Improving multi-class Boosting-based object detection

José Miguel Buenaposada [a,*] and Luis Baumela [b]
[a] *ETSII, Universidad Rey Juan Carlos, Móstoles, Spain*
E-mail: josemiguel.buenaposada@urjc.es
[b] *Departamento de Inteligencia Artificial, Universidad Politécnica de Madrid, Spain*
E-mail: lbaumela@fi.upm.es

Abstract. In recent years we have witnessed significant progress in the performance of object detection in images. This advance stems from the use of rich discriminative features produced by deep models and the adoption of new training techniques. Although these techniques have been extensively used in the mainstream deep learning-based models, it is still an open issue to analyze their impact in alternative, and computationally more efficient, ensemble-based approaches. In this paper we evaluate the impact of the adoption of data augmentation, bounding box refinement and multi-scale processing in the context of multi-class Boosting-based object detection. In our experiments we show that use of these training advancements significantly improves the object detection performance.

Keywords: Object Detection, Multi-class Boosting, Data augmentation, Bounding Box adjust

1. Introduction

Visual object detection is the first step in many relevant Computer Vision problems such as facial facial landmarks detection [1], body pose estimation [2], self driving cars [3], vehicle type analysis [4], text reading [5], etc. The object detection pipeline can be described in terms of three basic components: 1) *image features*: Haar-like-features, Histograms of Oriented Gradient (HoGs) features, Integral Channel Features (ICF), Locally Decorrelated Channel Features (LDCF), Convolutional Neural Networks (CNNs), etc; 2) *classification algorithm*: AdaBoost, RealBoost, Support Vector Machines (SVMs), Neural Nets, etc; 3) *detection strategy*: sliding window, candidate window proposals and bounding box regression, direct regression from image features (Single Shot Detectors), Hough voting, etc.

Since the publication of Viola and Jones [6] seminal work, the performance of object detection algorithms has improved by changing several parts of the detection pipeline. Visual features have evolved from being hand-crafted [7] to automatically selected from a pool with Boosting [8], Random Forest [9] or learned with a CNN [10]. The brute force sliding window approach [6–8] has evolved into fast region proposal algorithms [5, 10] and more recently Single Shot Detectors [11, 12]. The Boosting approach has received much attention because it is computationally very efficient and achieves very good performance in various object detection problems such as pedestrians [13, 14], multi-view faces [15] or multi-view cars [16]. The key to its success is the use of the feature selection capabilities of Boosting together with different pooling strategies [13] and rich image descriptions such as the ICF [8]. Further, object detection has evolved from detecting a single object category to being able to detect multiple categories and different views at the same time [17]. The usual framework for Boosting-based object detection uses binary classification (e.g. AdaBoost). In this regard, essentially multi-class detection problems, such as face [15] or car [16] detection, have been usually solved with binary classi-

*Corresponding author. E-mail: josemiguel.buenaposada@urjc.es.

fiers: either with a monolithic detector (i.e. Object-vs-Background) or with a detector per object view or positive class (i.e. using K detectors).

Powered by the use of deep neural nets [18, 19], modern object detection algorithms have achieved remarkable precision [10–12]. This is due to the use of discriminative features produced by deep models and the adoption of new training strategies. However, these approaches require advanced computational resources, such as Graphical Processing Units (GPUs) to achieve real-time performance. This prevents their use in devices with limitations in computing power or energy supply, such as aerial vehicles, micro-robots or mobile phones. Hence, the necessity of developing very efficient object detection algorithms, such as, for example, those based on Boosting approaches.

One of such methodological advances in object detection is the refinement of the bounding box of the detected objects. Classifiers are usually trained with fixed bounding box size and aspect ratio (AR). The bounding box refinement step is crucial to achieve top performance when dealing with objects showing different aspect ratios depending on their pose or configuration. Modern CNN-based detectors such as Faster RCNN [10], YOLO [11] and SSD [12] perform bounding box regression. Also, some of the best results in the KITTI car detection benchmark [20, 21] iteratively adjust the detection bounding box. In section 4 we introduce a bounding box refinement scheme. The most similar approach to ours was introduced by Juránek *et al*. [22]. They use a Real AdaBoost binary classifier for car detection. In their work the estimation of the 3D orientation of the car is performed using a similar scheme to our bounding box aspect ratio estimation. However, in their solution the bounding box aspect ratio is fixed and depends on the car orientation. This is a limitation, since cars with the same orientation may have different aspect ratios.

Another training improvement is the augmentation of training data by producing realistic image transformations that do not change their labels. In the SSD detector [12] the newly generated data accounts for an 8.8% increase of the performance in the VOC 2007 test dataset. It is also a common strategy in the top performing car detectors in the KITTI dataset [21]. An alternative way of using additional data is to pre-train the model in a related but different dataset. All top performing algorithms in KITTI cars detection use pre-trained models in the ImageNet classification problem [20, 21, 23]. In the Boosting detection literature it is typical to use the geometric augmentation with image translation, rotation [8], image horizontal flip [24] and aspect ratio changes [25].

Recently, theoretically sound results in the context of multi-class Boosting provide new tools to address the unbalanced and asymmetric classification problems arising in object detection [24]. In this paper we endow this algorithm with modern training strategies, such as data augmentation, bounding box regression and multi-scale processing and evaluate the increase in performance achieved with these improvements.

2. Multi-class boosting algorithm

A Boosting classification algorithm is a supervised learning scheme that builds a binary prediction model by combining a collection of simpler base models or weak learners [26]. It receives as input a set of N training data instances $\{(\mathbf{x}_i, l_i)\}_{i=1}^{N}$, where $\mathbf{x}_i \in \mathbb{R}^m$ represents an object to be classified with class label l_i. In our multi-class Boosting scheme [27] each label $l_i \in L = \{1, 2, \ldots, K\}$ has a corresponding margin vector $\mathbf{y}_l \in Y$, where $Y = \{\mathbf{y}_1, \ldots, \mathbf{y}_K\}$. $\mathbf{y}_l$ has a value 1 in the l-th coordinate and $\frac{-1}{K-1}$ elsewhere. So, if $l = 1$, the margin vector representing class 1 is $\mathbf{y}_1 = \left(1, \frac{-1}{K-1}, \ldots, \frac{-1}{K-1}\right)^\top$. Hence, it is immediate to see the equivalence between classifiers G defined over L and classifiers $\mathbf{g}$ defined over Y, $G(\mathbf{x}) = l \in L \Leftrightarrow \mathbf{g}(\mathbf{x}) = \mathbf{y}_l \in Y$.

2.1. BAdaCost: Cost-sensitive Multi-class Boosting classification

Cost-sensitive classification endows the traditional Boosting scheme with the capability to modify pairwise class boundaries. In this way, we can reduce the number of errors between positive classes (e.g. different target orientations) and improve recall when object classes have different aspect ratios. To this end we use BAdaCost [27] (Boosting Adapted for Cost matrix), a recently introduced multi-class cost-sensitive Boosting classifier. In this section we briefly introduce it.

Costs are encoded in a $K \times K$-matrix $\mathbf{C}$, where each entry $C(i, j)$ represents the cost of miss-classifying an instance with real label i as j. Here it is assumed that $C(i, i) = 0, \forall i \in L$, i.e. the cost of correct classifications is zero.

Let $\mathbf{C}^*$ be a $K \times K$-matrix defined in the following way

$$C^*(i,j) = \begin{cases} C(i,j) & \text{if } i \neq j \\ -\sum_{h=1}^{K} C(j,h) & \text{if } i = j \end{cases}, \forall i,j \in L. \tag{1}$$

In a cost-sensitive classification problem each value $C^*(j,j)$ represents a "reward" associated to a correct classification. The j-th row in $\mathbf{C}^*$, denoted as $\mathbf{C}(j,-)$, is a margin vector that encodes the costs associated to the j-th label. The multi-class cost-sensitive margin associated to instance $(\mathbf{x}, l)$ is given by $z_C := \mathbf{C}^*(l,-)\cdot\mathbf{g}(\mathbf{x})$. It is easy to verify that if $\mathbf{g}(\mathbf{x}) = \mathbf{y}_i \in Y$, for a certain $i \in L$, then $\mathbf{C}^*(l,-)\cdot\mathbf{g}(\mathbf{x}) = \frac{K}{K-1}\mathbf{C}^*(l,i)$. Hence, using this generalized margin, BAdaCost defines a *Cost-sensitive Multi-Class Exponential Loss Function* (*CMELF*):

$$\begin{aligned} \mathcal{L}_C(l, \mathbf{g}(\mathbf{x})) :&= \exp(z_C) \\ &= \exp\left(\mathbf{C}^*(l,-)\cdot\mathbf{g}(\mathbf{x})\right) \\ &= \exp\left(\frac{K}{K-1}\mathbf{C}^*(l, G(\mathbf{x}))\right). \end{aligned} \tag{2}$$

The margin, z_C, yields negative values when the classification is correct under the cost-sensitive point of view, and positive values for costly (wrong) outcomes. The CMELF is a generalization of the Multi-class Exponential Loss introduced in [28].

BAdaCost resorts to the CMELF (2) for evaluating classifications encoded with margin vectors. The expected loss is minimized using a stage-wise additive gradient descent approach. The strong classifier that arises has the following structure:

$$\begin{aligned} H(\mathbf{x}) &= \arg\min_k \left(\mathbf{C}^*(k,-)\cdot\sum_{m=1}^{M}\beta_m\mathbf{g}_m(\mathbf{x})\right) \\ &= \arg\min_k \left(\mathbf{C}^*(k,-)\cdot\mathbf{f}(\mathbf{x})\right), \end{aligned} \tag{3}$$

where $\mathbf{f}(\mathbf{x})$ is a linear combination of M cost-sensitive weak learners, $\{\mathbf{g}_m(\mathbf{x})\}_{m=1}^{M}$, that the algorithm learns incrementally. In this case $\mathbf{f}(\mathbf{x})$ is a vector with the estimated class margin vector from the feature vector $\mathbf{x}$.

2.2. Object detection score for BAdaCost

When building an object detector it is necessary to have a confidence measure or detection score. In BAdaCost the predicted costs incurred when classifying sample $\mathbf{x}$ in one of the K classes are given by the vector:

$$\mathbf{c} = \mathbf{C}^*\cdot\mathbf{f}(\mathbf{x}) = (c_1, \ldots, c_K)^\top. \tag{4}$$

From now on, in multi-class detection problems, we assume that the background (negative) class has label $l = 1$ and the object classes (e.g. different views of a car) have label $l > 1$. Therefore, we can compute the score of $\mathbf{x}$ as

$$s(\mathbf{x}) = (c_1 - min(c_2, \ldots, c_K)). \tag{5}$$

This score has desirable properties for detection problems: 1) $s(\mathbf{x}) > 0$ when the winner class (i.e. the class with lowest cost) has label $l > 1$; 2) $s(\mathbf{x}) < 0$ when the winner class is $l = 1$. With this score we can improve the classifier efficiency by using a cascade calibration algorithm, for example [29], and stop the evaluation of weak learners whenever the score falls below a calibrated threshold.

3. Boosting Data Augmentation (DA)

One of the main problems for training an object detector is the limited amount of training data, that could cause the classifiers to overfit. The solution adopted in the literature is to generate new synthetic data from the training set. This is known as data augmentation (DA).

We model the object to be detected as multiple positive classes depending on the orientation, see Fig. 6. To augment and balance our dataset we increase the number of images in the small classes. To this end we sequentially apply a combination of basic photometric changes to the image RGB values (see Fig. 1). We start by generating a random number r in the range $[0, 1]$. If $r > 0.5$ we make the following transformations:

1. Brightness change,
2. Contrast change,
3. Saturation change,
4. Hue Change.

Otherwise:

1. Brightness change,
2. Saturation change,
3. Hue Change,
4. Contrast change.

These photometric changes are implemented as:

– **Brightness change.** Generate a random number r_b in the range $[0,1]$. If $r_b < p_b$ a random value in the range $[-\Delta b, \Delta b]$ is added to all pixel RGB values.
– **Contrast change.** Generate a random number r_c in the range $[0,1]$. If $r_c < p_c$ all pixel values are multiplied by a random value in the range $[c_{low}, c_{high}]$.
– **Saturation change.** Generate a random number r_s in the range $[0,1]$. If $r_s < p_s$ the image is transformed to the HSV color space. All pixel saturation values in the image are multiplied by a random value in the range $[s_{low}, s_{high}]$ and the resultant HSV values are transformed back to RGB.
– **Hue change.** Generate a random number r_h in the range $[0,1]$. If $r_h < p_h$ the image is transformed to the HSV color space, a random value in the range $[-\Delta h, \Delta h]$ is added to all hue values in the image and the new HSV values are transformed back to RGB.

As we show in section 6.1 the generation of synthetic data is crucial to achieve a 5.5% improvement in detection results (see Table 2, Moderate). In our detection experiments some views have fewer examples. The use of photometric augmentation allows us to balance the dataset. Further, it allows us to increase the classifier capacity with more and deeper decision trees. This is consistent with similar results in the problem of pedestrian detection [25].

4. Bounding Box Aspect Ratio estimation (BB)

As we show in our experiments, refining the bounding box estimation increases the performance of our detector. To this end we modify the multi-class Boosting classifier introduced in section 2.

The classifier learns m cost-sensitive decision trees with Gini impurity measure. The modifications of the tree to make it cost-sensitive are two-fold:

1. On each tree node S, the probability of class l, $p(l)$, is multiplied by the percentage of costs associated to class l (i.e. the sum of a cost matrix row), $c(l)$:

$$p'(l) = \underbrace{\frac{\sum_{i \in S} w_i \cdot I(l_i = l)}{\sum_{i \in S} w_i}}_{p(l)} \underbrace{\frac{K \cdot \sum_{k=1}^{K} C(l,k)}{\sum_{i=1}^{K}\sum_{j=1}^{K} C(i,j)}}_{c(l)}, \quad (6)$$

where $I()$ is the indicator function, and w_i is the weight assigned by the BAdaCost algorithm to the i-th datum in node S. Then, $p'(l)$ is used in the Gini impurity.
2. Then we classify with the minimum cost rule on each leaf node S:

$$h_S = \arg\min_l (p(1), \ldots, p(K)) \cdot \mathbf{C}(-, l) \quad (7)$$

where $p(l) = \frac{\sum_{i \in S} w_i \cdot I(l_i = l)}{\sum_{i \in S} w_i}$ and $\mathbf{C}(-,l)$ is the l-th column of the cost matrix.

During the training phase of the baseline BAdaCost algorithm, for every decision tree and leaf node S, we store the minimum cost label, h_S. During the detection phase, the trees are traversed with the feature vector $\mathbf{x}$ corresponding to a candidate window (see Fig.2). As we have seen in section 2, we compute vector $\mathbf{f}(\mathbf{x})$ as a linear combination of tree labels h, coded as margin vectors $\mathbf{g}(\mathbf{x}) = \mathbf{y}_h \in Y$. Then we use vector $\mathbf{f}(\mathbf{x})$ in equation (3) to obtain the minimum cost view class estimation. To correct the bounding box we have two options:

A. The baseline BAdaCost detector computes the mean AR of each class from the ground truth training data bounding boxes in a vector $\mathbf{a}[h]$. The final estimated class, h, is that corresponding to the minimum value in equation (3) using the estimated $\mathbf{f}(x) = \sum_m \beta_m \mathbf{g}_m(\mathbf{x})$. The aspect ratio of each detection is that associated to h.
B. Our new procedure to estimate the aspect ratio follows a similar approach to the computation of $\mathbf{f}(\mathbf{x})$ in BAdaCost. The m-th WL tree outputs two vectors of size K (i.e. number of classes): a) the $\mathbf{g}_m(\mathbf{x})$ margin vector that codifies the estimated class and b) the $\mathbf{a}_m(\mathbf{x})$ vector of per class mean aspect ratios of the training data that fall in the corresponding leaf node. Then, both sets of vectors are added using the boosting weight of each tree, generating $\mathbf{f}(x) = \sum_m \beta_m \mathbf{g}_m(\mathbf{x})$ vector (for classes) and $\mathbf{a}(\mathbf{x}) = \sum_m \beta_m \mathbf{a}_m(\mathbf{x})$ vector (for aspect ratios). The final estimated class, h, is that corresponding to the minimum value in equation (3) using the estimated $\mathbf{f}(\mathbf{x})$. The aspect ratio is that corresponding to the estimated class, $\mathbf{a}(\mathbf{x})[h]$ (see Fig. 2).

We correct each window keeping the height constant and adapting the AR.

Fig. 1. Sample KITTI photometric data augmentation. In the first column we display the original training image, followed by 8 generated images.

Note that in option B we may exclude the first $t_p - 1$ trees from the model. The rationale is that in the first trees the classifier is not accurate enough. In the experimental section we will see that this is in fact true, but the difference in performance is marginal. The overall approach to simultaneous detection and AR estimation is shown in Fig. 2.

5. Multi-scale detection

The classifiers used in object detection are typically learned with a fixed scale. Both Boosting-based [8, 13, 15, 24, 25, 30] and CNN-based [10, 11] detectors are all trained with a fixed base image size. However, objects of the same class at different sizes usually need different features to be detected. CNN-based detectors overcome this problem using feature maps extracted from layers with different resolutions [12, 31]. Boosting-based detectors use a fixed size classifier in an sliding window on a pyramid of feature maps extracted at decreasing image resolutions. This enables the detection of objects at different sizes. The detection rate also improves using a classifier trained at several base sizes [16, 32].

In this paper we use a multi-scale (MS) detection approach training different BAdaCost detectors at different base sizes. For the KITTI car dataset, our multi-scale approach trains three classifiers with 1.75 aspect ratio: 48×84, 56×98 and 64×112 pixels. For the KITTI cyclists we also train three classifiers but in this case with aspect ratio 1.5: 48×72, 56×84 and 64×96 pixels. Note here that the fixed aspect ratio is only used to detect the presence of the target. Then the detected window is adjusted with the procedure described in section 4. Finally, we slide these detectors along an efficiently estimated feature pyramid [8, 33], see Fig. 3.

As we show in Fig. 5, features selected at different scales show better definition of interesting areas as the resolution increases. In the car problem, these features are mainly around the rear lights and the edges that define the car shape at different orientations (see Fig. 4). In section 6 we show that we achieve a significant improvement in recall using detectors trained at different scales.

6. Experiments

To make our experiments we have added BAdaCost's cost-sensitive decision trees and multi-class detection to Piotr Dollar's Matlab Toolbox[1]. Our implementation is publicly available at https://github.com/jmbuena/toolbox.badacost.public.

In the experiments we use detection problems where the target object changes its Bounding Box AR depending on the view angle. Car and cyclist detection in the KITTI dataset [34] are good examples of this kind of problems. The database presents three subsets: easy, moderate and hard (easy ⊂ moderate ⊂ hard). We carry out the evaluation in each level separately. In total there are 7481 images for training and 7518 for testing. Since the testing images have no ground truth, we split the training set into training and validation subsets: the first 6733 images (90%) for training (KITTI-train90) and the last 748 images (10%) for validation (KITTI-train10). In all the experiments we use cost-sensitive decision tree weak learners that select features from LDCF channels [33]. We augment our data

[1] https://github.com/pdollar/toolbox

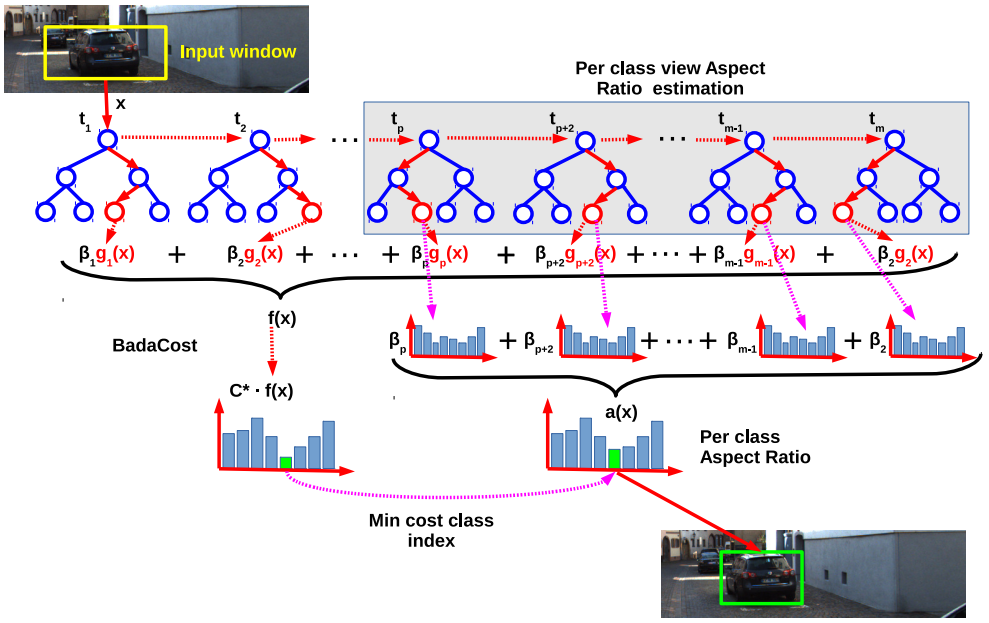

Fig. 2. Bounding Box aspect ratio estimation. We learn an ensemble of multi-class cost-sensitive trees. We estimate the AR distribution using all the trees from t_p onwards. The estimated AR is that of the minimal cost class.

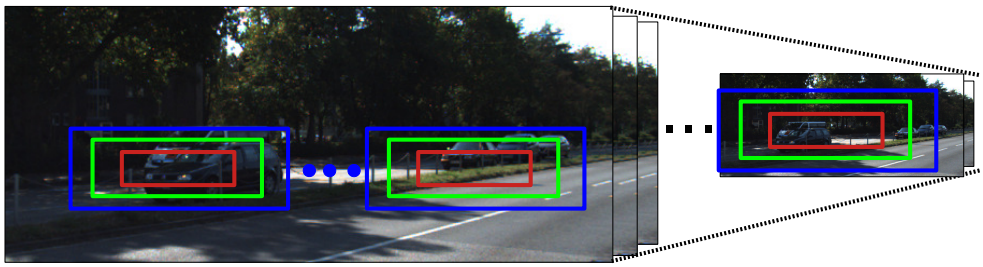

Fig. 3. Multi-scale detection. We apply classifiers trained with different base scale (i.e. blue, green and red rectangles) to detect objects with various dimensions.

by flipping images horizontally. To compare detectors in the KITTI dataset we use the Average Precision (AP), computed using 11 (AP_{11}) and 40 (AP_{40}) points in the precision-recall curve [35]. The traditional performance measure in KITTI was AP_{11}. However, since October 2019 it has changed to AP_{40}. Thus, we report both. We select the classifier parameters by grid search in the validation set, KITTI-train10.

For data augmentation we use the following parameters (see section 3): $p_b = 0.5$, $\Delta b = 32$ for brightness; $p_c = 0.5$, $c_{low} = 0.5$, $c_{high} = 1.5$ for contrast; $p_s = 0.5$, $s_{low} = 0.5$, $s_{high} = 1.5$ for saturation; and $p_h = 0.5$, $\Delta h = 18$ for hue changes.

6.1. Car detection experiments

Here we improve the results in our previous work [36] by increasing the number of view classes (K=20 to 25), the depth of the weak learners trees (D=8 to 10) and the regularization (shrinkage from 0.1 to 0.05, fraction of features searched in the training of each weak learner from 1/16 to 1/32). Increasing tree depths improves classifier capacity, hence enabling the use of more orientation classes and achieving better predictions, but at the expense of stronger regularization. Therefore, in the experiments we divide the car training images into K=25 view classes (see Fig. 6) and use KITTI-train90 for training and KITTI-train10 for validation. In all the experiments we train a 48×84 pixels car model, AR=1.75. We start the pyramid one octave above the actual image size.

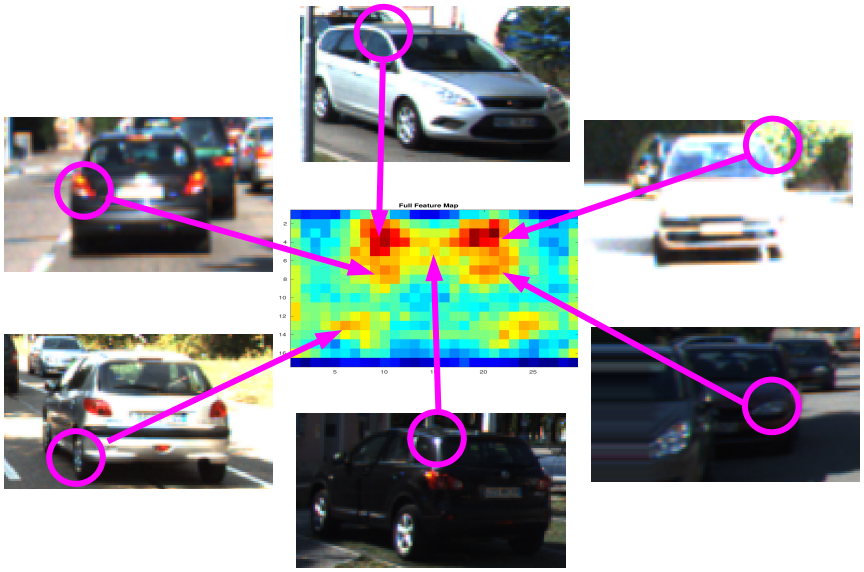

Fig. 4. Car parts and location of the features selected by the detector.

During training, we perform 4 rounds of hard negatives mining with the KITTI training image subset (KITTI-train90). The best number of cost-sensitive trees is T=1024 (4 rounds with 32, 128, 256 and T weak learners, respectively), tree depth is D=10, the number of negatives per round is N=7500 and the total amount of negatives used in the final round is NA=30000.

The cost matrix is set to weigh up gross errors between view classes. This is important because estimating the wrong class will produce a Bounding Box with the wrong AR (e.g. frontal car, AR=1.0, left side car, AR >>1.0). We show our cost matrix in Fig. 7. The non-car class has label 1. Positive classes have the labels shown in Fig. 6 increased in one to account for the non-car class.

First, we train only one detector with the mean aspect ratio of each view class stored in the tree leaves. We set the detection threshold to an intersection over union (IoU) value 0.7, as established in the KITTI benchmark. Then, we use different AR estimation strategies. First we test the AR estimation algorithm introduced section 4 with different values of t_p (first tree to use in the estimation). In Fig. 8 we show that we get marginal improvements when selecting the best t_p.

Second, we compare different AR estimation strategies (we use here $t_p = 1$): use the output of the fixed size sliding window detector trained at 48×84 pixels (Fixed-Equal), modify the fixed aspect ratio window with the estimated class view (Fixed-Class-Mean) and, finally, our proposal (Estimated-AR). In Fig. 9 we confirm that a fixed aspect ratio detector, Fixed-Equal, produces the worst results. Results in the Moderate KITTI subset (the one used for ranking) improve significantly with the Fixed-Class-Mean strategy. Finally, we may improve even further the AP_{40} by using the proposed Estimated-AR strategy. We improve AP_{40} by 0.7%, 0.4% and 0.4%, respectively, in the Easy, Moderate and Hard settings. Since our bounding box estimation procedure is computationally cheap, it is a significant improvement. In fact, it only takes slightly more memory to store the K possible aspect ratios, $\mathbf{a}_m$, on each tree node. On the other hand, the added computation is negligible and consists on computing the aspect ratios weighted vector, $\mathbf{a}(\mathbf{x}) = \sum_{m=t_p}^{M} \beta_m \mathbf{a}_m(\mathbf{x})$.

To further analyze the performance of our procedure (Estimated-AR) with respect to the baseline (Fixed-Class-Mean), we have performed an additional experiment varying the IoU threshold (see table 1). The good behavior of our approach is more evident when we look for higher overlapping in the detection. With a minimum required IoU= 0.8, Estimated-AR, in Moderate subset, improves by 11,54% (from 51.1% to 57%) and with IoU= 0.9 the improvement is 105.26% (from 1.9% to 3.9%).

In an ablation analysis shown in Fig. 10 and Table 2, we evaluate different detector configurations and use synthetic data to enhance even further the performance of our detector. Data augmentation allows us to avoid

Table 1
AP_{40} for different IoU values on the KITTI train90/train10 car detection experiment.

	Algorithm / IoU	0.5	0.6	0.7	0.8	0.9
Easy	Fixed-Class-Mean	99.5%	99.2%	95.6%	52.2%	1.2%
	Estimated-AR	99.5%	99.2%	96.3%	60.8%	3.2%
Moderate	Fixed-Class-Mean	96.4 %	95.8%	88.9%	51.1%	1.9%
	Estimated-AR	96.4%	95.8%	89.3%	57.0%	3.9%
Hard	Fixed-Class-Mean	81.1%	78.3%	71.6%	39.9%	1.3%
	Estimated-AR	81.1%	78.4%	72.0%	44.8%	2.9%

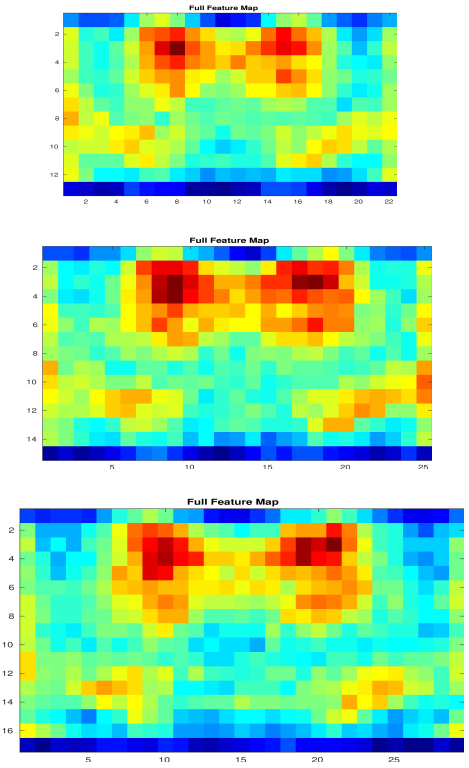

Fig. 5. Location of features selected by different car detectors. Top, middle and bottom images show respectively the LDCF features of the detector trained with 48×84, 56×98 and 64×112 pixel images.

overfitting and increase the Boosting classifier capacity with deeper decision trees using a finer quantization in car orientation. We use K=25 views whereas in our previous approaches [24, 36] we used $K = 20$. We depart from the baseline results in [24], "BAdaCost, K=20," obtained with T=1024 trees of depth D=8, K=20 views and $NA = 30000$ hard negatives (shrinkage 0.1 and $fracFtrs$=1/16 fraction of features explored on each tree training). Increasing the tree depth to $D = 10$ we have to regularize more (shrinkage 0.05 and $fracFtrs$=1/32, less features explored on each weak learner), but we improve AP_{40} by 2.9% (see "BAdaCost+I, K=20", Moderate results). If we now increase the number of views to $K = 25$, in some cases the AP_{40} decreases. This is because there are new classes with few data (see "BAdaCost+I", Moderate results). By including the bounding box adjustment procedure described in section 4, we get slightly better results (see "'BAdaCost+I+BB"). On the other hand, when we augment the data with synthetic samples we obtain a substantial improvement (see "BAdaCost+DA"), greater than that achieved with the bounding box adjustment, aided by the increased capacity of the classifier and the increased number of views. Combining the bounding box adjustment (BB) and the data augmentation (DA) we get the best single classifier detector (see "BAdaCost+DA+BB") which in the Moderate examples is reporting 6.1% AP_{40} improvement with respect to base line (83.2 to 89.3).

To further improve the AP of the detector we have to take into account that a car at different resolutions may not need the same features. Therefore, we train three detectors at different base sizes: 48×84, 56×98 and 64×112 pixels. We apply these classifiers to the image pyramid obtaining slightly better results in the hard examples (see "BAdaCost+DA+BB+MS", Hard). Finally, in the testing subset of KITTI[2] (see Fig. 11 and Table 3) the multiscale approach ("BadaCost+DA+BB+MS", Moderate) is significantly better than the single scale approach ("BAdaCost+DA+BB", Moderate) increasing in 2.8% the AP_{40}. The main reason is that it gets slightly better precision and better recall.

Since the KITTI dataset is not very large, the best CNN based detectors use a VGG network [37] pretrained with ImageNet [38] and finetuned with KITTI. Thus, they are not fully comparable to the Boosting methods trained only with KITTI. Moreover, there are

[2]See BdCost+DA+BB and BdCost+DA+BB+MS at the KITTI results page: http://www.cvlibs.net/datasets/kitti/eval_object.php

Fig. 6. The car classes we use in our experiments. Note the relation between classes, i.e. car view orientation, and bounding box aspect ratios.

Table 2

Car detection ablation study in KITTI's validation set (KITTI-train10). Key: stronger regularization (I), Bounding Box estimation (BB), Data Augmentation (DA), and multi-scale detection (MS), average precision with n curve points (AP_n).

Training / Testing	**Algorithm**	**Easy**		**Moderate**		**Hard**	
		AP_{11}	AP_{40}	AP_{11}	AP_{40}	AP_{11}	AP_{40}
train90 / train10	BAdacost, $K = 20$ [24]	84.8%	88.3%	82.1%	83.2%	66.9%	66.2%
	BAdacost+I, $K = 20$	87.3%	91.9%	86.4%	86.1%	69.1%	68.8%
	BAdacost+I	87.7%	92.5%	85.9%	85.9%	69.3%	68.9%
	BAdacost+BB	88.3%	93.1%	86.3%	86.2%	69.2%	69.0%
	BAdacost+DA	88.1%	93.0%	87.1%	88.7%	69.9%	71.5%
	BAdacost+DA+BB	**95.4%**	**96.3%**	88.0%	**89.3%**	70.5%	72.0%
	BAdacost+DA+BB+MS	93.6%	96.1%	**88.1%**	89.0%	**70.6%**	**73.5%**

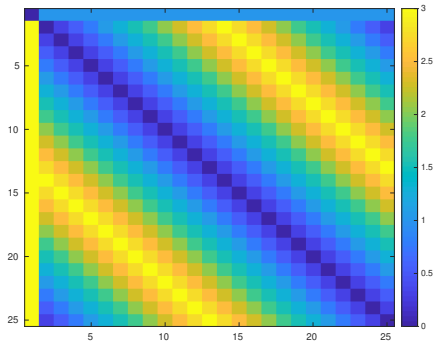

Fig. 7. The cost matrix, **C**, used in the KITTI car dataset experiments.

various of methods using LIDAR or the stereo image pair data. However, we are only interested in algorithms comparable with ours. In Fig. 11 and Table 3 we show the results of the top three detectors that use a single RGB image: TuSimple [23], DeepManta [20] and RRC [21]. The reason for their success is the use of convolutional features adapted for the problem at hand. However, the need for a powerful GPU could be determinant in some engineering applications with limited devices. In these cases, the Boosting algorithms presented in this paper represent a computationally efficient alternative.

We have also compared the execution time of RRC [21], one of leading deep learning-based car detectors, with our Boosting-based detector in a CPU. We have compiled the implementation of RRC[3] with OpenBLAS support[4], that adds thread-based parallelism to matrix operations. On average RRC takes 82 seconds to process an image, with 100% CPU usage (Intel Xeon E5620 2.4GHz). Such long processing time is caused by the large size of KITTI images (1242×375 pixels) and the fact that RRC doubles their resolution to detect small cars. On the other hand, our DA+BB+MS car detector takes on average 34 seconds per frame when doubling the input image size (i.e. one octave up the input size) and 6 to 12 seconds when using the standard size. The simpler BAdaCost detector, DA+BB, takes 17 seconds per image with double resolution and 4 seconds using the plain input images. Our implementation also uses OpenMP parallelism. However, our algorithm only uses 50% of the CPU.

[3] https://github.com/xiaohaoChen/rrc_detection
[4] http://www.openblas.net

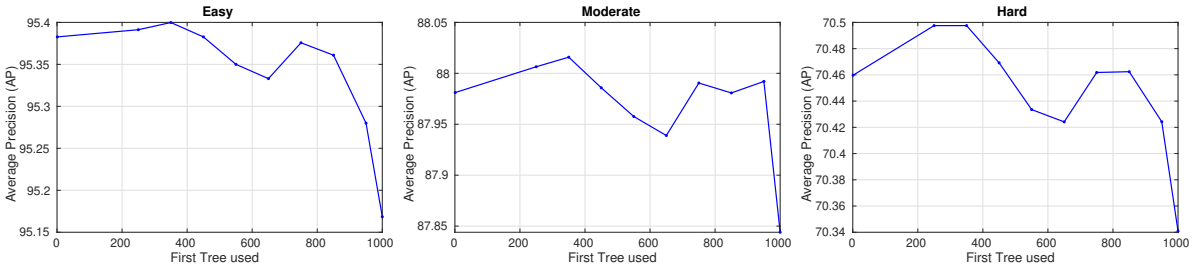

Fig. 8. Aspect ratio AP results depending on the first tree used in the estimation.

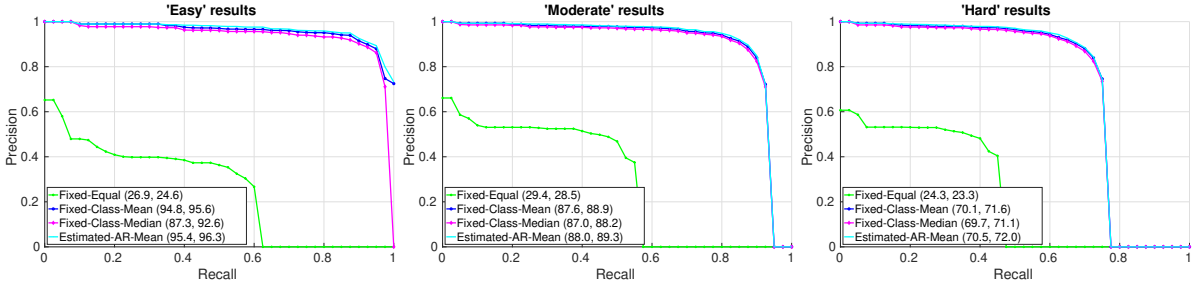

Fig. 9. Precision-recall curves and (AP_{11}, AP_{40}) values for different AR estimation strategies.

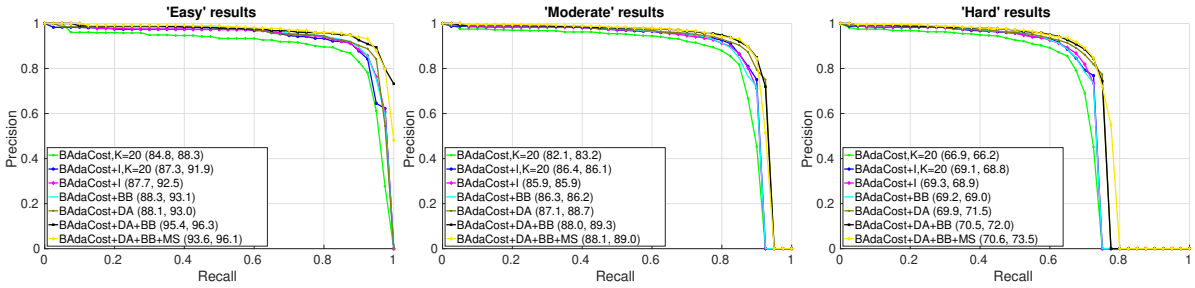

Fig. 10. Precision-recall curves and (AP_{11}, AP_{40}) values for the car detection ablation experiment.

Confirming the superior efficiency of Boosting-based detection approaches.

Finally, we compare our approach with other Boosting algorithms in KITTI testing (see Table 3): SubCat [16], Regionlets [17] and spLBP [39]. We use LDCF features [33], like SubCat, whereas Regionlets and spLBP benefit from a more discriminative set of features. However, we cannot fairly compare the Regionlets result with ours, since the KITTI experiment is not described in the original paper. Similarly, the spLBP paper does not describe the images used for training their test result. With all this in mind, with respect to Regionlets, our method has better precision up to a recall of 0.7 (see Fig. 11, Moderate). Compared with SubCat, our approach is better in the Easy subset and with recall values below 0.7.

6.2. *Cyclist detection experiments*

For completeness, in this section we evaluate the proposed technique in the cyclists detection KITTI dataset. As for cars, shrinkage is 0.05 and 1/32 the fraction of features to train each weak learner. There are only 1027 cyclists examples in the KITTI-train90 subset. Given the few available training images, the number of view classes is set to $K = 5$. This election provides training data for all views, i.e. 101, 298, 155, 161 and 312 training examples respectively for class 1, 2, 3, 4 and 5. After image flipping we get 2054 training samples (i.e. 181, 652, 318, 189, 714 respectively on each positive class.) We train a 48×72 pixels model, $AR = 1.5$. We start the pyramid one octave above the actual image size. We use the parameters obtained with the best AP: the best number of cost-sensitive trees

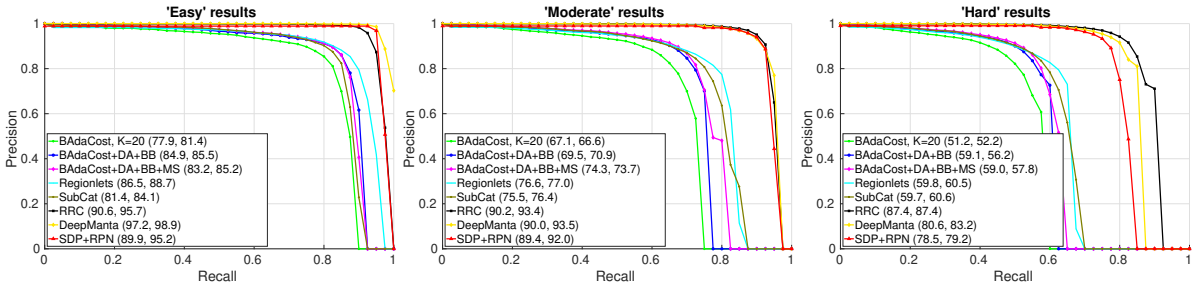

Fig. 11. Car detection comparison in KITTI-testing using the evaluation server. We report (AP_{11}, AP_{40}) for each method.

Table 3

Car detection comparison experiment in KITTI-testing evaluation server. Best BAdaCost results are shown in **bold** and overall best results in blue.

Training / Testing	**Algorithm**	**Easy**		**Moderate**		**Hard**	
		AP_{11}	AP_{40}	AP_{11}	AP_{40}	AP_{11}	AP_{40}
train90 / testing	BAdacost, $K = 20$ [24]	77.9%	81.4%	67.1%	66.6%	51.2%	52.2%
	BAdacost+DA+BB	**84.9%**	**85.5%**	69.5%	70.9%	**59.1%**	56.2%
	BAdacost+DA+BB+MS	83.2%	85.2%	**74.3%**	**73.7%**	59.0%	**57.8%**
training / testing (Boosting)	Subcat [16]	81.4%	84.1%	75.5%	76.4%	59.7%	60.6%
unknown / testing (Boosting)	Regionlets [17]	86.5%	88.7%	76.6%	77.0%	59.8%	60.5%
	spLBP [39]	80.2%	81.7%	77.4%	78.7%	60.6%	61.7%
ImageNet+training/ testing (CNNs)	TuSimple [23]	90.8%	95.1%	90.3%	94.5%	82.9%	86.5%
	DeepManta [20]	97.2%	98.9%	90.0%	93.5%	80.6%	83.2%
	RRC [21]	90.6%	95.7%	90.2%	93.4%	87.4%	87.4%

is T=2048 (4 rounds with 32, 128, 256 and T weak learners, respectively), tree depth is D=5, the number of negatives per round is N=2500 and the total amount of negatives is NA=10000. Finally, the cost matrix is defined to weigh up gross errors between view classes.

Again, we train one detector with the mean aspect ratio of each view class stored on the tree leaves. We use KITTI-train90 for training and KITTI-train10 for testing. We set the detection threshold to an IoU of 0.5, as established in the KITTI benchmark for cyclists, needing less precision in the bounding box than in the cars case. Thus, it is expected for the bounding box adjustment procedure to have less impact in this detection problem.

We show in Fig. 12 sample training data and the selected features. In Fig. 13 and Table 4 we display the result of this final set of experiments. Our detections deteriorate if we include the bounding box adjustment procedure, "BAdaCost+I+BB." The reason is that there is a lack of data to estimate the per class aspect ratio for each node in our decision trees. However, there is data to estimate the overall class mean aspect ratio. Hence the improvement with "BAdacost+I."

Fig. 12. Features selected by BAdaCost in the cyclists detection problem (top left) and sample training images (rest). Note that most of the features are around the rider and the wheels.

When we augment the data with synthetic samples, "BAdaCost+DA," we get better AP_{40} than "BAdaCost+BB" and "BAdaCost+I" in the Moderate and Hard cases. However, the new data worsen the estimation in the Easy group. By combining the bounding box adjustment (BB) and the data augmentation (DA) we get the best single classifier detector, "BAdaCost+DA+BB," which in the Moderate examples is reporting 2.7% AP_{40} improvement with respect to base line, from 80.4 to 83.1.

Finally, to improve AP_{40}, we train three detectors at different base sizes: 48×72, 56×84 and 64×96. We use these classifiers with an image pyramid obtaining and improvement greater than 6% with respect to the base line in all cases, "BAdaCost+DA+BB+MS."

As in the car case, the best CNN based detectors with cyclists that use only RGB images [21, 23, 40] use a pretrained VGG network on ImageNet and finetune it with the KITTI data. It makes these results no fully comparable to the Boosting ones trained only on the KITTI dataset. In the cyclists, because of the lack of training data, almost all methods take advantage of the LIDAR measures and/or the stereo pairs available. We use only RGB images data. In Table 5 we show the results of the three best published cyclists detectors on KITTI testing using only RGB information: RRC [21], TuSimple [23] and SubCNN [40].

In Table 5 we cite [17] the top Boosting result in the KITTI evaluation server. However, we cannot fairly compare this result with ours, since the KITTI experiment is not described in the paper.

In summary, our results in this section support the use of new techniques borrowed from the deep learning literature for training Boosting based detectors.

7. Conclusions

Detection algorithms have evolved over time by changing various components of their pipeline. Some of these improvements, however, have been exploited only in the context of modern deep neural nets. In this paper we have improved the performance of Boosting-based detectors by using data augmentation, refining the detection bounding box, and using multi-scale processing. This is a relevant result in the construction of detectors, given the computational efficiency of this family of algorithms.

In the experiments we show that Boosting-based approaches significantly improve their performance with respect to the multi-class baseline Boosting scheme [24] when using the new training techniques. We achieve an improvement of 7.1% and 5% respectively in the AP_{40} of the car and cyclist KITTI benchmarks. On the other hand, the our Boosting-based detectors are between 2X and 3.5X faster than their deep learning-based counterpart, at the expense of a significant decrease in precision.

In spite of the remarkable accuracy achieved by deep learning-based detectors, we believe there is still room for research in approaches based on more efficient traditional machine learning techniques and their application in different problems [41]. In the future we will investigate better box regression algorithms, the use of efficient and more discriminative image features and detection strategies better than the brute force sliding window.

Acknowledgements

This work was funded by the Spanish *Ministerio de Economía y Competitividad*, project TIN2016-75982-C2-2-R. José M. Buenaposada was also partially funded by the *Comunidad de Madrid* project RoboCity2030-DIH-CM (S2018/NMT-4331).

References

[1] V. Kazemi and J. Sullivan, One Millisecond Face Alignment with an Ensemble of Regression Trees, in: *CVPR*, 2014.

[2] V. Ferrari, M. Marin-Jimenez and A. Zisserman, Progressive Search Space Reduction for Human Pose Estimation, in: *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, 2008.

[3] X. Chen, K. Kundu, Z. Zhang, H. Ma, S. Fidler and R. Urtasun, Monocular 3D Object Detection for Autonomous Driving, in: *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, 2016, pp. 2147–2156.

[4] M.A. Molina-Cabello, R.M. Luque-Baena, E. López-Rubio and K. Thurnhofer-Hemsi, Vehicle Type Detection by Ensembles of Convolutional Neural Networks Operating on Super-resolved Images, *Integrated Computer-Aided Engineering* **25**(4) (2018), 321–333.

[5] M. Jaderberg, A. Vedaldi and A. Zisserman, Deep Features for Text Spotting, in: *Proc. European Conf. Computer Vision*, 2014.

[6] P.A. Viola and M.J. Jones, Fast and Robust Classification using Asymmetric AdaBoost and a Detector Cascade, in: *Conf. Neural Information Processing Systems*, 2001, pp. 1311–1318.

[7] N. Dalal and B. Triggs, Histograms of oriented gradients for human detection, in: *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, Vol. 1, 2005, pp. 886–893 vol. 1. ISSN 1063-6919. doi:10.1109/CVPR.2005.177.

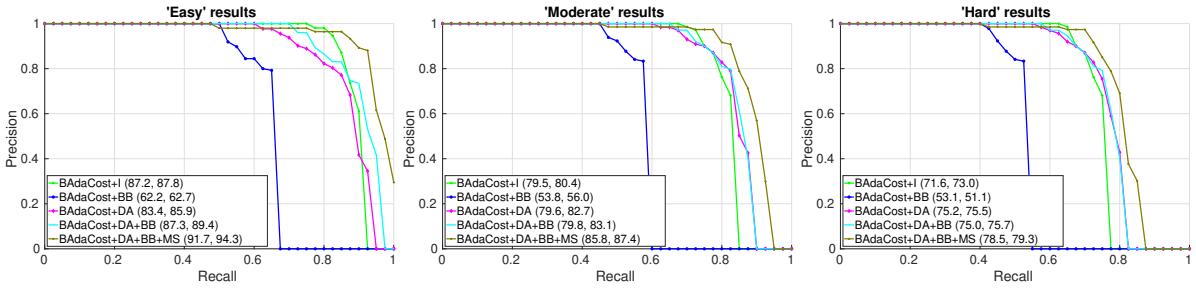

Fig. 13. Precision-recall curves and (AP_{11}, AP_{40}) for the cyclist detection ablation study.

Table 4

Cyclist detection ablation study. Key: Stronger regularization (I), Bounding Box estimation (BB), Data Augmentation (DA), and multi-scale detection (MS), average precision with n curve points (AP_n).

Training / Testing	**Algorithm**	**Easy**		**Moderate**		**Hard**	
		AP_{11}	AP_{40}	AP_{11}	AP_{40}	AP_{11}	AP_{40}
train90 / train10	BAdacost+I	87.2%	87.8%	79.5%	80.4%	71.6%	73.0%
	BAdacost+BB	62.2%	62.7%	53.8%	56.0%	53.1%	51.1%
	BAdacost+DA	83.4%	85.9%	79.6%	82.7%	75.2%	75.5%
	BAdacost+DA+BB	87.3%	89.4%	79.7%	83.1%	75.0%	75.7%
	BAdacost+DA+BB+MS	**91.7%**	**94.3%**	**85.8%**	**87.4%**	**78.5%**	**79.3%**

Table 5

Cyclist detection experiment comparison in KITTI-testing evaluation server. Best BAdaCost results are shown in **bold** and overall best results in blue.

Training / Testing	**Algorithm**	**Easy**		**Moderate**		**Hard**	
		AP_{11}	AP_{40}	AP_{11}	AP_{40}	AP_{11}	AP_{40}
train90 / testing	BAdacost+DA+BB	30.3%	26.9%	24.3%	20.0%	21.0%	16.8%
	BAdacost+DA+BB+MS	**36.9%**	**33.9%**	**29.3%**	**25.5%**	**25.4%**	**21.1%**
unkown / testing (Boosting)	Regionlets [17]	70.1%	71.1%	58.7%	58.5%	51.8%	50.8%
ImageNet+training/ testing (CNNs)	RRC [21]	85.0%	86.8%	76.5%	76.8%	65.5%	66.6%
	TuSimple [23]	81.4%	83.7%	74.3%	75.2%	64.9%	65.2%
	SubCNN [40]	77.8%	79.4%	70.8%	71.7%	62.7%	62.7%

[8] P. Dollar, R. Appel, S. Belongie and P. Perona, Fast Feature Pyramids for Object Detection, *IEEE Trans. Pattern Analysis and Machine Intelligence* **36**(8) (2014), 1532–1545.

[9] J. Gall, A. Yao, N. Razavi, L.V. Gool and V. Lempitsky, Hough Forests for Object Detection, Tracking, and Action Recognition, *IEEE Trans. Pattern Analysis and Machine Intelligence* **33**(11) (2011), 2188–2202. doi:10.1109/TPAMI.2011.70.

[10] S. Ren, K. He, R. Girshick and J. Sun, Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks, in: *Conf. Neural Information Processing Systems*, 2015.

[11] J. Redmon, S. Divvala, R. Girshick and A. Farhadi, You Only Look Once: Unified, Real-Time Object Detection, in: *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, 2016.

[12] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S.E. Reed, C. Fu and A.C. Berg, SSD: Single Shot MultiBox Detector, in: *Proc. European Conf. Computer Vision*, 2016, pp. 21–37.

[13] S. Zhang, R. Benenson and B. Schiele, Filtered channel features for pedestrian detection, in: *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, 2015.

[14] R. Benenson, M. Mathias, R. Timofte and L. Van Gool, Pedestrian detection at 100 frames per second, in: *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, 2012.

[15] M. Mathias, R. Benenson, M. Pedersoli and L. Van Gool, Face detection without bells and whistles, in: *Proc. European Conf. Computer Vision*, 2014.

[16] E. Ohn-Bar and M.M. Trivedi, Learning to Detect Vehicles by Clustering Appearance Patterns, *IEEE Trans. Intelligent Transportation Systems* **16**(5) (2015), 2511–2521.

[17] X. Wang, M. Yang, S. Zhu and Y. Lin, Regionlets for Generic Object Detection, *IEEE Trans. Pattern Analysis and Machine Intelligence* **37**(10) (2015), 2071–2084.

[18] M. Ahmadlou and H. Adeli, Enhanced Probabilistic Neural Network with Local Decision Circles: A Robust Classifier, *Integr. Comput.-Aided Eng.* **17**(3) (2010), 197–210–.

[19] M.H. Rafiei and H. Adeli, A New Neural Dynamic Classification Algorithm, *IEEE Transactions on Neural Networks and Learning Systems* **28**(12) (2017), 3074–3083.

[20] F. Chabot, M. Chaouch, J. Rabarisoa, C. Teulière and T. Chateau, Deep MANTA: A Coarse-to-fine Many-Task Network for joint 2D and 3D vehicle analysis from monocular image, *CoRR* **abs/1703.07570** (2017).

[21] J. Ren, X. Chen, J. Liu, W. Sun, J. Pang, Q. Yan, Y.-W. Tai and L. Xu, Accurate Single Stage Detector Using Recurrent Rolling Convolution, in: *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, 2017.

[22] R. Juranek, A. Herout, M. Dubska and P. Zemcik, Real-Time Pose Estimation Piggybacked on Object Detection, in: *Proc. Int'l Conf. Computer Vision*, 2015.

[23] F. Yang, W. Choi and Y. Lin, Exploit All the Layers: Fast and Accurate CNN Object Detector With Scale Dependent Pooling and Cascaded Rejection Classifiers, in: *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, 2016.

[24] A. Fernández-Baldera, J.M. Buenaposada and L. Baumela, BAdaCost: Multi-class Boosting with Costs, *Pattern Recognition* **79** (2018), 467–479.

[25] E. Ohn-Bar and M.M. Trivedi, To Boost or Not to Boost? On the Limits of Boosted Trees for Object Detection, in: *Proc. Int'l Conf. Pattern Recognition*, 2016.

[26] J. Friedman, T. Hastie and R. Tibshirani, Additive Logistic Regression: a Statistical View of Boosting, *The Annals of Statistics* **28**(2) (2000), 337–407.

[27] A. Fernández-Baldera, J.M. Buenaposada and L. Baumela, Multi-class Boosting for Imbalanced Data, in: *Proc. of Iberian Conf. Pattern Recognition and Image Analysis*, 2015, pp. 57–64.

[28] J. Zhu, H. Zou, S. Rosset and T. Hastie, Multi-class AdaBoost, *Statistics and Its Interface* **2** (2009), 349–360.

[29] C. Zhang and P.A. Viola, Multiple-Instance Pruning For Learning Efficient Cascade Detectors, in: *Conf. Neural Information Processing Systems*, 2007, pp. 1681–1688.

[30] B. Yang, J. Yan, Z. Lei and S.Z. Li, Aggregate channel features for multi-view face detection, in: *IEEE Int'l J. Conference on Biometrics*, 2014, pp. 1–8.

[31] P. Hu and D. Ramanan, Finding Tiny Faces, in: *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, 2017.

[32] P. Dollar, S. Belongie and P. Perona, The Fastest Pedestrian Detector in the West, in: *Proc. British Machine Vision Conf.*, 2010, pp. 68.1–11. ISBN 1-901725-40-5.

[33] J.H.H. W. Nam P. Dollar, Local Decorrelation For Improved Pedestrian Detection, in: *Conf. Neural Information Processing Systems*, 2014.

[34] A. Geiger, P. Lenz and R. Urtasun, Are we ready for Autonomous Driving? The KITTI Vision Benchmark Suite, in: *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, 2012.

[35] A. Simonelli, S.R.R. Bulò, L. Porzi, M. López-Antequera and P. Kontschieder, Disentangling Monocular 3D Object Detection, 2019.

[36] J.M. Buenaposada and L. Baumela, Boosting Object Detection in Cyberphysical Systems, in: *Understanding the Brain Function and Emotions, LNCS volume 11486)*, Springer International Publishing, Cham, 2019, pp. 309–318.

[37] K. Simonyan and A. Zisserman, Very Deep Convolutional Networks for Large-Scale Image Recognition, in: *International Conference on Learning Representations*, 2015.

[38] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A.C. Berg and L. Fei-Fei, ImageNet Large Scale Visual Recognition Challenge, *International Journal of Computer Vision (IJCV)* **115**(3) (2015), 211–252. doi:10.1007/s11263-015-0816-y.

[39] Q. Hu, S. Paisitkriangkrai, C. Shen, A. van den Hengel and F. Porikli, Fast Detection of Multiple Objects in Traffic Scenes With a Common Detection Framework, *IEEE Transactions on Intelligent Transportation Systems* **17**(4) (2016), 1002–1014.

[40] Y. Xiang, W. Choi, Y. Lin and S. Savarese, Subcategory-Aware Convolutional Neural Networks for Object Proposals and Detection, in: *2017 IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2017, pp. 924–933.

[41] S. Roegiers, G. Allebosch, P. Veelaert and W. Philips, Human action recognition using hierarchic body related occupancy maps, *Integrated Computer-Aided Engineering* **26**(3) (2019), 223–241.

[42] D. Serpanos, The Cyber-Physical Systems Revolution, *Computer* **51**(3) (2018), 70–73.

[43] W. Wolf, Cyber-physical Systems, *Computer* **42**(3) (2009), 88–89.

[44] E. Diller and M. Metin Sitti, Micro-Scale Mobile Robotics, *Foundations and Trends in Robotics* **2**(3) (2013), 143–259.

[45] L. Liu, W. Ouyang, X. Wang, P.W. Fieguth, J. Chen, X. Liu and M. Pietikäinen, Deep Learning for Generic Object Detection: A Survey, *CoRR* **abs/1809.02165** (2018). http://arxiv.org/abs/1809.02165.

[46] H. Guo and H.L. Viktor, Learning from Imbalanced Data Sets with Boosting and Data Generation: The DataBoost-IM Approach, *SIGKDD Explorations Newsletter* **6**(1) (2004), 30–39.

[47] R. Kumar, A. Banerjee, B.C. Vemuri and H. Pfister, Trainable Convolution Filters and Their Application to Face Recognition, *IEEE Trans. Pattern Analysis and Machine Intelligence* **34**(7) (2012), 1423–1436.

[48] H. Han, C. Otto, X. Liu and A.K. Jain, Demographic Estimation from Face Images: Human vs. Machine Performance, *IEEE Trans. Pattern Analysis and Machine Intelligence* **37**(6) (2015), 1148–1161.

[49] W. L., G. Hua, G. Sukthankar, J. Xue, Z. Niu and N. Zheng, Video Object Discovery and Co-Segmentation with Extremely Weak Supervision, *IEEE Trans. Pattern Analysis and Machine Intelligence* (2017).

[50] J. Shotton, A. Blake and R. Cipolla, Multiscale Categorical Object Recognition Using Contour Fragments, *IEEE Trans. Pattern Analysis and Machine Intelligence* **30**(7) (2008), 1270–1281.

[51] S. Mahamud, M. Hebert and J. Shi, Object recognition using boosted discriminants, in: *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, Vol. 1, 2001, pp. I–551-I-558.

[52] T.G. Dietterich and G. Bakiri, Solving Multiclass Learning Problems via Error-Correcting Output Codes, *J. of Artificial Intelligence Research* (1995), 263–286.

[53] Y. Freund and R.E. Schapire, Experiments with a New Boosting Algorithm, in: *Proc. Int'l Conf. on Machine Learning*, 1996, pp. 148–156.

[54] Y. Freund and R.E. Schapire, A decision theoretic generalization of on-line learning and an application to boosting, *J. of Computer and System Sciences* **55** (1997), 199–139.

[55] R.E. Schapire, Using Output Codes to Boost Multiclass Learning Problems, in: *Proc. Int'l Conf. on Machine Learning*, 1997, pp. 313–321.

[56] D.B. O'Brien, M.R. Gupta and R.M. Gray, Cost-sensitive Multi-class classification from probability estimates, in: *Proc. Int'l Conf. on Machine Learning*, 2008, pp. 712–719.

[57] K.M. Ting and Z. Zheng, Boosting cost-sensitive trees, in: *Proc. International Conference on Discovery Science*, 1998, pp. 244–255.

[58] P. Domingos, MetaCost: A General Method for Making Classifiers Cost-Sensitive, in: *Proc. Int'l Conf. on Knowledge Discovery and Data Mining*, 1999, pp. 155–164.

[59] W. Fan, S.J. Stolfo, J. Zhang and P.K. Chan, AdaCost: Misclassification Cost-sensitive Boosting, in: *Proc. Int'l Conf. on Machine Learning*, 1999, pp. 97–105.

[60] V. Guruswami and A. Sahai, Multiclass learning, boosting and error correcting codes, in: *Proc. Annual Conference on Learning Theory*, 1999, pp. 145–155.

[61] R.E. Schapire and Y. Singer, Improved Boosting Algorithms Using Confidence-rated Predictions, *Machine Learning* **37** (1999), 297–336.

[62] E.L. Allwein, R.E. Schapire and Y. Singer, Reducing Multiclass to Binary: A Unifying Approach for Margin Classifiers, *J. of Machine Learning Research* **1** (2000), 113–141.

[63] K.M. Ting, A Comparative Study of Cost-Sensitive Boosting Algorithms, in: *Proc. Int'l Conf. on Machine Learning*, 2000, pp. 983–990.

[64] C. Elkan, The Foundations of Cost-Sensitive Learning, in: *Proc. Int'l Joint Conf. on Artificial Intelligence*, 2001, pp. 973–978.

[65] N. Abe, B. Zadrozny and J. Langford, An iterative method for multi-class cost-sensitive learning, in: *Proc. Int'l Conf. on Knowledge Discovery and Data Mining*, 2004, pp. 3–11.

[66] Y. Lee, Y. Lin and G. Wahba, Multicategory Support Vector Machines: theory and application to the classification of microarray data and satellite radiance data, *J. American Statistical Association* **99** (2004), 67–81.

[67] A. Torralba, K.P. Murphy and W.T. Freeman, Sharing Features: Efficient Boosting Procedures for Multiclass Object Detection, in: *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, 2004, pp. 762–769.

[68] P. Viola and M.J. Jones, Robust Real-Time Face Detection, *Int'l J.Computer Vision* **57**(2) (2004), 137–154.

[69] L. Bourdev and J. Brandt, Robust object detection via soft cascade, in: *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, Vol. 2, 2005, pp. 236–243.

[70] Y. Sun, A.K.C. Wong and Y. Wang, Parameter Inference of Cost-Sensitive Boosting Algorithms, in: *Proc. Int'l Conf. on Machine Learning and Data Mining*, 2005, pp. 21–30.

[71] J. Demsar, Statistical Comparisons of Classifiers over Multiple Data Sets, *J. of Machine Learning Research* **7** (2006), 1–30.

[72] Y. Sun, M.S. Kamel and Y. Wang, Boosting for Learning Multiple Classes with Imbalanced Class Distribution, in: *Proc. Int'l Conference on Data Mining*, 2006, pp. 592–602.

[73] H. Masnadi-Shirazi and N. Vasconcelos, Asymmetric Boosting, in: *Proc. Int'l Conf. on Machine Learning*, 2007, pp. 609–619.

[74] Y. Sun, M.S. Kamel, A.K.C. Wong and Y. Wang, Cost-sensitive boosting for classification of imbalanced data, *Pattern Recognition* **40**(12) (2007), 3358–3378.

[75] A.C. Lozano and N. Abe, Multi-class Cost-sensitive Boosting with p-norm Loss Functions, in: *Proc. Int'l Conf. on Knowledge Discovery and Data Mining*, 2008, pp. 506–514.

[76] H. Zou, J. Zhu and T. Hastie, New multicategory boosting algorithms based on multicategory Fisher-consistent losses, *Annals of Applied Statistics* **2** (2008), 1290–1306.

[77] H. He and E.A. Garcia, Learning from Imbalanced Data, *IEEE Trans. on Konwledge and Data Engineering* **21**(9) (2009), 1263–1284.

[78] F. Xia, Y. Yang, L. Zhou, F. Li, M. Cai and D.D. Zeng, A Closed-form Reduction of Multi-class Cost-sensitive Learning to Weighted Multi-class Learning, *Pattern Recognition* **42**(7) (2009), 1572–1581.

[79] Z. Zhou and X. Liu, On Multi-Class Cost-Sensitive Learning, *Computational Intelligence* **26**(3) (2010), 232–257.

[80] M. Everingham, L. Van Gool, C.K.I. Williams, J. Winn and A. Zisserman, The Pascal Visual Object Classes (VOC) Challenge, *Int'l J.Computer Vision* **88**(2) (2010), 303–338.

[81] V. Jain and E. Learned-Miller, FDDB: A Benchmark for Face Detection in Unconstrained Settings, Technical Report, UM-CS-2010-009, University of Massachusetts, Amherst, 2010.

[82] M.K. östinger, P. Wohlhart, P.M. Roth and H. Bischof, Annotated Facial Landmarks in the Wild: A large-scale, real-world database for facial landmark localization, in: *Proc. Int'l Conf. Computer Vision Workshops*, 2011, pp. 2144–2151.

[83] H. Masnadi-Shirazi and N. Vasconcelos, Cost-Sensitive Boosting, *IEEE Trans. Pattern Analysis and Machine Intelligence* **33**(2) (2011), 294–309.

[84] M.J. Saberian and N. Vasconcelos, Multiclass Boosting: Theory and Algorithms, in: *Conf. Neural Information Processing Systems*, 2011.

[85] I. Landesa-Vázquez and J.L. Alba-Castro, Shedding light on the asymmetric learning capability of AdaBoost, *Pattern Recognition Letters* **33**(3) (2012), 247–255.

[86] X. Liu and Z. Zhou, Towards Cost-Sensitive Learning for Real-World Applications, in: *Pacific-Asia Conf. Knowledge Discovery and Data Mining Workshops*, Vol. 7104, 2012, pp. 494–505.

[87] X. Zhu and D. Ramanan, Face detection, pose estimation, and landmark localization in the wild, in: *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, 2012, pp. 2879–2886.

[88] R. Benenson, M. Mathias, T. Tuytelaars and L. Van Gool, Seeking the strongest rigid detector, in: *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, 2013.

[89] M. Mathias, R. Benenson, R. Timofte and L. Van Gool, Handling Occlusions with Franken-classifiers, in: *Proc. Int'l Conf. Computer Vision*, 2013.

[90] J. Wang, Boosting the generalized margin in cost-sensitive multiclass classification, *J. Computational and Graphical Statistics* **22**(1) (2013), 178–192.

[91] A. Fernández-Baldera and L. Baumela, Multi-class boosting with asymmetric weak-learners, *Pattern Recognition* **47**(5) (2014), 2080–2090.

[92] R. Benenson, M. Omran, J. Hosang, and B. Schiele, Ten years of pedestrian detection, what have we learned?, in: *Proc. European Conf. Computer Vision, CVRSUAD workshop*, 2014.

[93] S.A.P. Parambath, N. Usunier and Y. Grandvalet, Optimizing F-measures by Cost-sensitive Classification, in: *Conf. Neural Information Processing Systems*, 2014, pp. 2123–2131.

[94] T. Wu, B. Li and S.C. Zhu, Learning And-Or Model to Represent Context and Occlusion for Car Detection and Viewpoint Estimation, *IEEE Trans. Pattern Analysis and Machine Intelligence* **38**(9) (2016), 1829–1843.

[95] B. Yang, J. Yan, Z. Lei and S.Z. Li, Convolutional Channel Features, in: *Proc. Int'l Conf. Computer Vision*, 2015, pp. 82–90.

[96] N. Nikolaou, N.U. Edakunni, M. Kull, P.A. Flach and G. Brown, Cost-sensitive boosting algorithms: Do we really need them?, *Machine Learning* **104**(2–3) (2016), 359–384.

[97] Z. Cai, Q. Fan, R.S. Feris and N. Vasconcelos, *A Unified Multi-scale Deep Convolutional Neural Network for Fast Object Detection*, in: *ECCV*, Springer, 2016, pp. 354–370.

[98] M. Oeljeklaus, F. Hoffmann and T. Bertram, A Fast Multi-Task CNN for Spatial Understanding of Traffic Scenes, in: *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, 2018, pp. 2825–2830.