Supplementary material: Efficient pedestrian detection by directly optimizing the partial area under the ROC curve

Sakrapee Paisitkriangkrai, Chunhua Shen, Anton van den Hengel
The University of Adelaide, Australia

October, 2013

In this document we provide a Lagrange dual derivation, our justification for finding best weak learners, a complete analysis of the computational complexity of our approach, the experimental setup and additional experimental results presented in the main paper.

1 Lagrange dual derivation

The pAUC optimization problem presented can be summarized into the following convex optimization problem [8]:

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

$\forall \boldsymbol{\pi} \in \boldsymbol{\Pi}_{m,n}$ and $\xi \geq 0$. Here we derive the Lagrange dual of the above optimization problem. The Lagrangian of (1) can be written as,

$$L = \frac{1}{2}\|\boldsymbol{w}\|_2^2 + \nu\xi - z\xi - \sum_{\boldsymbol{\pi}\in\boldsymbol{\Pi}_{m,n}} \boldsymbol{\lambda}_{(\boldsymbol{\pi})}\Big(\boldsymbol{w}^\top\big(\phi(\mathbf{H},\boldsymbol{\pi}^*) - \phi(\mathbf{H},\boldsymbol{\pi})\big) - \Delta_{(\alpha,\beta)}(\boldsymbol{\pi}^*,\boldsymbol{\pi}) + \xi\Big), \tag{2}$$

where $\boldsymbol{\lambda}$ and z are Lagrange multiplier and $\boldsymbol{\lambda} \geq \mathbf{0}$ and $z \geq 0$. Here $\boldsymbol{\lambda}_{(\boldsymbol{\pi})}$ is denoted as the Lagrange multiplier associated with the inequality constraint for $\boldsymbol{\pi} \in \boldsymbol{\Pi}_{m,n}$. At optimum, the first derivative of the Lagrangian with respect to the primal variables, $\frac{\partial L}{\partial \xi}$ and $\frac{\partial L}{\partial \boldsymbol{w}}$, must be zero.

$$\begin{aligned}\frac{\partial L}{\partial \xi} = 0 &\rightarrow \nu - \textstyle\sum_{\boldsymbol{\pi}} \boldsymbol{\lambda}_{(\boldsymbol{\pi})} - z = 0 \\ &\rightarrow 0 \leq \textstyle\sum_{\boldsymbol{\pi}} \boldsymbol{\lambda}_{(\boldsymbol{\pi})} \leq \nu\end{aligned} \tag{3}$$

and

$$\frac{\partial L}{\partial \boldsymbol{w}} = 0 \rightarrow \boldsymbol{w} - \textstyle\sum_{\boldsymbol{\pi}} \boldsymbol{\lambda}_{(\boldsymbol{\pi})}\phi_\Delta(\mathbf{H},\boldsymbol{\pi}) = 0, \tag{4}$$

where $\phi_\Delta(\mathbf{H},\boldsymbol{\pi}) = \phi(\mathbf{H},\boldsymbol{\pi}^*) - \phi(\mathbf{H},\boldsymbol{\pi})$. By substituting (3) and (4) into (2), the dual problem of (1) can be written as,

$$\begin{aligned}\max_{\boldsymbol{\lambda}} \quad & \textstyle\sum_{\boldsymbol{\pi}} \boldsymbol{\lambda}_{(\boldsymbol{\pi})}\Delta_{(\alpha,\beta)}(\boldsymbol{\pi}^*,\boldsymbol{\pi}) - \frac{1}{2}\sum_{\boldsymbol{\pi},\hat{\boldsymbol{\pi}}} \boldsymbol{\lambda}_{(\boldsymbol{\pi})}\boldsymbol{\lambda}_{(\hat{\boldsymbol{\pi}})}\langle\phi_\Delta(\mathbf{H},\boldsymbol{\pi}),\phi_\Delta(\mathbf{H},\hat{\boldsymbol{\pi}})\rangle \\ \text{s.t.} \quad & 0 \leq \textstyle\sum_{\boldsymbol{\pi}} \boldsymbol{\lambda}(\boldsymbol{\pi}) \leq \nu.\end{aligned} \tag{5}$$

From (4), the relationship between the primal variables and the dual variables (KKT condition) at optimality is $\boldsymbol{w} = \sum_{\boldsymbol{\pi}\in\boldsymbol{\Pi}_{m,n}} \boldsymbol{\lambda}_{(\boldsymbol{\pi})}\big(\phi(\mathbf{H},\boldsymbol{\pi}^*) - \phi(\mathbf{H},\boldsymbol{\pi})\big)$.

2 Finding best weak learners

In this paper, we select a subset of discriminative weak learners from the set of infinitely large weak learners. For decision stumps, the size of $\mathcal{H}$ is equal to the number of features times the number of training samples. For decision trees, the size of $\mathcal{H}$ grows exponentially with the tree depth. To identify an optimal set of weak learners, at each iteration, we choose the weak learner that achieves the maximal decrease in the duality gap of the current solution *i.e.*, this weak learner is analogous to the weak learner that moves in the direction of steepest descent in AdaBoost. From the main paper, the subproblem for selecting the best weak learner is:

$$\hbar^*(\cdot) = \underset{\hbar \in \mathcal{H}}{\operatorname{argmax}} \left| \sum_{\boldsymbol{\pi}} \boldsymbol{\lambda}_{(\boldsymbol{\pi})} \big(\phi(\boldsymbol{h}, \boldsymbol{\pi}^*) - \phi(\boldsymbol{h}, \boldsymbol{\pi}) \big) \right|. \tag{6}$$

Here we show that using (6) to choose a weak learner is not heuristic in terms of solving the primal problem.

Claim 1 (Finding best weak learners). *At iteration $t+1$, the weak learner selected using* (6) *decreases the duality gap the most for the current solution obtained at iteration t, in terms of solving the SVM primal problem or the SVM dual problem.*

To prove the above claim, let $\ddot{\mathcal{H}}$ denote the current set of selected weak classifiers and $\tilde{\mathcal{H}}$ denotes the rest of weak classifiers that have not been selected. The dual objective for both selected and unselected set of weak classifiers is

$$\begin{aligned} \max_{\boldsymbol{\lambda}} \quad & \textstyle\sum_{\boldsymbol{\pi}} \boldsymbol{\lambda}_{(\boldsymbol{\pi})} \Delta_{(\alpha,\beta)}(\boldsymbol{\pi}^*, \boldsymbol{\pi}) - \frac{1}{2} \sum_{\boldsymbol{\pi}, \hat{\boldsymbol{\pi}}} \boldsymbol{\mu}_{\ddot{\mathcal{H}}}(\boldsymbol{\pi}, \hat{\boldsymbol{\pi}}) - \frac{1}{2} \sum_{\boldsymbol{\pi}, \hat{\boldsymbol{\pi}}} \boldsymbol{\mu}_{\tilde{\mathcal{H}}}(\boldsymbol{\pi}, \hat{\boldsymbol{\pi}}) \\ \text{s.t.} \quad & 0 \leq \textstyle\sum_{\boldsymbol{\pi}} \boldsymbol{\lambda}(\boldsymbol{\pi}) \leq \nu. \end{aligned} \tag{7}$$

where $\boldsymbol{\mu}_{\mathcal{H}}(\boldsymbol{\pi}, \hat{\boldsymbol{\pi}}) = \boldsymbol{\lambda}_{(\boldsymbol{\pi})} \boldsymbol{\lambda}_{(\hat{\boldsymbol{\pi}})} \langle \phi_\Delta(\mathbf{H}, \boldsymbol{\pi}), \phi_\Delta(\mathbf{H}, \hat{\boldsymbol{\pi}}) \rangle$, $\mathbf{H} \subseteq \mathcal{H}$. For the current set of selected weak classifiers, the objective value of (5) is equal to the sum of the first two terms of (7). Hence the duality gap is in the last term of the objective function (7), $-\frac{1}{2} \sum_{\boldsymbol{\pi}, \hat{\boldsymbol{\pi}}} \boldsymbol{\mu}_{\tilde{\mathcal{H}}}(\boldsymbol{\pi}, \hat{\boldsymbol{\pi}})$. Clearly, minimizing this duality gap leads to the base learning rule (6), *i.e.*,

$$\begin{aligned} \hbar^*(\cdot) &= \underset{\hbar \in \tilde{\mathcal{H}}}{\operatorname{argmin}} \; -\frac{1}{2} \textstyle\sum_{\boldsymbol{\pi}, \hat{\boldsymbol{\pi}}} \boldsymbol{\lambda}_{(\boldsymbol{\pi})} \boldsymbol{\lambda}_{(\hat{\boldsymbol{\pi}})} \langle \phi_\Delta(\boldsymbol{h}, \boldsymbol{\pi}), \phi_\Delta(\boldsymbol{h}, \hat{\boldsymbol{\pi}}) \rangle \\ &= \underset{\hbar \in \tilde{\mathcal{H}}}{\operatorname{argmax}} \; \textstyle\sum_{\boldsymbol{\pi}, \hat{\boldsymbol{\pi}}} \boldsymbol{\lambda}_{(\boldsymbol{\pi})} \boldsymbol{\lambda}_{(\hat{\boldsymbol{\pi}})} \langle \phi_\Delta(\boldsymbol{h}, \boldsymbol{\pi}), \phi_\Delta(\boldsymbol{h}, \hat{\boldsymbol{\pi}}) \rangle \\ &= \underset{\hbar \in \tilde{\mathcal{H}}}{\operatorname{argmax}} \left| \textstyle\sum_{\boldsymbol{\pi}} \boldsymbol{\lambda}_{(\boldsymbol{\pi})} \big(\phi(\boldsymbol{h}, \boldsymbol{\pi}^*) - \phi(\boldsymbol{h}, \boldsymbol{\pi}) \big) \right|. \end{aligned}$$

3 Computational Complexity

We summarize the algorithm of our pAUCEns in Algorithm 1. To analyze the complexity, we decompose our detector into 3 stages: feature acquisition, finding weak learner and solving the pAUC optimization problem. We first analyze the complexity of acquiring low level features, which is performed once during the initialization. Let us assume we use gradient histogram features (or any features which can be computed efficiently in linear time with the use of integral images [14] or integral histograms [10]), the feature extraction step costs $\mathcal{O}(Bb)$ time, where B is the number of HOG blocks and b is the total number of histogram bins in each HOG block.

Step ① learns the weak classifier with the minimal weighted error and add this weak learner to the ensemble set. In this step, we train a weak classifier using weighted Fisher Linear Discriminant Analysis (WLDA) and decision stumps. WLDA projects multi-dimensional HOG features onto a line and we use the decision stump to learn the optimal threshold which gives a minimal weighted error. For each HOG block, WLDA can be solved efficiently using a generalized eigenvalue decomposition. This procedure costs $\mathcal{O}(Bb^3)$. For fast training of decision stumps, we first sort feature values and scan through all

	pAUCEns	AdaBoost
Feature extraction (once)	$\mathcal{O}(Bb)$	$\mathcal{O}(Bb)$
Traing weak learner - Step ① in Alg. 1	$\mathcal{O}\big(B(m+n)\log(m+n)+Bb^3\big)$	$\mathcal{O}\big(B(m+n)\log(m+n)+Bb^3\big)$
Solve $\boldsymbol{w}$ - Step ③ in Alg. 1	$\mathcal{O}\Big(r(m+n)\big(\log(m+n)+t_{\max}\big)\Big)$	$\mathcal{O}(m+n)$
Total	$\mathcal{O}\Big(t_{\max}\big[(B+r)(m+n)\log(m+n)+r(m+n)t_{\max}+Bb^3\big]\Big)$	$\mathcal{O}\Big(t_{\max}\big[B(m+n)\log(m+n)+Bb^3\big]\Big)$

Table 1: A comparison between the computational complexity of our approach and AdaBoost.

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. 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 [8];
④ Update sample weights, $\boldsymbol{u}$;
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 ;

possible thresholds sequentially [14]. Training decision stumps takes $\mathcal{O}\big(B(m+n)\log(m+n)\big)$ for sorting and scanning. Hence, training the weak learner at each iteration takes $\mathcal{O}\big(B(m+n)\log(m+n)+Bb^3\big)$.

We next analyze the time complexity of step ③ which calls Algorithm 2. Algorithm 2 solves the structural SVM problem using the efficient cutting-plane algorithm. Step ① in Algorithm 2 costs $\mathcal{O}\big(t_{\max}(m+n)\big)$ since the linear kernel scales linearly with the number of training samples [3]. Here $t_{\max}$ is the maximum number of features (weak classifiers). Using the efficient algorithm of [8], step ② costs $\mathcal{O}\big((m+n)\log(m+n)\big)$ time. As shown in [4], the number of iterations of Algorithm 2 is upper bounded by the value which is independent of the number of training samples. Here, we assume that the number of cutting-plane iterations required is bounded by r. In total, the time complexity of Algorithm 2 (Step ③ in Algorithm 1) is $\mathcal{O}\Big(r\big(\log(m+n)+t_{\max}\big)(m+n)\Big)$. Step ④ updates the sample variables which can be executed in linear time. In summary, the total time complexity for training $t_{\max}$ boosting iterations using our approach is $\mathcal{O}\Big(t_{\max}\big[(B+r)(m+n)\log(m+n)+r(m+n)t_{\max}+Bb^3\big]\Big)$. In contrast, the time complexity of AdaBoost based detector is $\mathcal{O}\Big(t_{\max}\big[B(m+n)\log(m+n)+Bb^3\big]\Big)$. We summarizes the compuational complexity of our approach in Table 1.

4 Experiments

Synthetic data set We first illustrate the effectiveness of our approach on a synthetic data set similar to the one used in [13]. The radius and angle of the positive data is drawn from a uniform distribution $[0, 1.5]$ and $[0, 2\pi]$, respectively. The radius of the negative data is drawn from a normal distribution with mean of 2 and the standard deviation of 0.4. The angle of the negative data is drawn from a uniform distribution similar to the positive data. We generate 400 positive data and 400 negative data for training and validation purposes (200 for training and 200 for validating the asymmetric parameter). For testing, we evaluate the learned classifier with 2000 positive and negative data. pAUCEns is compared against the baseline AdaBoost, Cost-Sensitive AdaBoost (CS-AdaBoost) [6] and Asymmetric AdaBoost

Algorithm 2 Cutting-plane algorithm for solving for the weak learners' coefficients

Input:
1) A set of weak learners' outputs $\mathbf{H} = (\mathbf{H}_+, \mathbf{H}_-)$;
2) The learning objective based on the partial AUC, α and β;
3) The regularization parameter, ν;
4) The cutting-plane termination threshold, ϵ;
Output: The weak learners' coefficients $\boldsymbol{w}$, the working set $\mathcal{C}$ and the dual variables $\boldsymbol{\lambda}$, ρ;
Initialize: $\mathcal{C} = \varnothing$;
$g(\mathbf{H}, \boldsymbol{\pi}, \boldsymbol{w}) = \Delta_{(\alpha,\beta)}(\boldsymbol{\pi}^*, \boldsymbol{\pi}) - \frac{1}{mn(\beta-\alpha)} \sum_{i,j} \pi_{ij} \boldsymbol{w}^\top (\boldsymbol{h}_i^+ - \boldsymbol{h}_j^-)$;
Repeat
① Solve the dual problem using linear SVM,

$$\min_{\boldsymbol{w},\xi} \frac{1}{2}\|\boldsymbol{w}\|_2^2 + \nu\,\xi \quad \text{s.t. } g(\mathbf{H}, \boldsymbol{\pi}, \boldsymbol{w}) \leq \xi, \forall \boldsymbol{\pi} \in \mathcal{C};$$

② Compute the most violated constraint,

$$\bar{\boldsymbol{\pi}} = \min_{\boldsymbol{\pi} \in \boldsymbol{\Pi}_{m,n}} g(\mathbf{H}, \boldsymbol{\pi}, \boldsymbol{w});$$

③ $\mathcal{C} \leftarrow \mathcal{C} \cup \{\bar{\boldsymbol{\pi}}\}$;
Until $g(\mathbf{H}, \boldsymbol{\pi}, \boldsymbol{w}) \leq \xi + \epsilon$;

(AsymBoost) [13]. For CS-AdaBoost, we set the cost for misclassifying positive and negative data as follows. We assign the asymmetric factor $k = C_1/C_2$ and restrict $0.5(C_1 + C_2) = 1$. We then choose the best k which returns the highest partial AUC from $\{0.5, 0.6, \cdots, 2.4, 2.5\}$. For AsymBoost, we choose the best asymmetric factor k which returns the highest partial AUC from $\{2^{-1}, 2^{-0.8}, \cdots, 2^{1.8}, 2^2\}$. For our approach, the regularization parameter is chosen from $\{10^{-4}, 10^{-3.5}, \cdots, 10^{1.5}, 10^2\}$.

We use vertical and horizontal decision stumps as the weak classifier. We evaluate the partial AUC of each algorithm at $[0, 0.2]$ FPRs. For each algorithm, we train a strong classifier consisting of 10 and 25 weak classifiers. Fig. 1 illustrates the boundary decision. Our approach outperforms all other asymmetric classifiers. At 10 weak classifiers, it achieves a pAUC score of 0.84 while the asymmetric classifier only achieves 0.82. We observed 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 in terms of the partial AUC score in a FPR range $[0, 0.2]$ with 10 and 25 weak classifiers. AdaBoost performs worst on this toy data set since it optimizes the overall classification accuracy, *i.e.*, it treats both positive and negative samples equally. 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.

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. The asymmetric factor and the regularization parameter are cross-validated as previously described. 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 partial AUC score. We observe that our approach places more emphasis on positive samples at the corners (at $\pi/4$, $3\pi/4$, $-\pi/4$ and $-3\pi/4$ angles) than other algorithms.

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 [16], AsymBoost [13] and CS-AdaBoost [6]. 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 partial AUC score by varying the threshold value in the FPR range $[0, 0.1]$. For each

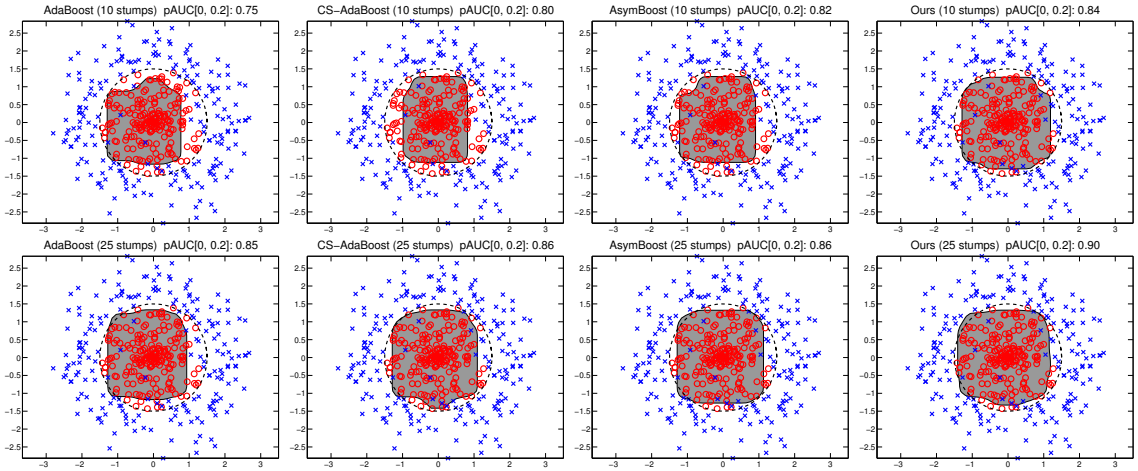

Figure 1: Decision boundaries on the toy data set where each strong classifier consists of **Top row:** 10 weak classifiers and **Bottom row:** 25 weak classifiers. The partial AUC score at FPR in the range $[0, 0.2]$ is also displayed. Our approach achieves the best pAUC score of 0.84 and 0.9 at 10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

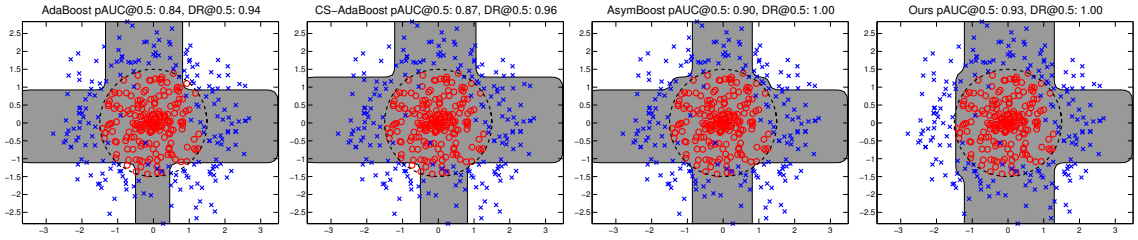

Figure 2: Decision boundaries on a toy data set with 10 weak classifiers at FPR of 0.5 The partial AUC score and detection rate at 50% false positive rate are also shown. Our approach performs best on both evaluation criteria. Our approach preserves a larger decision boundary near positive samples at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

algorithm, the experiment is repeated 20 times and the average partial AUC 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. For USPS, we use raw pixel values and categorize the data sets into two classes: even digits and odd digits. For scenes, we divide the 15-scene data sets used in [5] into 2 groups: indoor and outdoor scenes. We use CENTRIST as our feature descriptors and build 50 visual code words using the histogram intersection kernel [17]. Each image is represented in a spatial hierarchy manner. Each image consists of 31 sub-windows. In total, there are 1550 feature dimensions per image. For faces, we use face data sets from [14] and randomly extract 5000 negative patches from background images. We apply principle component analysis (PCA) to preserve 95% total variation. The new data set has a dimension of 93.

Pedestrian detection - Strong classifier We evaluate our approach on the pedestrian detection task. We train our approach on the INRIA pedestrian data set. There are 2416 cropped mirrored pedestrian images and 1200 large background images in the training set. The test set contains 288 images containing 588 annotated pedestrians. For the positive training data, we use all 2416 INRIA cropped pedestrian images. Each training sample is scaled to 64×128 pixels with 16 pixels additional borders for preserving the contour information. 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 [1] and covariance (COV) features[1] [12]. For both HOG and COV features, we define blocks with different scales (a minimum of 12×12 pixels, and a maximum of 64×128 pixels) and width-length ratios ($1:1$, $1:2$, $2:1$, $1:3$, and $3:1$). Each block is divided into 2×2 cells. For HOG, the number of orientation bins is set to 9. Thus there are 36-dimensional HOG features generated in each block. For COV features, we use the following image statistics $\Big[x,\, y,\, I,\, |I_x|,\, |I_y|,\, \sqrt{I_x^2 + I_y^2},\, |I_{xx}|,\, |I_{yy}|,\, \arctan(|I_x|/|I_y|)\Big]$, where x and y are the pixel location, I is the pixel intensity, I_x and I_y are first order intensity derivatives, I_{xx} and I_{yy} are second order intensity derivatives and the edge orientation. Each pixel is mapped to a 9-dimensional feature image. We then calculate 36 correlation coefficients in each block and concatenate these features to previously computed HOG features. ℓ_1-norm normalization is then applied independently to HOG and COV feature vector. Given a 64×128-pixels image, there is a total of 7735 blocks . Furthermore, we use integral image to speed up the computation as in [2]. At each iteration, we randomly sample 10% of the whole possible blocks for training a weak classifier. We have used weighted linear discriminant analysis (WLDA) as weak classifiers. We train 500 weak classifiers and set 5 multi-exits [9]. To be more specific, we set the threshold at 10, 20, 50, 100 and 200 weak classifiers. These exits reduce the evaluation during testing significantly. The regularization parameter ν is cross-validated from $\{0.1, 0.5, 1, 2, 10\}$ and the pAUC range is set to $[0, 0.1]$. 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 [2].

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 maximize the pAUC in the FPR range $[0.49, 0.51]$. We train a multi-exit cascade [9] with 19 exit. The number of weak classifiers in each node is set as follows: 5, 5, 10, 10, 20, 20, 40, 40, 80, $\cdots$. At each exit, correctly classified negative samples are replaced by incorrectly classified negative samples bootstrapped from a large set of background images. In this experiment, we use the software of [11] to compute the partial AUC score in the FPPI range: $[0, 0.1]$ and $[0, 1]$, and report experimental results in Table 3. Here we only compare the detectors which are trained on the INRIA training set. At the FPPI range $[0, 0.1]$, 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. At the FPPI range $[0, 1]$, our approach performs best on ETH test set.

[1] 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) [7] and self-similarity of low-level features (CSS) [15].

	Ours	**ChnFtrs**	**ConvNet**	**CrossTalk**	**FPDW**	**FeatSynth**	**FtrMine**	**HOG**	**HikSvm**	**HogLbp**	**LatSvm-V1**	**LatSvm-V2**	**MultiFtr**	**Pls**	**PoseInv**	**Shapelet**	**VJ**	**VeryFast**
Complete set - Partial AUC(0,0.1) score (in %)																		
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	**71.4**	77.3	82.1	74.4	81.3	-	-	89.9	93.5	93.0	96.1	84.5	85.0	85.8	95.5	98.2	97.8	-
ETH	67.5	75.8	68.5	70.6	77.8	-	-	80.0	86.3	72.2	87.5	**67.3**	75.9	73.4	98.3	97.6	95.6	72.6
Caltech-UsaTest	89.2	87.6	94.4	**86.7**	88.6	-	93.8	94.4	95.2	91.4	95.4	90.5	91.5	89.5	97.9	98.3	99.6	-
Reasonable (min. 50 pixels tall & nopartial occlusion) - Partial AUC(0,0.1) score (in %)																		
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
Caltech-UsaTest	68.3	66.9	82.3	**61.8**	68.2	-	81.7	79.3	85.0	72.9	85.2	69.9	76.6	71.1	93.9	94.8	96.9	-
Large (min. 100 pixels tall) - Partial AUC(0,0.1) score (in %)																		
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
Caltech-UsaTest	36.4	37.0	30.9	33.4	41.0	-	70.9	49.2	56.6	**27.9**	60.5	34.5	52.5	43.2	92.0	82.2	90.5	-
Near (min. 80 pixels tall) - Partial AUC(0,0.1) score (in %)																		
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
Caltech-UsaTest	44.0	43.4	48.2	40.0	45.5	-	67.6	58.4	65.8	**38.1**	67.7	42.6	59.4	49.0	91.8	88.6	93.5	-
Medium (min. 30 pixels tall and max. 80 pixels tall) - Partial AUC(0,0.1) score (in %)																		
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
Caltech-UsaTest	86.3	84.3	97.0	**83.2**	85.3	-	91.8	92.6	93.8	94.0	95.3	89.7	89.2	87.1	96.6	98.2	99.7	-
Far - Partial AUC(0,0.1) score (in %)																		
Caltech-UsaTest	100.0	**96.5**	100.0	97.3	96.8	-	98.0	97.6	99.6	100.0	99.1	98.0	97.8	99.2	100.0	99.8	99.6	-
Partial occlusion - Partial AUC(0,0.1) score (in %)																		
Caltech-UsaTest	85.3	84.6	91.2	82.6	86.8	-	93.1	91.1	95.9	84.2	92.4	85.5	91.8	**81.4**	97.9	95.3	99.8	-
Heavy occlusion - Partial AUC(0,0.1) score (in %)																		
Caltech-UsaTest	96.8	97.1	98.5	**96.3**	98.2	-	98.9	97.7	97.0	97.6	97.9	97.1	97.7	97.0	99.1	99.1	98.9	-
Atypical - Partial AUC(0,0.1) score (in %)																		
Caltech-UsaTest	82.2	77.9	86.5	**73.2**	82.4	-	93.3	94.3	94.2	89.0	92.8	83.3	90.3	85.5	97.0	97.1	99.2	-

Table 2: 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**.

	Ours	**ChnFtrs**	**ConvNet**	**CrossTalk**	**FPDW**	**FeatSynth**	**FtrMine**	**HOG**	**HikSvm**	**HogLbp**	**LatSvm-V1**	**LatSvm-V2**	**MultiFtr**	**Pls**	**PoseInv**	**Shapelet**	**VJ**	**VeryFast**
Complete set - Partial AUC(0,1.0) score (in %)																		
INRIA-Fixed	16.0	13.3	12.0	12.7	13.6	19.0	43.8	32.9	29.9	26.2	28.8	12.9	24.2	29.0	66.7	66.8	59.4	**10.3**
TudBrussels	55.7	57.6	66.8	**55.0**	59.0	-	-	73.6	76.4	77.2	85.7	67.2	70.5	66.1	83.8	93.8	92.7	-
ETH	**41.4**	48.7	47.1	43.8	51.5	-	-	54.9	61.6	51.1	69.1	49.3	51.7	47.4	86.5	85.6	84.5	46.9
Caltech-UsaTest	82.1	**77.1**	90.9	77.8	78.1	-	86.7	85.5	86.8	87.9	91.7	84.2	83.4	81.2	92.6	95.4	99.1	-
Reasonable (min. 50 pixels tall & nopartial occlusion) - Partial AUC(0,1.0) score (in %)																		
INRIA-Fixed	16.0	13.3	12.0	12.7	13.6	19.0	43.8	32.9	29.9	26.2	28.8	12.9	24.2	29.0	66.7	66.8	59.4	**10.3**
TudBrussels	49.5	48.8	59.1	**47.0**	50.4	-	-	68.1	72.4	71.8	84.0	59.6	64.8	59.1	80.8	92.5	91.1	-
ETH	**37.1**	44.2	38.9	39.1	46.8	-	-	51.1	59.2	43.7	66.6	41.1	47.5	42.2	85.2	83.9	83.6	42.5
Caltech-UsaTest	50.7	46.4	71.5	**46.0**	46.9	-	66.3	57.8	62.0	62.2	73.4	56.0	59.3	52.9	78.2	87.0	91.8	-
Large (min. 100 pixels tall) - Partial AUC(0,1.0) score (in %)																		
INRIA-Fixed	14.4	11.6	10.5	10.7	11.7	17.3	42.8	31.8	28.5	24.8	27.2	9.9	22.6	27.8	66.4	66.1	59.0	**9.1**
TudBrussels	39.1	36.2	**33.5**	37.3	35.0	-	-	56.2	52.2	46.3	64.5	43.1	55.5	43.3	70.0	80.3	86.0	-
ETH	24.8	30.2	**24.4**	28.2	33.4	-	-	33.1	36.4	29.5	47.6	26.8	35.1	26.6	63.9	75.8	76.7	24.4
Caltech-UsaTest	26.5	24.1	**14.8**	25.8	26.4	-	47.8	28.0	26.5	18.4	40.7	22.5	34.3	30.4	54.5	69.6	80.9	-
Near (min. 80 pixels tall) - Partial AUC(0,1.0) score (in %)																		
INRIA-Fixed	14.4	11.6	11.3	11.0	11.9	17.3	42.6	31.5	28.5	24.7	27.5	11.1	22.7	27.7	66.0	66.1	58.7	**9.7**
TudBrussels	**37.5**	39.5	40.4	40.3	38.8	-	-	61.1	58.7	50.5	70.9	47.1	57.2	49.6	80.0	85.6	89.0	-
ETH	29.4	35.2	**28.9**	30.9	37.5	-	-	40.5	45.6	31.7	52.2	31.4	39.4	34.1	80.6	79.9	80.0	29.8
Caltech-UsaTest	28.6	27.4	27.3	28.9	28.4	-	48.9	33.1	34.3	**24.7**	47.2	26.7	40.8	31.2	66.8	75.7	85.3	-
Medium (min. 30 pixels tall and max. 80 pixels tall) - Partial AUC(0,1.0) score (in %)																		
INRIA-Fixed	100.0	100.0	33.2	99.7	100.0	100.0	100.0	100.0	100.0	85.3	85.3	99.7	86.1	100.0	99.7	99.7	91.5	**27.9**
TudBrussels	58.8	57.4	67.8	**55.5**	59.7	-	-	71.4	74.9	82.9	85.5	68.2	68.7	65.0	79.4	94.1	91.7	-
ETH	44.0	42.9	55.4	**42.1**	45.4	-	-	49.9	54.7	61.2	71.5	57.3	47.3	45.0	73.9	74.5	71.2	48.3
Caltech-UsaTest	76.3	**69.5**	92.2	70.6	70.6	-	82.1	81.4	82.6	91.5	91.1	80.8	77.8	75.8	88.8	94.7	98.7	-
Far - Partial AUC(0,1.0) score (in %)																		
Caltech-UsaTest	100.0	**93.4**	100.0	95.4	94.2	-	95.0	96.2	98.2	100.0	97.8	97.4	95.9	98.7	100.0	99.9	99.3	-
Partial occlusion - Partial AUC(0,1.0) score (in %)																		
Caltech-UsaTest	68.2	**61.3**	81.9	69.2	66.9	-	81.6	77.1	80.3	75.0	84.1	76.7	78.6	68.1	85.7	90.5	96.9	-
Heavy occlusion - Partial AUC(0,1.0) score (in %)																		
Caltech-UsaTest	92.0	91.3	96.3	**90.7**	91.8	-	95.5	93.7	93.2	95.6	94.4	93.3	95.0	92.3	97.2	97.3	98.2	-
Atypical - Partial AUC(0,1.0) score (in %)																		
Caltech-UsaTest	64.8	57.3	74.5	**55.4**	61.9	-	76.8	75.3	77.4	77.2	85.8	70.8	73.3	69.1	86.0	92.1	93.6	-

Table 3: 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 $[0, 1]$. A **smaller pAUC score** means a **better detector**.

References

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

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

[3] T. Joachims. Training linear svms in linear time. In *Proc. of Intl. Conf. on Knowledge Discovery and Data Mining*, 2006.

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

[5] S. Lazebnik, C. Schmid, and J. Ponce. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, New York City, USA, 2006.

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

[7] 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.

[8] H. Narasimhan and S. Agarwal. A structural svm based approach for optimizing partial auc. In *Proc. Int. Conf. Mach. Learn.*, 2013.

[9] 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.

[10] F. Porikli. Integral histogram: A fast way to extract histograms in cartesian spaces. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, 2005.

[11] 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.

[12] 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.

[13] 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.

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

[15] 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.

[16] 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.

[17] J. Wu and J. M. Rehg. CENTRIST: A visual descriptor for scene categorization. *IEEE Trans. Pattern Anal. Mach. Intell.*, 33(8):1489–1501, 2011.