq 0 \tag{8}$$

$$\frac{\partial F}{\partial f_k} \geq 0 \ \forall f_k \geq 0, \tag{9}$$

i.e. both the expected cascade complexity and its false-positive rate, are non-decreasing functions of f_k. Hence, for any monotonically increasing $\mathcal{F}$, the optimal solution is to minimize the false-positive rate of each stage, i.e. by choosing the cascade h^* with stage false-positive rates

$$f_k^* = \arg\min_{f_k} f_k \tag{10}$$

$$d_k = (D^*)^{1/K} \tag{11}$$

for all $k \in \{1, \dots, K\}$.

3.3. Embedded cascades

The cascade stages are binary decision rules

$$s_i(\mathbf{x}) = sgn[g_i(\mathbf{x})] \tag{12}$$

where $g_i(\mathbf{x})$ are real functions, here referred to as predictors. Usually, these are linear combinations of weak learners $\phi_{i,j}(\mathbf{x})$,

$$g_i(\mathbf{x}) = \sum_{j=1}^{J_i} \alpha_{i,j}\phi_{i,j}(\mathbf{x}), \tag{13}$$

where $\phi_{i,j}(\mathbf{x})$ is a thresholded feature response, also known as decision stump, and J_i is the number of weak learners combined by $g_i(\mathbf{x})$.

Embedded cascades (also known as nested [18] or chained [19] cascades) are implemented with embedded predictors. These are predictors such that, for all i, the terms of $g_i(\mathbf{x})$ are also terms of $g_{i+1}(\mathbf{x})$. A single-feature embedded cascade is an embedded cascade where the predictor of each stage adds one weak learner to the predictor of its predecessor,

$$g_{i+1}(\mathbf{x}) = g_i(\mathbf{x}) + \alpha_{i+1}\phi_{i+1}(\mathbf{x}) \tag{14}$$

$$= \sum_{j=1}^{i+1} \alpha_j\phi_j(\mathbf{x}) \tag{15}$$

with $g_0(\mathbf{x}) = 0$. The associated cascade is equivalent to introducing a rejection point after each weak learner of a decision rule produced by boosting. In what follows, we concentrate on single feature embedded cascades, but most results hold for embedded cascades of any configuration.

3.4. Optimal embedded cascades

Consider a sequence of predictors, $g_k^*(\mathbf{x})$, optimal in the sense of (10) for $k = 1, \dots, i$. By definition, $g_i(\mathbf{x})$ is the sum of i terms of the form of (15) such that

$$s_i^*(\mathbf{x}) = sgn[g_i^*(\mathbf{x})]$$

achieves the smallest possible false positive rate at detection rate $(D^*)^{1/K}$. The optimal design of $g_{i+1}(\mathbf{x})$ requires selecting $(\alpha_{i+1}, \phi_{i+1})$ such that

$$s_{i+1}(\mathbf{x}) = sgn[g_i^*(\mathbf{x}) + \alpha_{i+1}\phi_{i+1}(\mathbf{x})] \tag{16}$$

achieves the smallest possible false-positive rate at detection-rate $(D^*)^{1/K}$.

To gain some insight on how this may be solved, it is useful to recall the interpretation of boosting as gradient descent in the functional space $\mathcal{S}$ of linear combinations of weak learners [12]. Under this interpretation, the functions $g_k(\mathbf{x})$ are elements of $\mathcal{S}$, and a cost functional $\mathcal{C}(g)$ is defined in this space. Boosting can then be shown to perform a search for the best $(\alpha_{i+1}, \phi_{i+1})$, by taking a step of gradient descent: it selects the weak learner ϕ_{i+1} as the negative of the functional derivative of $\mathcal{C}$, evaluated at $g_i(\mathbf{x})$,

$$\phi_{i+1} = -\nabla\mathcal{C}(g_i(\mathbf{x}))$$

and the scaling constant α_{i+1} through a line search along the descent direction ϕ_{i+1} [12]. The cost functional is the empirical estimate of the margin

$$\mathcal{C}(g_i(\mathbf{x})) = \frac{1}{N}\sum_{k=1}^{N} e^{-y_k g(\mathbf{x}_k)} \tag{17}$$

from a training set $\mathcal{D} = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_N, y_N)\}$, also known as the boosting loss, or exponential loss. As we will see in the next section, this guarantees that the optimal solution converges to the minimum probability of error rule, as the weak learners are added to the ensemble.

This process can be equally applied to the solution of (16). There are only two differences, due to the fact that, instead of minimizing probability of error, we are now interested in minimizing false-positive error given a constraint on detection rate. The first is that the cost functional should offer guarantees of convergence to the optimal, detection rate constrained, decision rule. These guarantees should be identical to those offered by boosting with respect to convergence to the minimum probability of error rule. The second is that the detection rate is met at all iterations of the boosting process. In summary, we need an extension of boosting that supports a constraint on detection rate. This is usually referred as cost sensitive boosting.

4. Cost-sensitive boosting

A cost-sensitive detector $s(\mathbf{x})$ is a detector that assigns unequal importance to the two error types. An optimal cost-sensitive detector is designed by defining a loss function of the form

$$L(\mathbf{x}, y) = \begin{cases} 0, & \text{if } s(\mathbf{x}) = y \\ C_2 & \text{if } y = -1 \text{ and } s(\mathbf{x}) = 1 \\ C_1 & \text{if } y = 1 \text{ and } s(\mathbf{x}) = -1 \end{cases}, \tag{18}$$

where $C_i > 0$, and searching for the detector of minimum risk, or expected value of this loss The optimal solution is the well known Bayes decision rule [3], which declares $s(\mathbf{x}) = 1$ when

$$\log\left(\frac{P_{Y|\mathbf{X}}(1|\mathbf{x})}{P_{Y|\mathbf{X}}(-1|\mathbf{x})}\right) > \log\left(\frac{C_2}{C_1}\right) \tag{19}$$

and $s(\mathbf{x}) = -1$ otherwise. This can be summarized as

$$s_T(\mathbf{x}) = sgn\left[\log\left(\frac{P_{Y|\mathbf{X}}(1|\mathbf{x})}{P_{Y|\mathbf{X}}(-1|\mathbf{x})}\right) - T\right] \tag{20}$$

with $T = \log\frac{C_2}{C_1}$.

The relation between boosting and minimum probability of error classification follows from a result of Friedman [6], who has shown that the exponential loss

$$E[\exp(-yg(\mathbf{x}))] \tag{21}$$

is minimized by the symmetric logistic transform of $P_{Y|\mathbf{X}}(1|\mathbf{x})$,

$$g^*(\mathbf{x}) = \frac{1}{2}\log\frac{P_{Y|\mathbf{X}}(1|\mathbf{x})}{P_{Y|\mathbf{X}}(-1|\mathbf{x})}. \tag{22}$$

Since, for a large training sample $\mathcal{D}$, the empirical loss of (17) converges to (21), it follows that the result of the gradient descent performed by boosting is the optimal predictor $g^*(\mathbf{x})$, in the minimum probability of error sense.

4.1. Why boosting is not enough

The statement that the predictor found by boosting converges to (22), and the relation between this and the optimal cost-sensitive detector of (20) suggests a natural answer to the design of detector cascades. Simply compute each stage with the standard boosting algorithm, and replace the decision rule $sgn[g^*(\mathbf{x})]$ by $sgn[g^*(\mathbf{x}) - T]$. Given (22), this should be equivalent to (20), producing the optimal cost-sensitive detector. This is the procedure proposed by VJ, and used by all previous cascade design algorithms, including the optimal methods of [13, 2]. The only variations are the approaches used to produce the thresholds.

While all these statements are correct, they are predicated on the convergence of the boosted predictor to (22) *everywhere*. It should, however, be noted that this convergence does not necessarily hold for a finite number of iterations. In fact, because the minimization carried out by boosting is performed over a set $\mathcal{S}$ of functions, the fact that boosting produces a minimum probability of error rule *does not guarantee that convergence has taken place*. It suffices that the predictor $g(\mathbf{x})$ produced by boosting is equal to $g^*(\mathbf{x})$ along the classification boundary , i.e. the set $\mathbf{x}$ such that $g^*(\mathbf{x}) = 0$. Away from the boundary, $g(\mathbf{x})$ can be substantially different from $g^*(\mathbf{x})$, since all that matters for the optimality of the decision rule is that both have the same sign.

It is important to emphasize that this is not a mere theoretical curiosity. In fact, boosting is designed to emphasize convergence in the neighborhood of the cost-insensitive boundary. This is the role of the weight resampling mechanism, which quickly discards points located away from it. While these points are easy to classify in the cost-insensitive case, therefore warranting this neglect, they become the points adjacent to the boundary when costs are considered (note, from (20) that cost-sensitivity changes the location of the optimal boundary). Because the convergence of the boosted predictor to (22) does not necessarily hold in their neighborhood, the simple modification of its threshold is not likely to result in the optimal cost-sensitive decision rule of (20).

The inability of boosting to produce good cost-sensitive rules has been experimentally confirmed by various authors, and a number of cost-sensitive extensions have been proposed in the literature [4, 15, 14, 17]. Some of these have, in fact, been applied to the problem of cascade design [10, 7]. The main difficulty is that these algorithms are mostly heuristic, relying on somewhat arbitrary heuristics to modify boosting's weight update rule. We have not been able to successfully design high detection-rate cascades with any of them.

4.2. A cost-sensitive boosting algorithm

A better alternative is to generalize Friedman's result. In this context, we have recently shown that the asymmetric loss

$$E\left[I(y=1)e^{-y.C_1 g(\mathbf{x})} + I(y=-1)e^{-y.C_2 g(\mathbf{x})}\right], \quad (23)$$

is minimized by the asymmetric logistic transform of $P_{Y|\mathbf{X}}(1|\mathbf{x})$,

$$g^*(\mathbf{x}) = \frac{1}{C_1+C_2}\log\frac{P_{Y|\mathbf{X}}(1|\mathbf{x})C_1}{P_{Y|\mathbf{X}}(-1|\mathbf{x})C_2}, \quad (24)$$

where $I(y=1)$ and $I(y=-1)$ are indicator functions [11]. Hence, (23) is an asymmetric boosting loss function that can be minimized in a manner similar to AdaBoost, by gradient descent on the space of convex combinations of weak learners. Defining two sets

$$\mathcal{I}_+ = \{i|y_i = 1\} \qquad \mathcal{I}_- = \{i|y_i = -1\}, \quad (25)$$

this leads to a weight update rule of the form

$$w_i^{(m+1)} = \begin{cases} w_i^{(m)} e^{-C_1\alpha_m\phi_m(\mathbf{x}_i)}, & i \in \mathcal{I}_+ \\ w_i^{(m)} e^{C_2\alpha_m\phi_m(\mathbf{x}_i)}, & i \in \mathcal{I}_-. \end{cases} \quad (26)$$

For a given step size α, the gradient direction is

$$\phi_m(\mathbf{x}) = \arg\min_{\phi}\Big[(e^{C_1\alpha} - e^{-C_1\alpha})\cdot b + e^{-C_1\alpha}T_+ \\ +(e^{C_2\alpha} - e^{-C_2\alpha})\cdot d + e^{-C_2\alpha}T_-\Big] \quad (27)$$

and the optimal step size is the solution of

$$2C_1\cdot b\cdot\cosh(C_1\alpha) + 2C_2\cdot d\cdot\cosh(C_2\alpha) = \\ C_1\cdot T_+\cdot e^{-C_1\alpha} + C_2\cdot T_-\cdot e^{-C_2\alpha} \quad (28)$$

with

$$T_+ = \sum_{i\in\mathcal{I}_+} w_i^{(m)} \qquad T_- = \sum_{i\in\mathcal{I}_-} w_i^{(m)} \quad (29)$$

$$b = \sum_{i\in\mathcal{I}_+} w_i^{(m)} I(y_i \neq \phi(x_i)) \quad (30)$$

$$d = \sum_{i\in\mathcal{I}_-} w_i^{(m)} I(y_i \neq \phi(x_i)). \quad (31)$$

Given a training set $(\mathbf{x}_1, y_1)....(\mathbf{x}_n, y_n)$ where $y \in \{+1,-1\}$ is the class label of example $\mathbf{x}$, and costs C_1, C_2, the complete asymmetric boosting algorithm is as follows:

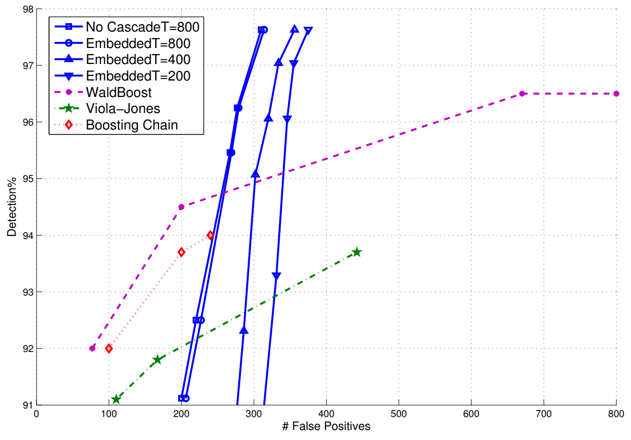

Figure 1. ROC of the various cascades discussed in the text, for face detection on the MIT-CMU test set.

- Initialize weights to uniform $w_i = \frac{1}{2|\mathcal{I}_+|}, \forall i \in \mathcal{I}_+, w_i = \frac{1}{2|\mathcal{I}_-|}\forall i \in \mathcal{I}_-$.
- For $t = 1, ..., T$ (Where T is the total number of weak learners.)
  1. for each j, train a weak learner/step-size pair $(\phi_j(\mathbf{x}), \alpha_j)$ using current weights w_i.
  2. Find the weak learner ϕ that minimizes the loss of (27) with α found by solving (28).
  3. update the weights according to (26).
- The final strong classifier implements the decision rule $s(\mathbf{x}) = \text{sign}[\sum_{m=1}^{T}\alpha_m\phi_m(\mathbf{x})]$.

5. Evaluation

We performed various experiments to evaluate the contributions of this paper. Unless otherwise noted, all experiments followed the experimental protocol of [16], using a face database of $10K$ positive and $350K$ negative examples, and weak learners combining decision stumps and Haar wavelet features (selected from a feature pool of $50,000$). Because asymmetric boosting maintains a high-detection rate throughout all iterations, it enables the design of cascades "a posteriori". This consists of simply adding exit points (i.e. rejecting examples classified as negatives) at intermediate steps of the detector. In particular, all single feature cascades discussed in what follows were produced by running asymmetric boosting to obtain a non-cascaded detector, and then introducing exit points after the evaluation of each weak learner.

Figure 1 presents a comparison between the performance of a single feature embedded cascade and the cascades produced by three state-of-the-art methods: VJ [16], Boosting

Method	VJ	Waldboost	Embedded200	Embedded400	Boosting Chain
#F	4297	400	200	400	700
Avg-Used	8	10.84	15.45	15.95	18.1

Table 1. comparison of length and average number of features used .

chain [19] and WaldBoost [13]. The evaluation was conducted on the CMU-MIT face test set, and the detection rate and number of total false positives are reported for a series of points on the ROC curves of the different cascades. We compared the performance of three single-feature embedded cascades obtained from the same boosting run, by considering 800, 400 and 200 features. Note that all three achieve much higher detection rates than those produced by the previous methods. In fact, the only method that achieves within 3% of their detection rate is WaldBoost, and only for substantially larger false positive rates. The previous methods produce smaller false positives in the low detection-rate regime, but we have not tried to optimize the performance in this area: all embedded cascades were obtained with ($C_1 = 5$,$C_2 = 1$), other cost configurations would have given greater preference to the low false-positive rate region.

It is also worth emphasizing that the design of the embedded cascade only required $350K$ random negative examples, extracted from images not containing faces. These were available in our lab from previous experiments with face detectors, but were not bootstrapped specifically for the design of this cascade. Although bootstrapping for more examples could have been used to improve performance, it is not a strict requirement for embedded cascades. This is in contrast to the millions of negative image patches used in the design of other methods. It is also interesting to note that all three embedded cascades achieve high detection-rates. In particular, note that the detection-rate does not seem to decrease from 200 to 800 features. This is evidence that asymmetric boosting maintains high-detection rates throughout the whole boosting run. It also challenges the assumption of independent errors which, for single feature embeddings, would imply an exponential decrease in detection rate. On the contrary, *there appears to be no penalty for the cascading operation*. To test this premise, we have also measured the ROC curve of the non-cascaded boosted detector. As can be seen from Figure 1, there is virtually no difference between its ROC and that of the tremendously faster single feature cascade.

Table 1 provides a comparison of the number of features (#F) and average number of features used (Avg-Used) by different methods. The non-cascaded detector of n features would use n features on average. The embedded cascade design reduces this number to about 15, a number comparable to that produced by the other methods. VJ and WaldBoost have slightly faster evaluation times, but significantly worse detection rates. Finally, the total training time was of 2 days for the 200 feature embedded cascade, and consisted uniquely of *CPU time* (no manual supervision), as opposed to the ”weeks” of trial and error reported by VJ.

References

[1] L. Bourdev and J. Brandt. Robust object detection via soft cascade. In *CVPR*, 2005.

[2] S. C. Brubaker, M. D. Mullin, and J. M. Rehg. Towards optimal training of cascaded detectors. In *ECCV*, 2006.

[3] R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern Classification*. John Wiley Sons Inc, New York, 2001.

[4] W. Fan, S. Stolfo, J. Zhang, and P. Chan. Adacost: Misclassification cost-sensitive boosting. In *ICML*, 1999.

[5] Y. Freund and R. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Computer and System Sciences*, 1997.

[6] J. Friedman, T. Hastie, and R. Tibshirani. Additive logistic regression: A statistical view of boosting. *Annals of Statistics*, 2000.

[7] X. Hou, C. Liu, and T. Tan. Online learning asymmetric boosted classifiers for object detection. In *CVPR*, 2007.

[8] S. Z. Li, Z. Zhang, H.-Y. Shum, and H. Zhang. Floatboost learning and statistical face detection. In *PAMI*, 2005.

[9] H. Luo. Optimization design of cascaded classifiers. In *CVPR*, 2005.

[10] Y. Ma and X. Ding. Robust real-time face detection based on cost-sensitive adaboost method . In *ICME*, 2003.

[11] H. Masnadi-Shirazi and N. Vasconcelos. Asymmetric boosting. In *ICML*, 2007.

[12] L. Mason, J. Baxter, P. Bartlett, and M. Frean. Boosting Algorithms as Gradient Descent. In *NIPS*, 2000.

[13] J. Sochman and J. Matas. Waldboost-learning for time constrained sequential detection. In *CVPR*, 2005.

[14] Y. Sun, A. K. C. Wong, and Y. Wang. Parameter inference of cost-sensitive boosting algorithms. In *MLDM*, 2005.

[15] K. M. Ting. A comparative study of cost-sensitive boosting algorithms. In *ICML*, 2000.

[16] P. Viola and M. Jones. Robust real-time object detection. In *Workshop on Statistical and Computational Theories of Vision*, 2001.

[17] P. Viola and M. Jones. Fast and robust classification using asymmetric adaboost and a detector cascade. In *NIPS*, 2002.

[18] B. Wu, H. AI, C. Huang, and S. Lao. Fast rotation invariant multi-view face detection based on real adaboost. In *FGR*, 2004.

[19] R. Xiao, L. Zhu, and H.-J. Zhang. Boosting chain learning for object detection. In *ICCV*, 2003.