Random Forests of Local Experts for Pedestrian Detection

Javier Marín[1], David Vázquez[1], Antonio M. López[1], Jaume Amores[1], Bastian Leibe[2]
[1] Computer Vision Center, Universitat Autònoma de Barcelona
[2] UMIC Research Centre, RWTH Aachen University

Abstract

Pedestrian detection is one of the most challenging tasks in computer vision, and has received a lot of attention in the last years. Recently, some authors have shown the advantages of using combinations of part/patch-based detectors in order to cope with the large variability of poses and the existence of partial occlusions. In this paper, we propose a pedestrian detection method that efficiently combines multiple local experts by means of a Random Forest ensemble. The proposed method works with rich block-based representations such as HOG and LBP, in such a way that the same features are reused by the multiple local experts, so that no extra computational cost is needed with respect to a holistic method. Furthermore, we demonstrate how to integrate the proposed approach with a cascaded architecture in order to achieve not only high accuracy but also an acceptable efficiency. In particular, the resulting detector operates at five frames per second using a laptop machine. We tested the proposed method with well-known challenging datasets such as Caltech, ETH, Daimler, and INRIA. The method proposed in this work consistently ranks among the top performers in all the datasets, being either the best method or having a small difference with the best one.

1. Introduction

Pedestrian detection is an extremely challenging task due to the large intra-class variability caused by different articulated poses and clothing, cluttered backgrounds, abundant partial occlusions and frequent changes in illumination. The seminal work of Dalal and Triggs [5] showed the importance of using rich block-based descriptors such as the Histograms of Oriented Gradients (HOG) representation, which provides both robustness and distinctiveness. Building upon this work, other authors have proposed additional features that enrich the visual representation, including the use of color through self-similarity features (CSS) [22], texture through block-based Local Binary Patterns (LBP) [23], and the design of efficient gradient-based features via integral channels [8, 7].

All of these approaches are holistic, in the sense that the whole pedestrian is described by a single feature vector and is classified at once. Recently, some authors have proposed successful methods for combining local detectors [13, 4, 25] and integrating the evidence from multiple local patches [14, 20, 16, 18, 11]. This type of approaches provides more flexibility in the spatial configuration of the different parts of the object, which leads to higher adaptability to the different poses of the pedestrian. Furthermore, it provides higher robustness against partial occlusions and atypical part appearances [14]. The most promising local part-based approach, proposed by Felzenszwalb et al. [13] has shown state-of-the-art results in several challenging datasets, being consistently ranked among the top performers.

Regarding the classification method, most approaches have made use of linear SVM classifiers [6, 23, 13, 22], which combine both the strength of the SVM machinery and the efficiency of a linear computation. AdaBoost is also a popular classifier for pedestrian detection, typically used in the presence of large numbers of features [24, 8, 7], or for speeding up the detection through cascaded layers of Boosting [21, 7, 27, 1].

Recently, Random Forest ensembles [14, 20, 16] have been proposed as an alternative type of ensemble classifier for pedestrian detection. However, traditionally based on simple pixel comparisons, their detection accuracy has remained moderate. In this paper, we propose a novel pedestrian detection approach that combines the flexibility of a part-based model with the fast execution time of a Random Forest classifier. In this proposed combination, the role of the part evaluations is taken over by local expert evaluations at the nodes of the decision tree. As an image window proceeds down the tree, a variable configuration of local experts is evaluated on its content, depending on the outcome of previous evaluations. Thus, our proposed approach can flexibly adapt to different pedestrian viewpoints and body poses. At the same time, by using an appropriate bootstrapping procedure, different trees of the forest cover different spatial configurations of parts in the object. This in turn leads to the implicit emergence of a part-based model through the proposed Random Forest of local experts. In addition to this, the decision tree structure ensures that only a small number of local experts are evaluated on each detection window, resulting in fast execution. The proposed detection system was evaluated with a variety of well-known pedestrian datasets such as Caltech [9], Daimler [10], ETH [12] and INRIA [5], where it consistently ranks among the top performers. This is on par with the most successful part-based detection system [13], while our method presents far less design complexity and higher computational efficiency.

The rest of the paper is organized as follows. Section 2 describes key concepts of the standard Random Forest classifier, section 3 introduces the proposed method, section 4 describes the state-of-the-art approaches related to ours, section 5 provides results and section 6 summarizes the work and discusses its contributions.

2. Standard weak learner model

Before discussing the classifier proposed in our work, let us first introduce the basic concepts and notation of the Random Forest ensemble [3]. For lack of space we restrict the explanation to only the standard weak learner model, and we refer to [3] for an in-depth description of the Random Forests (RF) classifier.

Given a tree of the forest, we follow the notation in [3] and denote as S_j the set of samples received by the j-th internal or split node of this tree. We denote as $h(\vec{v};\theta_j) \in \{0,1\}$ the split function associated with this node, where $\vec{v}$ is a feature vector and θ_j is the set of parameters defining the split function. The split function acts as a weak classifier that is part of the ensemble defined by the whole tree. At training time, the j-th node receives a subset of samples S_j, and based on this data the classifier $h(\vec{v};\theta_j)$ is trained. This is done by finding the optimal parameters θ_j for this classifier. At test time, the j-th node receives the feature vector $\vec{v}$, and this vector is passed to either the left or the right child depending on the output of $h(\vec{v};\theta_j) \in \{0,1\}$.

Criminisi et al. [3] define a general framework for defining the split function $h(\vec{v};\theta_j)$. In this framework, the set of parameters θ_j is defined as $\theta_j = (\phi,\psi,\tau)$, where the parameter ϕ is defined as a feature selection function that allows to disregard the noisy features in $\vec{v}$, the parameter ψ defines a geometric transformation that maps the data to a space where it is separable, and the parameter τ is a threshold that permits to classify the points.

In order to clarify the ideas, let us consider a common instantiation of this general framework. The feature selector ϕ is defined as the function $\phi(\vec{v}) = \vec{u}$ where $\vec{u} \in \mathbb{R}^s$ contains a subset of components of $\vec{v} \in \mathbb{R}^d$, $s < d$. The geometric transformation is parameterized by a vector $\vec{\psi} \in \mathbb{R}^s$ defining a linear projection $\phi(\vec{v}) \cdot \vec{\psi}$ over the selected features [1]. Finally, the split function $h(\vec{v};\theta)$ is defined as $[\phi(\vec{v})\cdot\vec{\psi} < \tau]$, where $[]$ is the indicator function. As a result, the classification is performed by first selecting some of the components of $\vec{v}$, then projecting the resulting vector, and then applying the threshold τ.

Let $\mathcal{T}$ be the search space where the parameters θ_j live. The optimal parameters θ_j are estimated as follows:

1. Randomly sample a small subset $\mathcal{T}_j \subset \mathcal{T}$.
2. For each $\theta \in \mathcal{T}_j$ do:
   (a) Split the set S_j into two subsets:
   $$\mathcal{S}_j^L = \{\vec{v} \in \mathcal{S}_j : h(\vec{v};\theta) = 0\}$$
   $$\mathcal{S}_j^R = \{\vec{v} \in \mathcal{S}_j : h(\vec{v};\theta) = 1\}$$
   (b) Evaluate the goodness of the previous partition using some measure of purity such as the information gain:
   $$I(\theta) = H(\mathcal{S}_j) - \sum_{child\in\{L,R\}} \frac{|\mathcal{S}_j^{child}|}{|\mathcal{S}_j|} H(\mathcal{S}_j^{child}) \tag{1}$$
   where $H(\mathcal{S})$ is the entropy:
   $$H(\mathcal{S}) = -\sum_{c\in\mathcal{C}} p(c)\log(p(c)).$$
3. Define the parameters for node j as:
   $$\theta_j = \arg\max_{\theta\in\mathcal{T}_j} I(\theta)$$

3. Proposed method

In this work we define a novel ensemble of local experts based on an averaged combination of random decision trees. We first describe the main differences with respect to the standard RF framework by using generic pattern recognition concepts. Afterwards we will introduce the concepts specific to pedestrian detection, and introduce our ensemble of local experts.

3.1. Weak learner model

The main difference with respect to the standard framework is that in each node the optimization of the parameters θ is not only based on a maximization of a purity measure (Eq. 1), but also on a maximum-margin optimizer which minimizes the classification error over the samples of the

[1] Using some abuse of notation, we write $\vec{\psi}$ as a vector in order to express the linear projection $\phi(\vec{v}) \cdot \vec{\psi}$. However, in the general framework of Criminisi et al. [3] the parameter ψ defines a generic geometric transformation, and thus should be expressed as a function in the general case.

node $\mathcal{S}_j$. In particular, this is done by optimizing the linear transformation $\vec{\psi}$ based on the linear SVM learning algorithm. Later on we will see that the joint use of this learner together with and an appropriate feature selector $\phi(\vec{v})$ provides the desired ensemble of local experts.

Keeping the discussion still under generic pattern recognition terms, the optimization process for each node j is composed of the following steps:

1. Randomly generate a subset $\{\phi_1, \ldots, \phi_K\}$ of K feature selectors $\phi_k(\vec{v})$. The generation of these K feature selectors is explained in section 3.2.
2. For $k = 1, \ldots, K$ do:
   (a) Let $\mathcal{S}_j^{\phi_k}$ be the transformed set of samples: $\mathcal{S}_j^{\phi_k} = \{\phi_k(\vec{v}) : \vec{v} \in \mathcal{S}_j\}$.

   (b) Obtain a discriminant linear transformation $\vec{\psi}_k$ by learning a linear SVM classifier over the transformed samples $\mathcal{S}_j^{\phi_k}$.

   (c) Find the threshold τ_k that maximizes the purity (Eq. 1) of the following partition:
   $$\begin{aligned} \mathcal{S}_j^L &= \{\vec{v} \in \mathcal{S}_j : \vec{\psi}_k^T \cdot \phi_k(\vec{v}) \leq \tau_k\} \\ \mathcal{S}_j^R &= \{\vec{v} \in \mathcal{S}_j : \vec{\psi}_k^T \cdot \phi_k(\vec{v}) > \tau_k\} \end{aligned}$$
   Note that the projected values $\vec{\psi}_k^T \cdot \phi_k(\vec{v})$ are classification scores provided by the previously learned linear SVM classifier.

   (d) Let $P_k = I(\phi_k, \vec{\psi}_k, \tau_k)$ be the maximum purity value obtained in the previous step.
3. Let $k^* = \arg\max_{k=1,\ldots,K} P_k$. Define the split function for node j as:
$$h(\vec{v}; \theta_j) = [\vec{\psi}_{k^*} \cdot \phi_{k^*}(\vec{v}) \leq \tau_{k^*}] \ . \tag{2}$$

The most important difference between the proposed weak learner model and the standard one lies in the optimization of the linear transformation in step 2.b. In our case, this is carried out by a discriminant optimizer such as the linear SVM learner. This learner obtains a hyperplane that optimally separates the set of training samples at each node. In contrast, in the standard weak learner model the optimization consists of randomly generating a few transformations and then evaluating each transformation together with the rest of parameters (the feature selector ϕ and threshold τ) in order to obtain the combination that maximizes the purity of the resulting partition. While the latter approach also provides a discriminant classification of the samples, there is no guarantee that the resulting hyperplane provides an optimal maximum-margin discrimination. Furthermore, the use of a discriminant classifier such as the linear SVM, together with an appropriate definition of the feature selector ϕ (see section 3.2) allows us to train our ensemble of local experts inside the RF framework.

3.2. Feature selector

We define our ensemble of local experts through the definition of an appropriate feature selector $\phi(\vec{v})$. Fig. 1(a) shows an illustration of the idea: given an image window, a block based descriptor $\vec{v}$ such as HOG is extracted by partitioning the window into $N \times M$ blocks [2]. Given this block-based descriptor $\vec{v}$, each feature selector ϕ_k defines a rectangular region formed by contiguous blocks.

In particular, the k-th feature selector ϕ_k is generated by randomly selecting the coordinates (i, j) of the top-left block, and randomly generating the width W and height H of the rectangular area, where $1 \leq W \leq L$ and $1 \leq H \leq L$, with L the predefined maximum size.

Given the previous definition of the feature selector ϕ_k, the k-th local expert is defined as $E_k(\vec{v}) = \vec{\psi}_k^T \cdot \phi_k(\vec{v})$. As explained in section 3.1, the transformation $\vec{\psi}_k$ is learned by using a discriminant learner such as linear SVM, using the transformed samples $\mathcal{S}_j^{\phi_k}$ as training set. This is equivalent to extracting a local block-based feature vector from the same rectangular area across the different image windows introduced into the node, and feeding them to a learner that obtains a model of this part of the window. In our case, however, an explicit extraction of local descriptors is not necessary, making the approach computationally efficient.

Note that there is a very large number of possible feature selectors ϕ_k that can be defined over a typical block-based descriptor such as HOG or HOG-LBP, and not all of them provide the same discriminatory power. Using the weak learner defined in Section 3.1, the j-th node randomly generates a fixed number K of feature selectors ϕ_k, learns the corresponding local experts E_k and selects the most discriminant one according to the given data. The selected local expert E_k also complements, in a classification sense, the ones selected by the other nodes in the same branch of the tree. This is due to the fact that the data samples received by the node j depend on the classification provided by its ancestors. As a result, each tree of the forest provides an ensemble of local experts which are both discriminant and complementary.

3.3. Definition of other RF components

The rest of the RF components are defined in a standard way [3]. This comprises the type of randomness, and the aggregation rule used for obtaining the final output of the forest. Regarding the type of randomness, we do not use bagging in this work, i.e., each tree of the forest receives the whole training set. This choice is recommended in the analysis of Criminisi et al. [3] and gave us slightly better results in preliminary tests.

Regarding the output of the forest, let $p_t(c|\vec{v})$ be the

[2] Note that neighbor blocks usually overlap each other, although this is not illustrated in the Fig. 1(a).

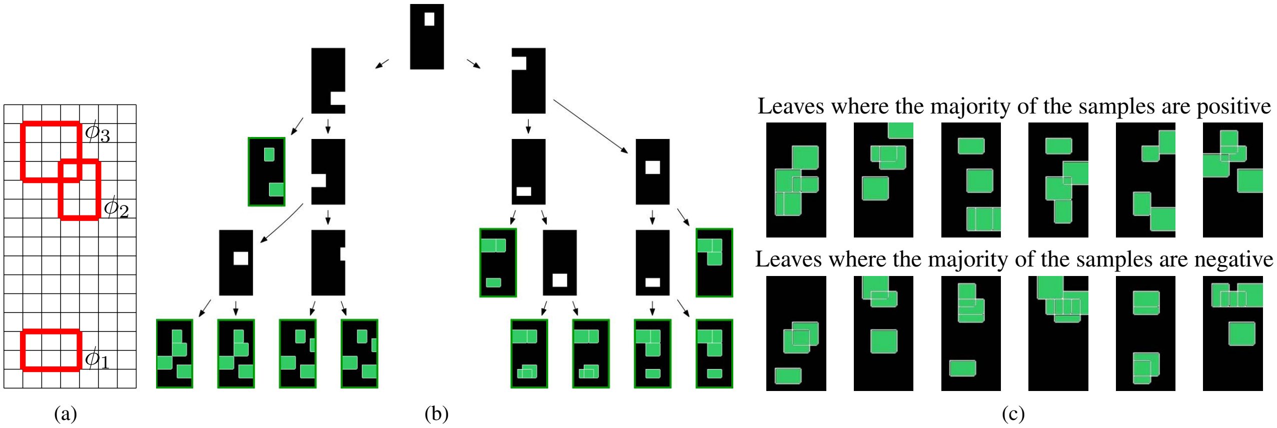

ϕ_3
ϕ_2
ϕ_1
Leaves where the majority of the samples are positive
Leaves where the majority of the samples are negative
(a)
(b)
(c)

Figure 1. (a) Illustration of our feature selector, (b) tree generated by our method (c) leaves of the forest. See main text.

probability that the window $\vec{v}$ belongs to class c, computed by the t-th tree of the forest. This probability is obtained during the training stage. Every leaf stores the class distribution of the training samples that reach it, and then each leaf probability is set according to this distribution. Given this, we use the average as aggregation rule in order to compute the probability for the whole forest: $p_{\mathcal{F}}(c|\vec{v}) = \frac{1}{T}\sum_{t=1}^{T} p_t(c|\vec{v})$, where T represents the number of trees in the RF $\mathcal{F}$.

Fig. 1(b) shows an example of tree generated with the proposed method, where for each internal node we show the patch of the window selected, and for each leaf we show, overlapped, all the patches selected from the root to this leaf. In Fig. 1(c) we show the typical combination of patches selected for positive and negative samples.

3.4. Bootstrapping procedure

We use bootstrapping at training time in order to select a subset of negative windows from the large pool of possible negatives. For this purpose, we propose to use an efficient procedure that consists of the following steps:

1. Set the initial training set as $\mathcal{S} = \mathcal{P} \cup \mathcal{N}$, where $\mathcal{P}$ is the set of cropped pedestrians, and $\mathcal{N}$ is an initial set of negative windows that are randomly sampled.
2. Set the initial forest as $\mathcal{F} = \emptyset$
3. For $i = 1, \ldots, N_{boot}$ do:
   (a) Train M new trees using the training set $\mathcal{S}$. Add the trees to the current forest $\mathcal{F}$.
   (b) Use the current forest $\mathcal{F}$ for detecting false positives in the training images. Consider these false positives as negative samples and add them to the training set $\mathcal{S}$.
   (c) Use the new training set $\mathcal{S}$ for updating the leaf probabilities $p(c|\vec{v})$ for all the trees in $\mathcal{F}$.

Our strategy, when compared with [20], allows to reduce the number of hard negatives obtained at each iteration. This is mainly due to the fact that at each iteration more trees are responsible for classifying, and that all the probabilities $p(c|\vec{v})$ stored at the leaf nodes are updated using the entire training set (which slightly increments their discriminative ability). Moreover, it is worth mentioning that the training time is reduced thanks to the smaller number of negative samples introduced at each iteration.

3.5. Soft Cascade

In order to speed up the detection of objects, we propose to use a Soft Cascade (SC) architecture [2]. Let T be the total number of trees in the forest, M be the number of trees used in an initial layer, η be the rejection threshold (see Section 5.1), and $\vec{v}$ be the block-based representation of the current window. We propose the following SC algorithm:

1. $score \leftarrow \frac{1}{M}\sum_{t=1}^{M} p_t(c = 1|\vec{v})$
2. $t \leftarrow M$
3. While $score > \eta$ and $t < T$ do:
   (a) $score \leftarrow \frac{1}{t+1}\left(score \cdot t + p_{t+1}(c = 1|\vec{v})\right)$
   (b) $t \leftarrow t + 1$
4. If $score < \eta$ reject window $\vec{v}$, otherwise output $score$.

The cascade works by first gathering enough evidence for the window $\vec{v}$, through the use of M trees in the initial layer (step 1). After this initialization, a new tree is added at each layer of the cascade (step 3.a) and the score is updated. The process continues until all the trees have been added or the score is lower than η. In this case the window is rejected and the evaluation stops.

As we will see in the results section, the SC provides a significantly faster detection. This is due to the fact that a large majority of windows are rejected at early stages of the cascade, and thus there is no need to compute the probability for all the trees of the forest on these windows.

4. Related work

Closely related to our work, there are recent patch-based methods that make use of a specific type of RF called Hough Forests (HF) [14, 20, 16]. The HF approach takes up the idea of the standard RF, but applies it on a patch level. Here, the leaf nodes take up the role of visual words, and each of them stores a vote distribution for the relative position of the object center. The votes from activated leaf nodes are combined in a Hough Voting space, and object locations are determined as local maxima of the vote distribution.

While HF is used for classifying the individual local patches, our RF is used for classifying the entire object at once. For this purpose, at training time each node of the tree receives a subset of window samples containing the entire object and decides which local patch is most discriminant based on the given data. As a result, each tree of the forest provides an ensemble of local experts, where each expert is specialized in a different local patch of the object.

Another important difference with respect to [14, 20, 16] is that in these approaches the collection of local patches is sampled beforehand from the window and introduced into the tree. Therefore, each node of the tree is forced to learn each patch of the collection, regardless of whether or not this patch is discriminant for classifying the whole object. In contrast, in the proposed method each node of the tree automatically selects the local patch that is found to be the most discriminant one, based on the subset of samples received. Furthermore, by using the RF machinery the local patch selected by each node complements, in a discriminative sense, the local patches selected by its ancestors in the tree, obtaining a strong ensemble of local experts. At the end of the process each tree of the forest has selected a different collection of discriminant local patches, increasing the robustness and generalization capability of the final classifier.

Recently, Menze et al. [17] proposed a new RF classifier based on ridge regression for obtaining discriminant linear splits at the node level. This work resembles our proposal in the sense that we also make use of linear discriminant learners at split nodes. The main difference is that, rather than using a linear regression model, we make use of linear SVMs which are better adapted to our classification task. This is combined with a patch-selection strategy in such a way that each split node determines a local expert of the ensemble. More similar to our work is [26], where each split node selects a rectangular patch and applies a linear SVM onto it. The main difference is that [26] requires to extract many complex region-based descriptors (one bag-of-words histogram for each node visited in the forest), while we only need to extract a single descriptor (in particular, we extract either the HOG or the HOG-LBP descriptor, see section 5) for the whole forest, which is much more efficient.

5. Results

The INRIA dataset [5] is currently being used in the literature as a training-validating dataset. Then, once the best parameters are found during the validation, authors usually report additional results on other challenging datasets. We followed a similar procedure.

Due to the large number of parameters, most of them were estimated by testing just a few reasonable values. The selected values are described in Section 5.1. There were two parameters, however, that were exhaustively optimized using a validation set (*i.e.* the INRIA testing dataset). In Section 5.2, we describe the corresponding experiments. In Section 5.3, we provide a comparison against the best state-of-the-art methods on these other datasets. Finally, Section 5.4 provides an evaluation of the computational cost obtained with different alternatives.

5.1. Experimental setup

In this work, we evaluated the use of both HOG [5] and HOG-LBP [23]. Some modifications were introduced into the HOG-LBP descriptor: i) we used the same spatial partition as in HOG for LPB, resulting in 105 spatial blocks; ii) we did not interpolate the pixels around the compared central one, in order to prevent the texture information from being distorted; iii) in order to add robustness against noise, we used an offset when comparing the central pixel with its neighbours; and iv) we only used the luminance channel. Altogether, these changes permitted us to reduce the computational cost while maintaining an accuracy similar to the one of the original definition [23].

Regarding the computation of the sliding window, we used a step size of eight pixels, which allows to reuse overlapping blocks. For the multi-resolution pyramid we set the scale stride to 1.05. During the Random Forest construction we used the following stopping criterion. A node is no longer split if either of the following conditions occurs: a) its depth is larger than 6 levels[3]; b) the subset of samples contains less than 10 samples; or c) the percentage of samples from the same class is above 99%. This type of stopping criterion is standard [3], and the specific values were observed to provide good results on the INRIA testing dataset. We used a fixed threshold $\eta = 0.1$ in the SC (section 3.5). In the Bootstrapping procedure (section 3.4) the hard negatives are defined as those negative samples whose classification confidence is larger than 0.25. We performed the standard per-image evaluation used in pedestrian detec-

[3] In preliminary experiments we saw that the performance was no longer improving when increasing the number of levels. Therefore, we decided to fix its value for the rest of the validation stage.

tion [15, 9, 10]. In order to quantify the performance we used the well-known Caltech pedestrian toolbox [9].

In the Caltech dataset, we added a so-called Candidate Generation Pruning (CGP) step to our system, in order to obtain a fair comparison with the best performer in this dataset [19]. The CGP step makes use of three assumptions that are accomplished for the mentioned dataset: i) the frames are automatically aligned with the horizon line, ii) the pedestrians are standing on the floor, and iii) the floor is flat. Assuming these conditions and making use of projective geometry [19], the CGP algorithm removes all the candidate windows whose relationship vertical position vs scale is not physically plausible. This permits us to both accelerate the detection of pedestrians (as fewer windows are evaluated by the classifier), and remove false positives standing on non-plausible locations of the scale-space pyramid, thus improving the resulting accuracy. Sections 5.3 and 5.4 show the performance of our system both including a CGP step and not including it.

5.2. Estimation of parameters

Two parameters were exhaustively optimized using the test set of the INRIA dataset. The first one is the maximum patch size L selected by each local expert (see section 3.2). This parameter represents the compromise between having an expert that is based on *local* (*i.e.* small) regions and the use of distinctive (*i.e.* large) regions. In Fig. 2(a) we can see that the accuracy increases as we increase the maximum patch size, until it reaches 3×3 blocks. Permitting larger patches leads to lower accuracy.

The second parameter is the number of bootstrapping iterations N_{boot}. This parameter is important due to the high computational cost of each bootstrapping iteration, which makes it necessary to estimate the minimum number of iterations that provide an accuracy converging to the maximum. In Fig. 2(b) we can see that the accuracy saturates with 20 iterations (we also tried 25 and 30 iterations, but the accuracy was no longer increasing). The results using HOG features were analogous to the HOG-LBP ones.

Respecting the SVM parameters, in order to speed up the training computation, these were fixed during the entire learning procedure.

5.3. Comparison with the state-of-the-art

Our final detector was evaluated using three well-known, challenging datasets: Caltech [9], Daimler [10] and ETH [12]. Results are shown in Fig. 3, where we also include results on INRIA for completeness, and where we compare the accuracy of our approach against the state-of-the-art. Regarding the Caltech dataset, most of the works in the literature only use the so-called "Reasonable" subset, so that we use this subset as reference. However, we also show results on the rest of the subsets of Caltech [9] for completeness. We only use our CGP approach in the Caltech testing dataset (where we used the Caltech training data for estimating the CGP parameters).

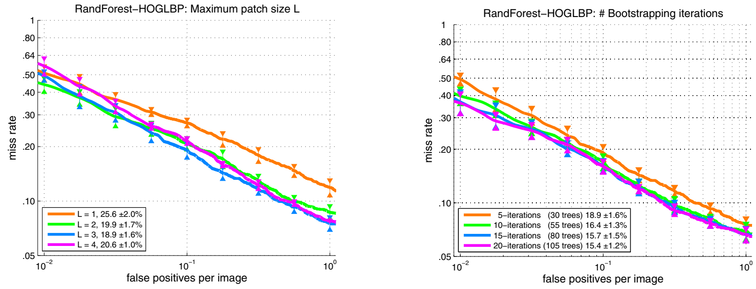
Our method matches or outperforms the state-of-the-art methods in all three datasets. Only in the Caltech "Reasonable" subset three methods outperform our approach (if we do not include the CGP component), although the third best performer has a similar accuracy to the one of our method. If we use CGP, the accuracy increases. In this case, our method matches the second best performer (MultiFtr-Motion [22]) and it is outperformed by only one method (MultiResC (CGP) [19]) in the reasonable evaluation subset. It is worth mentioning that these methods make use of additional sources of information (multi-resolution in MultiResC (CGP) [19] and motion in MultiFtr-Motion [22]). Both [22] and [19] make use of block-based representations in order to include these sources of information, so that they can be integrated in our framework with moderate changes. This would further increase the accuracy of our method.

Regarding the rest of the other Caltech evaluation subsets, we show results for the two most difficult cases: partial-occlusions and medium-scale (where the size of the pedestrian is fairly small). Our method suffers in both cases (Fig. 4 shows detection examples). For partial-occlusions, however, it ranks the first. This is consistent with the fact that local-patch based methods are usually more robust against partial occlusions than the holistic ones. For small pedestrians, it suffers more than the MultiResC method. Integrating multi-resolution in our method would help to handle better different sizes of the pedestrian, resulting in improved performance in this dataset. The overall and near-scale data-set are not included for lack of space. In the overall subset, our method ranks second (with miss rate 82.8%, 1.6 points worse than the first). In the near-scale, it ranks third (with miss rate 29.6%, 8.4 points worse than the first). Again, using multi-resolution would be beneficial here, for describing the fine details of close pedestrians.

5.4. Testing speed

In order to evaluate the computational cost, we used a laptop machine with a i7-2860QM CPU at 2,50GHz. Furthermore, we parallelized our code in order to compute several scales of the pyramid at the same time, and in order to compute several trees of the forest at the same time (this last parallelization was only performed for our baseline and it was not performed when using the SC component).

Table 1 shows the runtime of the proposed approach, including the baseline without any speed-up, the use of SC and the use of both SC and CGP. If we consider pedestrians with a minimum height of 96 pixels, the system operates at 4 fps with HOG, and 3 fps with HOG-LBP. This can be further sped up if we use some hardware optimization. For this purpose, we used AVX instructions in order to implement



(a) (b)

Figure 2. (a) Performance as a function of maximum patch size L, (b) performance as a function of the number of bootstrapping rounds.

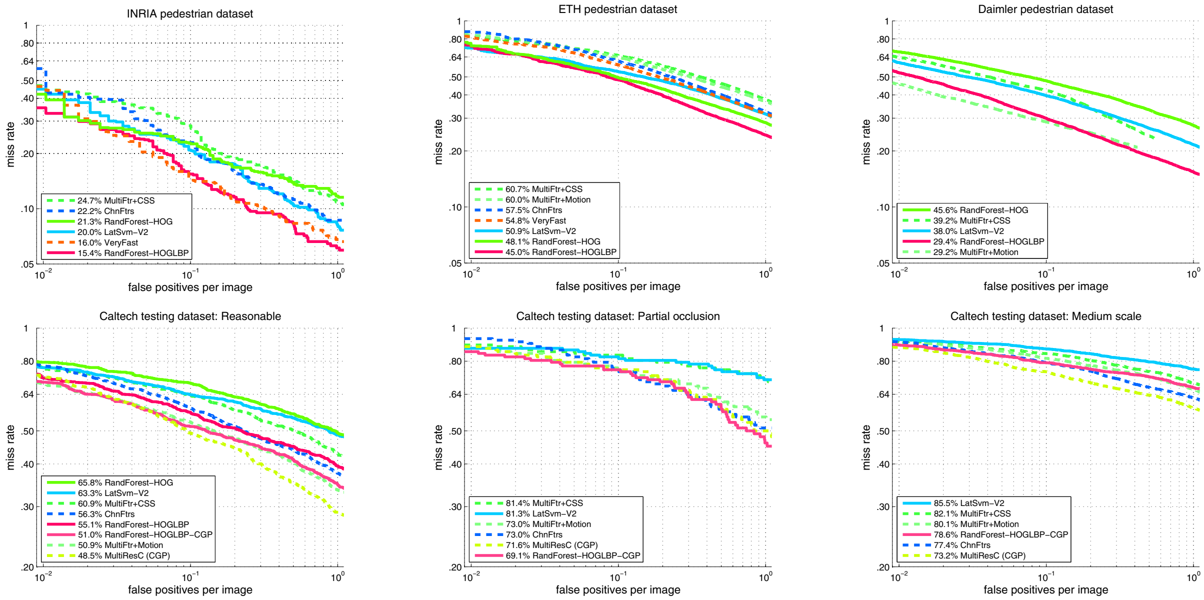


Figure 3. Miss rate versus false positive per image curves in the INRIA, Daimler, ETH and Caltech testing. For the Caltech testing dataset we show results under three different conditions: reasonable, partial occlusion and medium-scale(please refer to [9] for further details).

the dot product involved in the SVM classification. In this case, we reached 5.9 fps with HOG and 4.6 fps with HOG-LBP. These times make the resulting system fairly fast in comparison with the state-of-the-art, as evaluated in [9]. As an example, the two fastest detectors evaluated in that survey operate at 1.2 fps and 6.5 fps, while the proposed approach operates at 2.5 fps without any optimization (using only the SoftCascade and excluding the CGP step) and at 4.6 fps if we use both AVX instructions and the CGP step. At the same time, the proposed approach ranks in the top positions in terms of accuracy, as shown in this section.

	$\geq$ 50 pixels		$\geq$ 96 pixels	
	HOG	HOG-LBP	HOG	HOG-LBP
RandForest	0.15	0.09	0.75	0.53
SoftCascade	0.60	0.45	2.51	1.88
SoftCascade + CGP	1.23	0.93	4.01	3.17

Table 1. Detection times in frames per second (fps) in the Caltech dataset. The second and third columns show, respectively, the fps when the minimum pedestrian height is 50 and 96 pixels.

6. Conclusions

We presented a novel approach for estimating ensembles of local experts through the RF framework. The proposed approach works with rich block-based descriptors which are reused by the different experts of the ensemble in such a way that each expert selects the most discriminant local patch based on this descriptor. Making use of the RF framework, the patches selected by each tree are both discriminant and complementary, and at the end of the process the forest estimates a diverse collection of ensembles providing both robustness and generalization capabilities. As part of the work, we show how to integrate the proposed RF classifier with both a SC architecture and a simple yet effective CGP algorithm, which permit to significantly speed up the detection, as shown in the results. We also showed that the proposed CGP algorithm increases the accuracy by avoiding false positives in unexpected areas of the image.

Altogether the proposed work provides an interesting

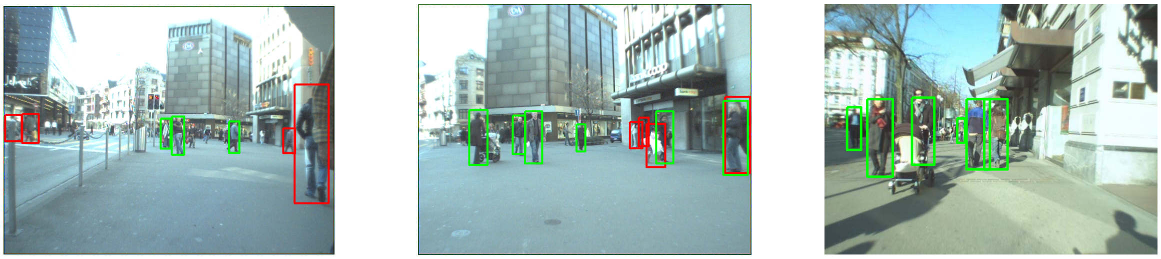


Figure 4. Examples with small pedestrians and occlusions. Green and red boxes symbolize true positives and false negatives respectively.

framework that permit to match the best approaches in terms of accuracy, as measured across several challenging datasets, without including additional sources of information such as motion, multi-resolution or colour. These sources can be easily integrated in the future in order to further increase accuracy. At the same time, we showed that the proposed architecture provides a quasi real-time performance on pair with some of the fastest approaches. This is due to the integration of the SC component, and the use of the CGP algorithm proposed in this work.

Acknowledgements

Work supported by the spanish projects TRA2011-29454-C03-01 and TIN2011-29494-C03-02.

References

[1] R. Benenson, M. Mathias, R. Timofte, and L. V. Gool. Pedestrian detection at 100 frames per second. In *CVPR*, 2012. 1

[2] L. Bourdev and J. Brandt. Robust object detection via soft cascade. In *CVPR*, 2005. 4

[3] A. Criminisi, J. Shotton, and E. Konukoglu. Decision forests for classification, regression, density estimation, manifold learning and semi-supervised learning. Technical report, Microsoft Research, 2011. 2, 3, 5

[4] S. Dai, M. Yang, Y. Wu, and A. Katsaggelos. Detector ensemble. In *CVPR*, 2007. 1

[5] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *CVPR*, 2005. 1, 2, 5

[6] N. Dalal, B. Triggs, and C. Schmid. Human detection using oriented histograms of flow and appearance. In *ECCV*, 2006. 1

[7] P. Dollár, R. Appel, and W. Kienzle. Crosstalk cascades for frame-rate pedestrian detection. In *ECCV*, 2012. 1

[8] P. Dollár, Z. Tu, H. Tao, and S. Belongie. Feature mining for image classification. In *CVPR*, 2007. 1

[9] P. Dollár, C. Wojek, B. Schiele, and P. Perona. Pedestrian detection: An evaluation of the state of the art. *TPAMI*, 2012. 2, 6, 7

[10] M. Enzweiler and D. M. Gavrila. Monocular pedestrian detection: Survey and experiments. *TPAMI*, 2009. 2, 6

[11] M. Enzweiler and D. M. Gavrila. A multilevel mixture-of-experts framework for pedestrian classification. *TIP*, 2011. 1

[12] A. Ess, B. Leibe, and L. van Gool. Depth and appearance for mobile scene analysis. In *ICCV*, 2007. 2, 6

[13] P. Felzenszwalb, R. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part-based models. *TPAMI*, 2010. 1, 2

[14] J. Gall and V. Lempitsky. Class-specific hough forests for object detection. In *CVPR*, 2009. 1, 5

[15] D. Gerónimo, A. López, A. Sappa, and T. Graf. Survey of pedestrian detection for advanced driver assistance systems. *TPAMI*, 2010. 6

[16] B. Leibe, A. Leonardis, and B. Schiele. Robust object detection with interleaved categorization and segmentation. *IJCV*, 2008. 1, 5

[17] B. H. Menze, B. M. Kelm, D. N. Splitthoff, U. Koethe, and F. A. Hamprecht. On oblique random forests. In *ECML-PKDD*, 2011. 5

[18] K. Mikolajczyk, C. Schmid, and A. Zisserman. Human detection based on a probabilistic assembly of robust part detectors. In *ECCV*, 2004. 1

[19] D. Park, D. Ramanan, and C. Fowlkes. Multiresolution models for object detection. In *ECCV*, 2010. 6

[20] D. Tang, Y. Liu, and T.-K. Kim. Fast pedestrian detection by cascaded random forest with dominant orientation templates. In *BMVC*, 2012. 1, 4, 5

[21] P. Viola and M. J. Jones. Robust-real time face detection. *IJCV*, 2004. 1

[22] S. Walk, N. Majer, K. Schindler, and B. Schiele. New features and insights for pedestrian detection. In *CVPR*, 2010. 1, 6

[23] X. Wang, T. X. Han, and Y. Shuicheng. An HOG-LBP human detector with partial occlusion handling. In *ICCV*, 2009. 1, 5

[24] C. Wojek and B. Schiele. A performance evaluation of single and multi-feature people detection. In *DAGM*, 2008. 1

[25] B. Wu and R. Nevatia. Detection and segmentation of multiple, partially occluded objects by grouping, merging, assigning part detection responses. *IJCV*, 2009. 1

[26] B. Yao, A. Khosla, and L. Fei-Fei. Combining randomization and discrimination for fine-grained image categorization. In *CVPR*, 2011. 5

[27] Q. Zhu, S. Avidan, M.-C. Yeh, and K.-T. Cheng. Fast human detection using a cascade of histograms of oriented gradients. In *CVPR*, 2006. 1