Efficient Pedestrian Detection by Directly Optimizing the Partial Area under the ROC Curve

Sakrapee Paisitkriangkrai, Chunhua Shen*, Anton van den Hengel
The Australian Centre for Visual Technologies, The University of Adelaide, SA 5005, Australia

Abstract

Many typical applications of object detection operate within a prescribed false-positive range. In this situation the performance of a detector should be assessed on the basis of the area under the ROC curve over that range, rather than over the full curve, as the performance outside the range is irrelevant. This measure is labelled as the partial area under the ROC curve (pAUC). Effective cascade-based classification, for example, depends on training node classifiers that achieve the maximal detection rate at a moderate false positive rate, e.g., around 40% to 50%. We propose a novel ensemble learning method which achieves a maximal detection rate at a user-defined range of false positive rates by directly optimizing the partial AUC using structured learning. By optimizing for different ranges of false positive rates, the proposed method can be used to train either a single strong classifier or a node classifier forming part of a cascade classifier. Experimental results on both synthetic and real-world data sets demonstrate the effectiveness of our approach, and we show that it is possible to train state-of-the-art pedestrian detectors using the proposed structured ensemble learning method.

1. Introduction

Object detection is one of several fundamental topics in computer vision. The task of object detection is to identify predefined objects in a given image using knowledge gained through analysis of a set of labelled positive and negative exemplars. The Viola and Jones' face detection algorithm [24] forms the basis of many state-of-the-art real-time algorithms for object detection tasks.

The most commonly adopted evaluation method by which to compare the detection performance of different algorithms is the Receiver Operating Characteristic (ROC) curve. The curve illustrates the varying performance of a binary classifier system as its discrimination threshold is altered. In the face and human detection literature, researchers are often interested in the low false positive area of the ROC curve since this region characterizes the performance needed for most real-world vision applications. This is due to the fact that object detection is a highly asymmetric classification problem as there are only a small number of target objects among millions of background patches in the single test image. A false positive rate of 10^{-3} per scanning window would result in thousands of false positives in the single image, which is impractical for most applications. For many tasks, and particularly human detection, researchers also report the partial area under the ROC curve (pAUC), typically over the range 0.01 and 1.0 false positives per image [7, 21]. As the name implies, pAUC is calculated as the area under the ROC curve between two specified false positive rates (FPRs). It summarizes the practical performance of a detector and often is the primary performance measure of interest.

Although pAUC is *the metric of interest* that has been adopted to evaluate detection performance, many classifiers do not directly optimize this evaluation criterion, and as a result, often under-perform. In this paper, we present a principled approach for learning an ensemble classifier which directly optimizes the *partial* area under the ROC curve, where the range over which the area is calculated may be selected according to the desired application. Built upon the structured learning framework, we thus propose here a novel form of ensemble classifier which directly optimizes the partial AUC score, which we call pAUCEns. As with all other boosting algorithms, our approach learns a predictor by building an ensemble of weak classification rules in a greedy fashion. It also relies on a sample re-weighting mechanism to pass the information between each iteration. However, unlike traditional boosting, at each iteration, the proposed approach places a greater emphasis on samples which have the incorrect ordering[1] to achieve the *optimal partial AUC score*. The result is the ensemble learning method which yields the scoring function consistent with the correct relative ordering of positive and negative samples and optimizes the partial AUC score in a false positive

*This work was in part supported by ARC grant FT120100969. Correspondence should be addressed to C. Shen (email: chhshen@gmail.com).

[1] The positive sample has an incorrect ordering if it is ranked below the negative sample. In other words, we want all positive samples to be ranked above all negative samples.

rate range $[\alpha, \beta]$ where $0 \leq \alpha < \beta \leq 1$.

Main contributions (1) We propose a new ensemble learning approach which explicitly optimizes the partial area under the ROC curve (pAUC) between any two given false positive rates. The method is of particular interest in the wide variety of applications where performance is most important over a particular range within the ROC curve. The approach shares similarities with conventional boosting methods, but differs significantly in that the proposed method optimizes a multivariate performance measure using structured learning. Our design is simple and a conventional boosting-based visual detector can be transformed into a pAUCEns-based visual detector with very few modifications to the existing code. Our approach is efficient since it exploits both the efficient weak classifier training and the efficient cutting plane solver for optimizing the partial AUC score in the structural SVM setting. (2) We show that our approach is more intuitive and simpler to use than alternative algorithms, such as Asymmetric AdaBoost [23] and Cost-Sensitive AdaBoost [14], where one needs to cross-validate the asymmetric parameter from a fixed set of discrete points. Furthermore, it is unclear how one would set the asymmetric parameter in order to achieve a maximal pAUC score for a specified false positive range. To our knowledge, our approach is the first principled ensemble method that directly optimizes the partial AUC in an arbitrary false positive range $[\alpha, \beta]$. (3) Experimental results on several data sets, especially on challenging human detection data sets, demonstrate the effectiveness of the proposed approach. Our pedestrian detector performs better than or on par with the state-of-the-art, despite the fact that our detector only uses two standard low-level image features.

Related work Various ensemble classifiers have been proposed in the literature. Of these, AdaBoost is one the most well known as it has achieved tremendous success in computer vision and machine learning applications. In object detection, the cost of missing a true target is often higher than the cost of a false positive. Classifiers that are optimal under the symmetric cost, and thus treat false positives and negatives equally, cannot exploit this information [17, 23]. Several cost sensitive learning algorithms, where the classifier weights a positive class more heavily than a negative class, have thus been proposed.

Viola and Jones introduced the asymmetry property in Asymetric AdaBoost (AsymBoost) [23]. However, the authors reported that this asymmetry is immediately absorbed by the first weak classifier. Heuristics are then used to avoid this problem. In addition, one needs to carefully cross-validate this asymmetric parameter in order to achieve the desired result. Masnadi-Shirazi and Vasconcelos [14] proposed a cost-sensitive boosting algorithm based on the statistical interpretation of boosting. Their approach is to optimize the cost-sensitive loss by means of gradient descent. Most works along this line address the pAUC evaluation criterion *indirectly*. In addition, one needs to carefully cross-validate the asymmetric parameter in order to maximize the detection rate in a particular false positive range.

Several algorithms that directly optimize the pAUC score have been proposed in bioinformatics [9, 11]. Komori and Eguchi optimize the pAUC using boosting-based algorithms [11]. This algorithm is heuristic in nature. Narasimhan and Agarwal develop a structural SVM based method which directly optimizes the pAUC score [16]. They demonstrate that their approach, which uses a support vector method, significantly outperforms several existing algorithms, including pAUCBoost [11] and asymmetric SVM [28]. Building on Narasimhan and Agarwal's work, we propose the principled fully-corrective ensemble method which directly optimizes the pAUC evaluation criterion. The approach is flexible and can be applied to an arbitrary false positive range $[\alpha, \beta]$. To our knowledge, our approach is the first principled ensemble learning method that directly optimizes the partial AUC in a false positive range not bounded by zero. It is important to emphasize here the difference between our approach and that of [16]. [16] train a linear structural SVM while our approach learns the ensemble of classifiers. For pedestrian detection, HOG with the ensemble of classifiers reduces the average miss-rate over HOG+SVM by more than 30% [2].

Notation Bold lower-case letters, *e.g.*, $\boldsymbol{w}$, denote column vectors and bold upper-case letters, *e.g.*, $\mathbf{H}$, denote matrices. Let $\{\boldsymbol{x}_i^+\}_{i=1}^m$ be the set of positive training data and $\{\boldsymbol{x}_j^-\}_{j=1}^n$ be the set of negative training data. A set of all training samples can be written as $\mathbf{S} = (\mathbf{S}_+, \mathbf{S}_-)$ where $\mathbf{S}_+ = (\boldsymbol{x}_1^+, \cdots, \boldsymbol{x}_m^+)$ and $\mathbf{S}_- = (\boldsymbol{x}_1^-, \cdots, \boldsymbol{x}_n^-)$. We denote by $\mathcal{H}$ a set of all possible outputs of weak learners. Assuming that we have k possible weak learners, the output of weak learners for positive and negative data can be represented as $\mathbf{H} = (\mathbf{H}_+, \mathbf{H}_-)$ where $\mathbf{H}_+ \in \mathbb{R}^{k \times m}$ and $\mathbf{H}_- \in \mathbb{R}^{k \times n}$, respectively. Here h_{ti}^+ is the label predicted by the weak learner $\hbar_t(\cdot)$ on the positive training data $\boldsymbol{x}_i^+$. Each column $\boldsymbol{h}_{:l}$ of the matrix $\mathbf{H}$ represents the output of all weak learners when applied to the training instance $\boldsymbol{x}_l$. Each row $\boldsymbol{h}_{t:}$ of the matrix $\mathbf{H}$ represents the output predicted by the weak learner $\hbar_t(\cdot)$ on all the training data. The goal is to learn a set of binary weak learners and a scoring function, $f : \mathbb{R}^k \rightarrow \mathbb{R}$, that has good performance in terms of the pAUC between some specified false positive rates α and β where $0 \leq \alpha < \beta \leq 1$.

Structured learning approach for optimizing pAUC Before we propose our approach, we briefly review the concept of $\text{SVM}_{\text{pAUC}}[\alpha, \beta]$ [16], in which our ensemble learning approach is built upon. Unless otherwise stated, we follow the symbols used in [16]. The area under the empirical

ROC curve (AUC) can be defined as,

$$\text{AUC} = \frac{1}{mn}\sum_{i=1}^{m}\sum_{j=1}^{n}\mathbf{1}\big(f(\boldsymbol{x}_i^+) > f(\boldsymbol{x}_j^-)\big), \tag{1}$$

and the partial AUC in the false positive range $[\alpha, \beta]$ can be written as [5, 16],

$$\begin{aligned}
\text{pAUC} &= \frac{1}{mn(\beta-\alpha)}\sum_{i=1}^{m}\big(p_1(\alpha) + p_2(\alpha,\beta) + p_3(\beta)\big),\\
p_1(\alpha) &= (j_\alpha - n\alpha)\cdot\mathbf{1}\big(f(\boldsymbol{x}_i^+) > f(\boldsymbol{x}_{(j_\alpha)}^-)\big),\\
p_2(\alpha,\beta) &= \textstyle\sum_{j=j_\alpha+1}^{j_\beta}\mathbf{1}\big(f(\boldsymbol{x}_i^+) > f(\boldsymbol{x}_{(j)}^-)\big),\\
p_3(\beta) &= (n\beta - j_\beta)\cdot\mathbf{1}\big(f(\boldsymbol{x}_i^+) > f(\boldsymbol{x}_{(j_\beta+1)}^-)\big),
\end{aligned} \tag{2}$$

where $j_\alpha = \lceil n\alpha \rceil$, $j_\beta = \lfloor n\beta \rfloor$, $\boldsymbol{x}_{(j)}^-$ denotes the negative instance in $\mathbf{S}_-$ ranked in the j-th position amongst negative samples in descending order of scores. $p_1(\alpha)$, $p_2(\alpha,\beta)$ and $p_3(\beta)$ correspond to the sum of detection rates at FPR $= \big[\alpha, \frac{j_\alpha}{n}\big]$, FPR $= \big[\frac{j_\alpha}{n}, \frac{j_\beta}{n}\big]$, and FPR $= \big[\frac{j_\beta}{n}, \beta\big]$, respectively.

Given a training sample $\mathbf{S} = (\mathbf{S}_+, \mathbf{S}_-)$, our objective is to find a linear function $\boldsymbol{w}^\top\boldsymbol{x}$ that optimizes the pAUC in an FPR range of $[\alpha, \beta]$. We cast this pAUC optimization problem as a structural learning task. For any ordering of the training instances, the relative ordering of m positive instances and n negative instances is represented via a matrix $\boldsymbol{\pi} \in \{0,1\}^{m\times n}$ where,

$$\pi_{ij} = \begin{cases} 0 & \text{if } \boldsymbol{x}_i^+ \text{ is ranked above } \boldsymbol{x}_j^- \\ 1 & \text{otherwise.} \end{cases} \tag{3}$$

We define the correct relative ordering of $\boldsymbol{\pi}$ as $\boldsymbol{\pi}^*$ where $\pi_{ij}^* = 0, \forall i,j$. The pAUC loss in the false positive range $[\alpha,\beta]$ of $\boldsymbol{\pi}$ with respect to $\boldsymbol{\pi}^*$ can be written as,

$$\begin{aligned}
\Delta_{(\alpha,\beta)}(\boldsymbol{\pi}^*,\boldsymbol{\pi}) = \frac{1}{mn(\beta-\alpha)}\sum_{i=1}^{m}\big[(j_\alpha - n\alpha)\pi_{i,(j_\alpha)_{\boldsymbol{\pi}}} + \\
\textstyle\sum_{j=j_\alpha+1}^{j_\beta}\pi_{i,(j)_{\boldsymbol{\pi}}} + (n\beta - j_\beta)\pi_{i,(j_\beta+1)_{\boldsymbol{\pi}}}\big],
\end{aligned} \tag{4}$$

where $(j)_{\boldsymbol{\pi}}$ denotes the index of the negative instance consistent with the matrix $\boldsymbol{\pi}$. We define the joint feature map ϕ of the form

$$\phi(\mathbf{S},\boldsymbol{\pi}) = \frac{1}{mn(\beta-\alpha)}\textstyle\sum_{i,j}(1-\pi_{ij})(\boldsymbol{x}_i^+ - \boldsymbol{x}_j^-). \tag{5}$$

The choice of $\phi(\mathbf{S},\boldsymbol{\pi})$ over $\boldsymbol{\pi} \in \boldsymbol{\Pi}_{m,n}$ guarantees that the variable $\boldsymbol{w}$, which optimizes $\boldsymbol{w}^\top\phi(\mathbf{S},\boldsymbol{\pi})$, will also produce the scoring function $f(\boldsymbol{x}) = \boldsymbol{w}^\top\boldsymbol{x}$ that achieves the optimal partial AUC score. The above problem can be summarized as the following convex optimization problem [16]:

$$\begin{aligned}
\min_{\boldsymbol{w},\xi}\quad & \frac{1}{2}\|\boldsymbol{w}\|_2^2 + \nu\,\xi \\
\text{s.t. } & \boldsymbol{w}^\top\big(\phi(\mathbf{S},\boldsymbol{\pi}^*) - \phi(\mathbf{S},\boldsymbol{\pi})\big) \geq \Delta_{(\alpha,\beta)}(\boldsymbol{\pi}^*,\boldsymbol{\pi}) - \xi,
\end{aligned} \tag{6}$$

$\forall\boldsymbol{\pi} \in \boldsymbol{\Pi}_{m,n}$ and $\xi \geq 0$. Note that $\boldsymbol{\pi}^*$ denote the correct relative ordering and $\boldsymbol{\pi}$ denote any arbitrary orderings.

2. Our approach

In order to design an ensemble-like algorithm for the pAUC, we first introduce a projection function, $\hbar(\cdot)$, which projects an instance vector $\boldsymbol{x}$ to $\{-1,+1\}$. This projection function is also known as the weak learner in boosting. In contrast to the previously described structured learning, we learn the scoring function, which optimizes the area under the curve between two false positive rates of the form: $f(\boldsymbol{x}) = \sum_{t=1}^{k} w_t\hbar_t(\boldsymbol{x})$ where $\boldsymbol{w} \in \mathbb{R}^k$ is the linear coefficient vector and $\{\hbar_t(\cdot)\}_{t=1}^k$ denote a set of binary weak learners. Let us assume that we have already learned a set of all projection functions. By using the same pAUC loss, $\Delta_{(\alpha,\beta)}(\cdot,\cdot)$, as in (4), and the same feature mapping, $\phi(\cdot,\cdot)$, as in (5), the optimization problem we want to solve is:

$$\begin{aligned}
\min_{\boldsymbol{w},\xi}\quad & \frac{1}{2}\|\boldsymbol{w}\|_2^2 + \nu\,\xi \\
\text{s.t. } & \boldsymbol{w}^\top\big(\phi(\mathbf{H},\boldsymbol{\pi}^*) - \phi(\mathbf{H},\boldsymbol{\pi})\big) \geq \Delta_{(\alpha,\beta)}(\boldsymbol{\pi}^*,\boldsymbol{\pi}) - \xi,
\end{aligned} \tag{7}$$

$\forall\boldsymbol{\pi} \in \boldsymbol{\Pi}_{m,n}$ and $\xi \geq 0$. $\mathbf{H} = (\mathbf{H}_+, \mathbf{H}_-)$ is the projected output for positive and negative training samples. $\phi(\mathbf{H},\boldsymbol{\pi}) = [\phi(\boldsymbol{h}_{1:},\boldsymbol{\pi}),\cdots,\phi(\boldsymbol{h}_{k:},\boldsymbol{\pi})]$ where $\phi(\boldsymbol{h}_{t:},\boldsymbol{\pi}) : (\mathbb{R}^m\times\mathbb{R}^n)\times\boldsymbol{\Pi}_{m,n} \to \mathbb{R}$ and it is defined as,

$$\begin{aligned}
\phi(\boldsymbol{h}_{t:},\boldsymbol{\pi}) = \frac{1}{mn(\beta-\alpha)}\textstyle\sum_{i,j}(1-\pi_{ij}) \\
\big(\hbar_t(\boldsymbol{x}_i^+) - \hbar_t(\boldsymbol{x}_j^-)\big).
\end{aligned} \tag{8}$$

The only difference between (6) and (7) is that the original data is now projected to a new non-linear feature space. We will show how this can further improved the pAUC score in the experiment section. The dual problem of (7) can be written as (see supplementary),

$$\begin{aligned}
\max_{\boldsymbol{\lambda}}\quad & \textstyle\sum_{\boldsymbol{\pi}}\boldsymbol{\lambda}_{(\boldsymbol{\pi})}\Delta_{(\alpha,\beta)}(\boldsymbol{\pi}^*,\boldsymbol{\pi}) - \\
& \frac{1}{2}\textstyle\sum_{\boldsymbol{\pi},\hat{\boldsymbol{\pi}}}\boldsymbol{\lambda}_{(\boldsymbol{\pi})}\boldsymbol{\lambda}_{(\hat{\boldsymbol{\pi}})}\langle\phi_\Delta(\mathbf{H},\boldsymbol{\pi}),\phi_\Delta(\mathbf{H},\hat{\boldsymbol{\pi}})\rangle \\
\text{s.t. }\quad & 0 \leq \textstyle\sum_{\boldsymbol{\pi}}\boldsymbol{\lambda}(\boldsymbol{\pi}) \leq \nu.
\end{aligned} \tag{9}$$

where $\boldsymbol{\lambda}$ is the dual variable, $\boldsymbol{\lambda}_{(\boldsymbol{\pi})}$ denotes the dual variable associated with the inequality constraint for $\boldsymbol{\pi} \in \boldsymbol{\Pi}_{m,n}$ and $\phi_\Delta(\mathbf{H},\boldsymbol{\pi}) = \phi(\mathbf{H},\boldsymbol{\pi}^*) - \phi(\mathbf{H},\boldsymbol{\pi})$. To derive the Lagrange dual problem, the following KKT condition is used,

$$\boldsymbol{w} = \sum_{\boldsymbol{\pi}\in\boldsymbol{\Pi}_{m,n}}\boldsymbol{\lambda}_{(\boldsymbol{\pi})}\big(\phi(\mathbf{H},\boldsymbol{\pi}^*) - \phi(\mathbf{H},\boldsymbol{\pi})\big). \tag{10}$$

Finding best weak learners In this section, we show how one can explicitly learn the projection function, $\hbar(\cdot)$. We use the idea of column generation to derive an ensemble-like algorithm similar to LPBoost [4]. The condition for applying the column generation is that the duality gap between the primal and dual problem is zero (strong duality). By inspecting the KKT condition, at optimality, (10) must hold for all $t = 1,\cdots,k$. In other words, $w_t = \sum_{\boldsymbol{\pi}\in\boldsymbol{\Pi}_{m,n}}\boldsymbol{\lambda}_{(\boldsymbol{\pi})}\big(\phi(\boldsymbol{h}_{t:},\boldsymbol{\pi}^*) - \phi(\boldsymbol{h}_{t:},\boldsymbol{\pi})\big)$ must hold for all t.

For the weak learner in the current working set, the corresponding condition in (10) is satisfied by the current solution. For the weak learner that are not yet selected, they do not appear in the current restricted optimization problem and the corresponding $w_t = 0$. It is easy to see that if $\sum_{\boldsymbol{\pi}\in\boldsymbol{\Pi}_{m,n}} \boldsymbol{\lambda}_{(\boldsymbol{\pi})}\big(\phi(\boldsymbol{h}_{t:}, \boldsymbol{\pi}^*) - \phi(\boldsymbol{h}_{t:}, \boldsymbol{\pi})\big) = 0$ for any $\hbar_t(\cdot)$ that are not in the current working set, then the current solution is already the globally optimal one. Hence the subproblem for selecting the best weak learner is:

$$\hbar^*(\cdot) = \operatorname*{argmax}_{\hbar\in\mathcal{H}} \Big|\sum\nolimits_{\boldsymbol{\pi}} \boldsymbol{\lambda}_{(\boldsymbol{\pi})}\big(\phi(\boldsymbol{h}, \boldsymbol{\pi}^*) - \phi(\boldsymbol{h}, \boldsymbol{\pi})\big)\Big|. \quad (11)$$

In other words, we pick the weak learner with the value $|\sum_{\boldsymbol{\pi}} \boldsymbol{\lambda}_{(\boldsymbol{\pi})}\big(\phi(\boldsymbol{h}, \boldsymbol{\pi}^*) - \phi(\boldsymbol{h}, \boldsymbol{\pi})\big)|$ most deviated from zero. At iteration t, we pick the most optimal weak learner from $\mathcal{H}$. Substituting (8) into (11), the subproblem for generating the optimal weak learner at iteration t can be defined as,

$$\begin{aligned}
\hbar_t^*(\cdot) &= \operatorname*{argmax}_{\hbar\in\mathcal{H}} \Big|\sum_{\boldsymbol{\pi}} \boldsymbol{\lambda}_{(\boldsymbol{\pi})} \sum_{i,j} \pi_{ij}\big(\hbar(\boldsymbol{x}_i^+) - \hbar(\boldsymbol{x}_j^-)\big)\Big| \\
&= \operatorname*{argmax}_{\hbar\in\mathcal{H}} \Big|\sum_{i,j} \big(\textstyle\sum_{\boldsymbol{\pi}} \boldsymbol{\lambda}_{(\boldsymbol{\pi})}\pi_{ij}\big)\big(\hbar(\boldsymbol{x}_i^+) - \hbar(\boldsymbol{x}_j^-)\big)\Big| \\
&= \operatorname*{argmax}_{\hbar\in\mathcal{H}} \Big|\textstyle\sum_l u_l y_l \hbar(\boldsymbol{x}_l)\Big| \\
&= \operatorname*{argmax}_{\hbar\in\mathcal{H}} \textstyle\sum_l u_l y_l \hbar(\boldsymbol{x}_l) \qquad (12)
\end{aligned}$$

where i, j, l index the positive training samples ($i = 1, \cdots, m$), the negative training samples ($j = 1, \cdots, n$) and the entire training samples ($l = 1, 2, \cdots, m+n$), respectively. Here

$$u_l = \begin{cases} \sum_{\boldsymbol{\pi},j} \boldsymbol{\lambda}_{(\boldsymbol{\pi})}\pi_{lj} & \text{if } y_l = +1 \\ \sum_{\boldsymbol{\pi},i} \boldsymbol{\lambda}_{(\boldsymbol{\pi})}\pi_{il} & \text{if } y_l = -1. \end{cases} \quad (13)$$

For decision stumps, the last equation in (12) is always valid since the weak learner set $\mathcal{H}$ is negation-closed [12]. In other words, if $\hbar(\cdot) \in \mathcal{H}$, then $[-\hbar](\cdot) \in \mathcal{H}$, and vice versa. Here $[-\hbar](\cdot) = -\hbar(\cdot)$. For decision stumps, one can flip the inequality sign such that $\hbar(\cdot) \in \mathcal{H}$ and $[-\hbar](\cdot) \in \mathcal{H}$. In fact, any linear classifiers of the form $\mathbf{sign}(\sum_t a_t x_t + a_0)$ are negation-closed. Using (12) to choose the best weak learner is not heuristic as the solution to (11) decreases the duality gap the most for the current solution. See supplementary for more details.

Optimizing weak learners' coefficients We solve for the optimal $\boldsymbol{w}$ that minimizes our objective function (7). However, the optimization problem (7) has an exponential number of constraints, one for each matrix $\boldsymbol{\pi} \in \boldsymbol{\Pi}_{m,n}$. As in [10, 16], we use the cutting plane method to solve this problem. The basic idea of the cutting plane is that a small subset of the constraints are sufficient to find an ϵ-approximate solution to the original problem. The algorithm starts with an empty constraint set and it adds the most violated constraint set at each iteration. The QP problem is solved using linear SVM and the process continues until no constraint is violated by more than ϵ. Since, the quadratic program is of constant size and the cutting plane method converges in a constant number of iterations, the major bottleneck lies in the combinatorial optimization (over $\boldsymbol{\Pi}_{m,n}$) associated with finding the most violated constraint set at each iteration. Narasimhan and Agarwal show how this combinatorial problem can be solved efficiently in a polynomial time [16]. We briefly discuss their efficient algorithm in this section.

The combinatorial optimization problem associated with finding the most violated constraint can be written as,

$$\bar{\boldsymbol{\pi}} = \operatorname*{argmax}_{\boldsymbol{\pi}\in\boldsymbol{\Pi}_{m,n}} Q_{\boldsymbol{w}}(\boldsymbol{\pi}), \quad (14)$$

where

$$\begin{aligned}
Q_{\boldsymbol{w}}(\boldsymbol{\pi}) =& \Delta_{(\alpha,\beta)}(\boldsymbol{\pi}^*, \boldsymbol{\pi}) - \\
& \frac{1}{mn(\beta-\alpha)} \sum_{i,j} \pi_{ij} \boldsymbol{w}^\top (\boldsymbol{h}_{:i}^+ - \boldsymbol{h}_{:j}^-). \quad (15)
\end{aligned}$$

The trick to speed up (14) is to note that any ordering of the instances that is consistent with $\boldsymbol{\pi}$ yields the same objective value, $Q_{\boldsymbol{w}}(\boldsymbol{\pi})$ in (15). In addition, one can break down (14) into smaller maximization problems by restricting the search space from $\boldsymbol{\Pi}_{m,n}$ to the set $\boldsymbol{\Pi}_{m,n}^{\boldsymbol{w}}$ where

$$\boldsymbol{\Pi}_{m,n}^{\boldsymbol{w}} = \big\{\boldsymbol{\pi} \in \boldsymbol{\Pi}_{m,n} \big|\, \forall i, j_1 < j_2 : \pi_{i,(j_1)_{\boldsymbol{w}}} \geq \pi_{i,(j_2)_{\boldsymbol{w}}}\big\}.$$

Here $\boldsymbol{\Pi}_{m,n}^{\boldsymbol{w}}$ represents the set of all matrices $\boldsymbol{\pi}$ in which the ordering of the scores of two negative instances, $\boldsymbol{w}^\top \boldsymbol{h}_{:j_1}^-$ and $\boldsymbol{w}^\top \boldsymbol{h}_{:j_2}^-$, is consistent. The new optimization problem is now easier to solve as the set of negative instances over which the loss term in (15) is computed is the same for all orderings in the search space. This simplification allows one to reduce the computational complexity of (15) to $\mathcal{O}\big((m+n)\log(m+n)\big)$. Interested reader may refer to [16].

Discussion Our final ensemble classifier has a similar form as the AdaBoost-based object detector of [24]. Based on Algorithm 1, step ① and ② of our algorithm are exactly the same as [24]. Similar to AdaBoost, u_l in step ① plays the role of sample weights associated to each training sample. The major difference between AdaBoost and our approach is in step ③ and ④ where the weak learner's coefficient is computed and the sample weights are updated. In AdaBoost, the weak learner's coefficient is calculated as $w_t = \frac{1}{2}\log\frac{1-\epsilon_t}{\epsilon_t}$ where $\epsilon_t = \sum_l u_l I\big(y_l \neq \hbar_t(\boldsymbol{x}_l)\big)$ and I is the indicator function. The sample weights are updated with $u_l = \frac{u_l \exp(-w_t y_l \hbar_t(\boldsymbol{x}_l))}{\sum_l u_l \exp(-w_t y_l \hbar_t(\boldsymbol{x}_l))}$. We point this out here since a minimal modification is required in order to transform the existing implementation of AdaBoost to pAUCEns. Given the existing code of AdaBoost and the publicly available implementation of [16], our pAUCEns can be implemented in less than 10 lines of codes. A computational complexity analysis of our approach can be found in the supplementary.

In the next section, we train two different types of classi-

Algorithm 1 The training algorithm for pAUCEns.

Input:
1) A set of training examples $\{\boldsymbol{x}_l, y_l\}, l = 1, \cdots, m+n$;
2) The maximum number of weak learners, $t_{\max}$;
3) The regularization parameter, ν;
4) The learning objective based on the partial AUC, α and β;
Output: The scoring function†, $f(\boldsymbol{x}) = \sum_{t=1}^{t_{\max}} w_t h_t(\boldsymbol{x})$, that optimizes the pAUC score in the FPR range $[\alpha, \beta]$;
Initialize:
1) $t = 0$;
2) Initilaize sample weights: $u_l = \frac{0.5}{m}$ if $y_l = +1$, else $u_l = \frac{0.5}{n}$;
3) Extract low level features and store them in the cache memory for fast data access;
while $t < t_{\max}$ **do**
① Train a new weak learner using (12). The weak learner corresponds to the weak classifier with the minimal weighted error (maximal edge);
② Add the best weak learner into the current set;
③ Solve the structured SVM problem using the cutting plane algorithm [16];
④ Update sample weights, $\boldsymbol{u}$, using (13);
⑤ $t \leftarrow t + 1$;
end
† For a node in a cascade classifier, we introduce the threshold, b, and adjust b using the validation set such that $\mathbf{sign}\big(f(\boldsymbol{x}) - b\big)$ achieves the node learning objective;

fiers: the strong classifier [6] and the node classifier [24,27]. For the strong classifier, we set the value of α and β based on the evaluation criterion. For the node classifier, we set the value of α and β in each node to be 0.49 and 0.51, respectively.

3. Experiments

Synthetic data set We first illustrate the effectiveness of our approach on a synthetic data set similar to the one used in [23]. We compare pAUCEns against the baseline AdaBoost, Cost-Sensitive AdaBoost (CS-AdaBoost) [14] and Asymmetric AdaBoost (AsymBoost) [23]. We use vertical and horizontal decision stumps as the weak classifier. We evaluate the partial AUC score of different algorithms at $[0, 0.2]$ FPRs. For each algorithm, we train a strong classifier consisting of 10 and 25 weak classifiers. Additional details of the experimental set-up are provided in the supplementary. Fig. 1 illustrates the boundary decision[2] and the pAUC score. Our approach outperforms all other asymmetric classifiers. We observe that pAUCEns places more emphasis on positive samples than negative samples to ensure the highest detection rate at the left-most part of the ROC curve (FPR < 0.2). Even though we choose the asymmetric parameter, k, from a large range of values, both CS-AdaBoost and AsymBoost perform slightly worse than our approach. AdaBoost performs worst on this toy data set since it optimizes the overall classification accuracy. However as the number of weak classifiers increases (> 50 stumps), we observe all algorithms perform similarly on this simple toy data set. This observation could explain the success of AdaBoost in many object detection applications even though AdaBoost only minimizes the symmetric error rate.

[2]We set the threshold such that the false positive rate is 0.2.

	pAUC(0, 0.1)
Ours (pAUCEns)	**56.05%**
SVM_{pAUC} $[0, 0.1]^{\dagger}$ [16]	54.98%
pAUCBoost $[0, 0.1]^{\dagger\dagger}$ [11]	48.65%
Asym SVM $[0, 0.1]^{\dagger\dagger}$ [28]	44.51%
$\text{SVM}^{\dagger\dagger}_{\text{AUC}}$ [10]	39.72%

Table 1: The pAUC score on Protein-protein interaction data set. The **higher the pAUC score**, the **better the classifier**. † The result reported here is better than the one reported in [16]. We suspect that we tuned the regularization parameter in the finer range. Results marked by †† were reported in [16]. The best classifier is shown in boldface.

In the next experiment, we train a strong classifier of 10 weak classifiers and compare the performance of different classifiers at FPR of 0.5. We choose this value since it is the node learning goal often used in training a cascade classifier. Also we only learn 10 weak classifiers since the first node of the cascade often contains a small number of weak classifiers for real-time performance. For pAUCEns, we set the value of $[\alpha, \beta]$ to be $[0.49, 0.51]$. In Fig. 2, we display the decision boundary of each algorithm, and display both their pAUC score (in the FPR range $[0.49, 0.51]$) and detection rate at 50% false positive rate. We observe that our approach and AsymBoost have the highest detection rate at 50% false positive rate. However, our approach outperforms AsymBoost on a pAUC score. We observe that our approach places more emphasis on positive samples near the corners (at $\pi/4$, $3\pi/4$, $-\pi/4$ and $-3\pi/4$ angles) than other algorithms.

Protein-protein interaction prediction In this experiment, we compare our approach with existing algorithms which optimize pAUC in bioinformatics. The problem we consider here is a protein-protein interaction prediction [19], in which the task is to predict whether a pair of proteins interact or not. We used the data set labelled 'Physical Interaction Task in Detailed feature type', which is publicly available on the internet[3]. The data set contains 2865 protein pairs known to be interacting (positive) and a random set of $237,384$ protein pairs labelled as non-interacting (negative). We use a subset of 85 features as in [16]. We randomly split the data into two groups: 10% for training/validation and 90% for evaluation. We choose the best regularization parameter form $\{5, 2, 1, 1/2, 1/5\}$ by 5-fold cross validation. We repeat our experiments 10 times using the same regularization parameter. We train a linear classifier as our weak learner using LIBLINEAR [8]. We set the maximum number of boosting iterations to 100 and report the pAUC score of our approach in Table 1. Baselines include SVM_{pAUC}, SVM_{AUC}, pAUCBoost and Asymmet-

[3]http://www.cs.cmu.edu/~qyj/papers_sulp/proteins05_pages/feature-download.html

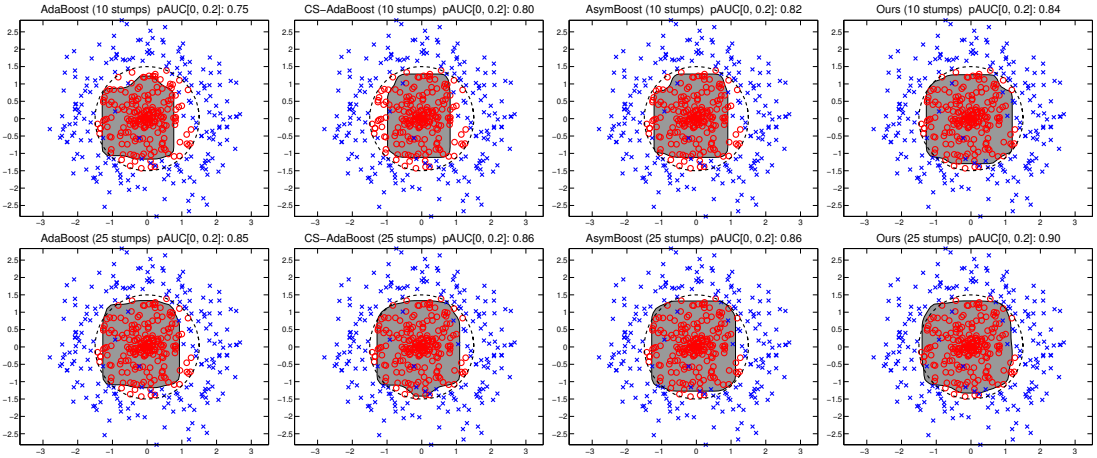

Figure 1: Decision boundaries on the toy data set where each strong classifier consists of **Top row:** 10 weak classifiers and **Bottom row:** 25 weak classifiers. Positive and negative data are represented by $\circ$ and $\times$, respectively. The partial AUC score in the FPR range $[0, 0.2]$ is also displayed. Our approach achieves the best pAUC score of 0.84 and 0.9 at 10 and 25 weak classifiers, respectively. At 25 weak classifiers, we observe that both traditional and asymmetric classifiers start to perform similarly.

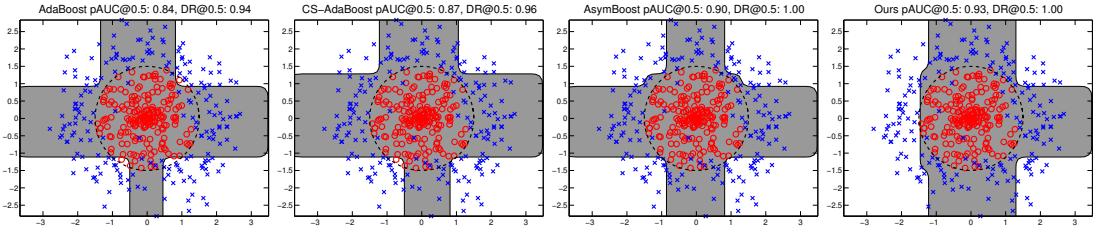

Figure 2: Decision boundaries on a toy data set with 10 weak classifiers at FPR of 0.5 The partial AUC score and detection rate at 50% false positive rate are also shown. Our approach performs best on both evaluation criteria. Our approach preserves a larger decision boundary near positive samples at $\pi/4$, $3\pi/4$, $-\pi/4$ and $-3\pi/4$ angles.

ric SVM. Our approach outperforms all existing algorithms which optimize either AUC or pAUC . We attribute our improvement over SVM_{pAUC} $[0, 0.1]$ [16], as a result of introducing a non-linearity into the original problem. This phenomenon has also been observed in face detection as reported in [27].

Comparison to other asymmetric boosting Here we compare pAUCEns against several boosting algorithms previously proposed for the problem of object detection, namely, AdaBoost with Fisher LDA post-processing [27], AsymBoost [23] and CS-AdaBoost [14]. The results of AdaBoost are also presented as the baseline. For each algorithm, we train a strong classifier consisting of 100 weak classifiers. We then calculate the pAUC score by varying the threshold value in the FPR range $[0, 0.1]$. For each algorithm, the experiment is repeated 20 times and the average pAUC score is reported. For AsymBoost, we choose k from $\{2^{-0.5}, 2^{-0.4}, \cdots, 2^{0.5}\}$ by cross-validation. For CS-AdaBoost, we choose k from $\{0.5, 0.75, \cdots, 3\}$ by cross-validation. We evaluate the performance of all algorithms on 3 vision data sets: USPS digits, scenes and face data sets. See supplementary for more details on feature extraction. We report the experimental results in Table 2. From the table, pAUCEns demonstrates the best performance on all three vision data sets.

Pedestrian detection - Strong classifier We evaluate our approach on the pedestrian detection task. We train our approach on the INRIA pedestrian data set. For the positive training data, we use all 2416 INRIA cropped pedestrian images. To generate the negative training data, we first train the cascade classifier with 20 nodes using Viola and Jones' approach. We then combine 2416 random negative windows generated in the first node with another 4832 negative windows generated in the subsequent nodes. The resulting 7248 negative windows are used for training the strong classifier. We generate a large pool of features by combining the histogram of oriented gradient (HOG) features [3]

	# iters	USPS	SCENE	FACE
Ours (pAUCEns)	10	**0.77 (0.02)**	**0.65 (0.02)**	**0.53 (0.02)**
	20	**0.85 (0.01)**	**0.75 (0.01)**	**0.67 (0.01)**
	100	**0.93 (0.01)**	**0.85 (0.01)**	**0.82 (0.01)**
AdaBoost [24]	10	0.75 (0.03)	0.63 (0.03)	0.52 (0.03)
	20	0.82 (0.03)	0.72 (0.03)	0.63 (0.03)
	100	0.90 (0.02)	0.83 (0.02)	0.78 (0.02)
Ada + LDA [27]	10	0.74 (0.02)	0.63 (0.02)	**0.53 (0.02)**
	20	0.79 (0.02)	0.73 (0.02)	0.63 (0.02)
	100	0.86 (0.01)	0.84 (0.01)	0.76 (0.01)
AsymBoost [23]	10	0.76 (0.02)	0.58 (0.02)	0.51 (0.02)
	20	0.83 (0.01)	0.69 (0.01)	0.65 (0.01)
	100	0.85 (0.03)	0.81 (0.03)	0.82 (0.03)
CS-AdaBoost [14]	10	0.75 (0.04)	0.64 (0.04)	0.52 (0.04)
	20	0.82 (0.03)	0.73 (0.03)	0.63 (0.03)
	100	0.91 (0.02)	0.84 (0.02)	0.78 (0.02)

Table 2: Average pAUC scores and their standard deviations on vision data sets at various boosting iterations. All experiments are repeated 20 times. The best average performance is shown in boldface.

Train \ Test (FPPI)	$[0, 0.005]$	$[0, 0.05]$	$[0, 0.5]$	$[0, 1]$
$\beta = 0.05$	$\mathbf{65.0}\%$	39.3%	20.6%	18.1%
$\beta = 0.1$	75.7%	$\mathbf{32.5}\%$	18.0%	15.8%
$\beta = 0.5$	73.4%	33.8%	$\mathbf{17.9}\%$	$\mathbf{15.3}\%$

Table 3: The pAUC score in the FPR range $[0, \beta]$ on the training set. Our objective here is to optimize the area under the curve between $[0, 0.005]$, $[0, 0.05]$, $[0, 0.5]$ and $[0, 1]$ FPPI on the INRIA test set. Since we plot FPPI versus miss rate, a **smaller pAUC score** means a **better detector**. The best detector at each FPPI range is shown in boldface. Clearly, a large value of β is best for a large FPPI range.

and covariance (COV) features[4] [22]. Additional details are provided in the supplementary. We use weighted linear discriminant analysis (WLDA) as weak classifiers [17]. We train 500 weak classifiers and set 5 multi-exits [18]. To be more specific, we set the threshold at 10, 20, 50, 100 and 200 weak classifiers. These exits reduce the evaluation time during testing significantly. The regularization parameter ν is cross-validated from $\{0.1, 0.5, 1, 2, 10\}$. Since we have not carefully cross-validated a finer range of ν, tuning this parameter could yield a further improvement. The training time of our approach is under two hours on a parallelized quad core Xeon machine.

During evaluation, each test image is scanned with 4×4 pixels step stride and the scale ratio of input image pyramid is 1.05. The overlapped detection windows are merged using the greedy non-maximum suppression strategy as introduced in [6]. We use the continuous AUC evaluation software of Sermanet *et al*. [20] and report the pAUC score between $[0, 0.005]$ FPPI (1 false positive), $[0, 0.05]$ FPPI (15 false positives), $[0, 0.5]$ FPPI (144 false positives) and $[0, 1]$ FPPI (288 false positives) in Table 3. From the table, we observe that setting the value of β to be minimal ($\beta = 0.05$) yeilds the best pAUC score at $[0, 0.005]$ FPPI . As we increase the FPPI range, the higher value of β tends to perform better. This table clearly illustrates the advantage of our approach.

Fig. 3 compares the performance of our approach with other state-of-the-art algorithms on the INRIA pedestrian data set. We use the evaluation software of Dollár *et al*. [7], which computes the AUC from 9 discrete points sampled between $[0.01, 1.0]$ FPPI . Our approach performs second best on this data set. It performs comparable to VeryFast [1] which trains multiple detectors at multiple scale. Upon a closer observation, our pAUCEns performs slightly better than VeryFast when the number of FPPI is less then 0.1 and VeryFast performs slightly better when the number of FPPI is greater 0.1. We evaluate our strong classifier on TUD-Brussels and ETH pedestrian data sets but we observe that the detection results contain a large number of false positives. Instead of bootstrapping with more negative samples as in [6,25], we train a cascade classifier in the next section.

[4]Covariance features capture the relationship between different image statistics and have been shown to perform well in our previous experiments. However, other discriminative features can also be used here instead, *e.g*., Haar-like features, Local Binary Pattern (LBP) [15], Sketch Tokens [13] and self-similarity of low-level features (CSS) [25].

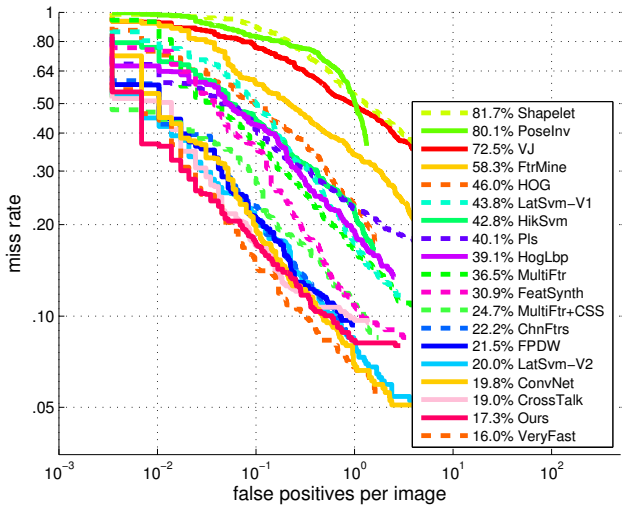

Figure 3: ROC curves of our approach and several state-of-the-art detectors on the INRIA test image. We train a strong classifier using HOG and COV features.

Pedestrian detection - Cascade classifier In this section, we train a cascade classifier using our pAUCEns. We train our detector on INRIA training set and evaluate the detector on INRIA, TUD-Brussels and ETH test sets. On both TUD-Brussels and ETH data sets, we upsample the original image to 1600×1200 pixels before applying our pedestrian detector. We train the human detector with a combination of HOG and COV features as previously described. To achieve the node learning goal of the cascade (each node achieves an extremely high detection rate ($> 99\%$) and a moderate false positive rage ($\approx 50\%$)), we optimize the pAUC in the FPR range $[0.49, 0.51]$. We train a multi-exit cascade [18] with 19 exit. In this experiment, we use the software of [20] to compute the continuous AUC score in the FPPI range $[0, 0.1]$. We sort different algorithms based on the pAUC score in the FPPI range $[0, 0.1]$ and report the results in Fig. 4. We compare our proposed approach with the baseline HOGCOV classifier (using AdaBoost). We observe that our approach reduces the average miss-rate over HOGCOV by 7% on INRIA test set. From Fig. 4, our approach achieves similar performance to the state-of-the-art detector. We then break-down experimental results of different measures using the partial AUC score (FPPI range $[0, 0.1]$) in Table 4. On average, our approach performs best on the *large* evaluation setting where pedestrians are at least 100 pixels tall. On other settings, our approach yields competitive results to the state-of-the-art detector in that category. In summary, our approach performs better than or on par with the state-of-the-art despite its simplicity (in comparison to LatSvm — a part-based approach which models unknown parts as latent variables). In addition, the current detector is only trained with two discriminative visual features (HOG and COV). Applying additional discriminative features, *e.g*., LBP [26] or motion features [25], could further improve the overall detection performance.

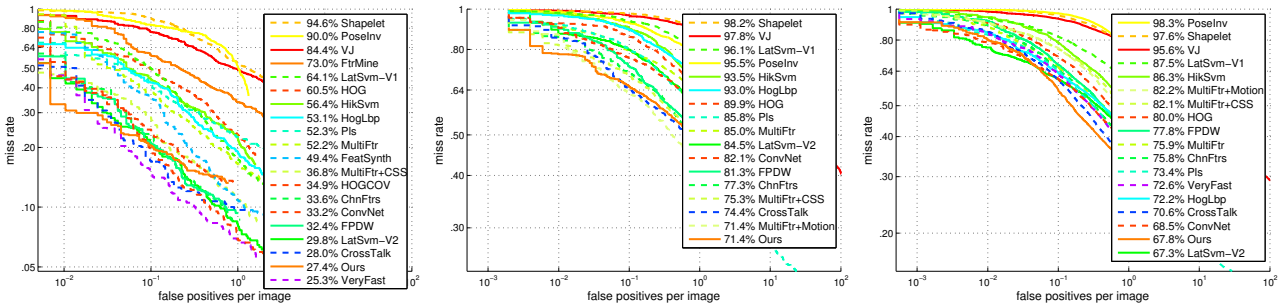

Figure 4: From left to right: performance on INRIA, TUD-Brussels and ETH test set. Algorithms are sorted using the **partial AUC score** in the FPPI range $[0, 0.1]$. Our pAUCEns consistently performs comparable to the state-of-the-art.

	Ours	**ChnFtrs**	**ConvNet**	**CrossTalk**	**FPDW**	**FeatSynth**	**FtrMine**	**HOG**	**HikSvm**	**HogLbp**	**LatSvm-V1**	**LatSvm-V2**	**MultiFtr**	**Pls**	**PoseInv**	**Shapelet**	**VJ**	**VeryFast**
Reasonable (min. 50 pixels tall & no/partial occlusion) - Partial AUC(0,0.1)%																		
INRIA-Fixed	27.4	31.6	31.9	26.9	30.9	49.3	71.5	59.4	53.9	50.5	62.6	29.1	51.3	50.4	89.4	93.0	83.2	**23.9**
TudBrussels	**65.8**	72.2	77.8	68.8	77.0	-	-	87.9	92.3	91.3	95.5	80.8	81.6	82.6	94.5	97.8	97.4	-
ETH	62.8	72.4	62.8	67.0	74.5	-	-	78.1	85.5	67.4	86.1	**61.4**	73.1	69.5	98.1	97.3	95.4	68.7
Large (min. 100 pixels tall) - Partial AUC(0,0.1)%																		
INRIA-Fixed	26.0	29.6	27.6	25.0	28.7	48.6	70.9	59.0	53.1	49.5	61.6	25.9	50.0	49.3	89.4	92.9	82.9	**21.7**
TudBrussels	**47.3**	50.0	49.5	52.8	53.8	-	-	88.4	85.1	73.9	88.9	67.9	71.4	66.8	94.2	92.8	96.3	-
ETH	**42.9**	57.6	48.1	48.6	62.7	-	-	56.9	66.0	54.2	74.7	45.5	59.4	50.8	96.3	93.4	92.0	48.6
Near (min. 80 pixels tall) - Partial AUC(0,0.1)%																		
INRIA-Fixed	25.9	30.1	30.8	25.4	29.5	48.3	70.9	58.5	53.0	49.4	62.0	27.6	50.3	49.5	89.2	92.9	82.8	**22.7**
TudBrussels	**49.0**	59.1	60.1	58.7	62.1	-	-	87.1	88.0	79.5	91.7	70.6	74.5	75.1	93.9	95.1	96.3	-
ETH	51.3	64.8	51.8	55.9	66.5	-	-	69.2	74.2	56.7	77.5	**51.0**	63.2	60.9	97.8	95.2	93.6	55.4
Medium (min. 30 pixels tall and max. 80 pixels tall) - Partial AUC(0,0.1)%																		
INRIA-Fixed	100.0	100.0	54.9	96.5	100.0	100.0	100.0	100.0	100.0	94.6	88.8	96.5	94.3	100.0	96.5	96.5	89.6	**51.3**
TudBrussels	75.3	78.0	84.6	**74.0**	81.3	-	-	86.4	93.3	97.0	96.1	85.7	83.7	86.3	94.0	98.6	97.3	-
ETH	67.2	**64.8**	76.0	65.0	67.1	-	-	69.8	80.9	78.7	88.3	76.7	68.6	67.0	96.3	89.8	89.0	73.2

Table 4: Performance comparison of various detectors on several pedestrian test sets. The best detector in each category from each data set is highlighted in bold. The AUC score is taken over the FPPI range $[0, 0.1]$. A **smaller pAUC score** means a **better detector**. The AUC score over the FPPI range $[0, 1]$ can be found in the supplementary.

Conclusion We have proposed a new ensemble learning method for object detection. The proposed approach is based on optimizing the partial AUC score in the FPR range $[\alpha, \beta]$. Extensive experiments demonstrate the effectiveness of the proposed approach in visual detection tasks. We plan to explore the possibility of applying the proposed approach to the multiple scales detector of [1] in order to improve the detection results of very low resolution pedestrian images.

References

[1] R. Benenson, M. Mathias, R. Timofte, and L. V. Gool. Pedestrian detection at 100 frames per second. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, 2012. 7, 8

[2] R. Benenson, M. Mathias, T. Tuytelaars, and L. V. Gool. Seeking the strongest rigid detector. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, 2013. 2

[3] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, volume 1, 2005. 6

[4] A. Demiriz, K. Bennett, and J. Shawe-Taylor. Linear programming boosting via column generation. *Mach. Learn.*, 46(1–3):225–254, 2002. 3

[5] L. E. Dodd and M. S. Pepe. Partial auc estimation and regression. *Biometrics*, 59(3):614–623, 2003. 3

[6] P. Dollár, Z. Tu, P. Perona, and S. Belongie. Integral channel features. In *Proc. of British Mach. Vis. Conf.*, 2009. 4, 7

[7] P. Dollár, C. Wojek, B. Schiele, and P. Perona. Pedestrian detection: An evaluation of the state of the art. *IEEE Trans. Pattern Anal. Mach. Intell.*, 34(4):743–761, 2012. 1, 7

[8] R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, and C.-J. Lin. LIBLINEAR: A library for large linear classification. *J. Mach. Learn. Res.*, 9:1871–1874, 2008. 5

[9] M.-J. Hsu and H.-M. Hsueh. The linear combinations of biomarkers which maximize the partial area under the roc curves. *Comp. Stats.*, 28(2):1–20, 2012. 2

[10] T. Joachims, T. Finley, and C.-N. J. Yu. Cutting-plane training of structural svms. *Mach. Learn.*, 77(1):27–59, 2009. 4, 5

[11] O. Komori and S. Eguchi. A boosting method for maximizing the partial area under the roc curve. *BMC Bioinformatics*, 11(1):314, 2010. 2, 5

[12] O. Komori and S. Eguchi. Boosting learning algorithm for pattern recognition and beyond. *IEICE Trans. Infor. and Syst.*, 94(10):1863–1869, 2011. 4

[13] J. J. Lim, C. L. Zitnick, and P. Dollar. Sketch Tokens: A learned mid-level representation for contour and object detection. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, 2013. 7

[14] H. Masnadi-Shirazi and N. Vasconcelos. Cost-sensitive boosting. *IEEE Trans. Pattern Anal. Mach. Intell.*, 33(2):294–309, 2011. 2, 5, 6

[15] Y. Mu, S. Yan, Y. Liu, T. Huang, and B. Zhou. Discriminative local binary patterns for human detection in personal album. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, Anchorage, AK, US, 2008. 7

[16] H. Narasimhan and S. Agarwal. A structural svm based approach for optimizing partial auc. In *Proc. Int. Conf. Mach. Learn.*, 2013. 2, 3, 4, 5, 6

[17] S. Paisitkriangkrai, C. Shen, and J. Zhang. Fast pedestrian detection using a cascade of boosted covariance features. *IEEE Trans. Circuits & Syst. for Vid. Tech.*, 18(8), 2008. 2, 7

[18] M.-T. Pham, V.-D. D. Hoang, and T.-J. Cham. Detection with multi-exit asymmetric boosting. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, 2008. 7

[19] Y. Qi, Z. Bar-Joseph, and J. Klein-Seetharaman. Evaluation of different biological data and computational classification methods for use in protein interaction prediction. *Proteins: Struct., Func., and Bioinfor.*, 63(3):490–500, 2006. 5

[20] P. Sermanet, K. Kavukcuoglu, S. Chintala, and Y. LeCun. Pedestrian detection with unsupervised multi-stage feature learning. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, 2013. 7

[21] C. Shen, P. Wang, S. Paisitkriangkrai, and A. van den Hengel. Training effective node classifiers for cascade classification. *Int. J. Comp. Vis.*, 103(3), 2013. 1

[22] O. Tuzel, F. Porikli, and P. Meer. Pedestrian detection via classification on Riemannian manifolds. *IEEE Trans. Pattern Anal. Mach. Intell.*, 30(10):1713–1727, 2008. 6

[23] P. Viola and M. Jones. Fast and robust classification using asymmetric AdaBoost and a detector cascade. In *Proc. Adv. Neural Inf. Process. Syst.*, pages 1311–1318. MIT Press, 2002. 2, 5, 6

[24] P. Viola and M. J. Jones. Robust real-time face detection. *Int. J. Comp. Vis.*, 57(2):137–154, 2004. 1, 4, 6

[25] S. Walk, N. Majer, K. Schindler, and B. Schiele. New features and insights for pedestrian detection. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, San Francisco, US, 2010. 7

[26] X. Wang, T. X. Han, and S. Yan. An HOG-LBP human detector with partial occlusion handling. In *Proc. IEEE Int. Conf. Comp. Vis.*, 2009. 7

[27] J. Wu, S. C. Brubaker, M. D. Mullin, and J. M. Rehg. Fast asymmetric learning for cascade face detection. *IEEE Trans. Pattern Anal. Mach. Intell.*, 30(3):369–382, 2008. 4, 6

[28] S.-H. Wu, K.-P. Lin, C.-M. Chen, and M.-S. Chen. Asymmetric support vector machines: low false-positive learning under the user tolerance. In *Proc. of Intl. Conf. on Knowledge Discovery and Data Mining*, 2008. 2, 5