Local Context Priors for Object Proposal Generation

Marko Ristin[1], Juergen Gall[2], and Luc Van Gool[1,3]

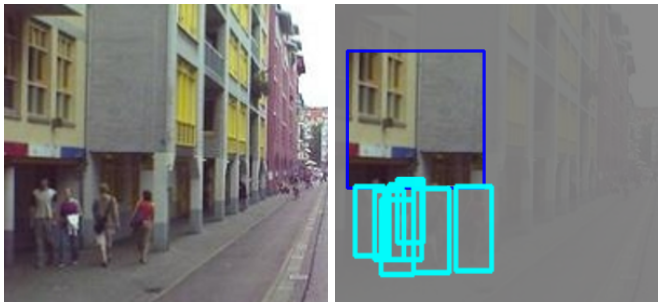
[1] ETH Zurich [2] MPI for Intelligent Systems [3] KU Leuven

Abstract. State-of-the-art methods for object detection are mostly based on an expensive exhaustive search over the image at different scales. In order to reduce the computational time, one can perform a selective search to obtain a small subset of relevant object hypotheses that need to be evaluated by the detector. For that purpose, we employ a regression to predict possible object scales and locations by exploiting the local context of an image. Furthermore, we show how a priori information, if available, can be integrated to improve the prediction. The experimental results on three datasets including the Caltech pedestrian and PASCAL VOC dataset show that our method achieves the detection performance of an exhaustive search approach with much less computational load. Since we model the prior distribution over the proposals locally, it generalizes well and can be successfully applied across datasets.

1 Introduction

Object detection is a well studied field in computer vision. While most works have focused on improving the accuracy [1], the computational burden of detectors is another important issue that needs to be solved. Object detectors commonly process the image at different scales, where only a small region of the image is processed for evaluating a single object hypothesis. In the extreme case, all possible rectangular regions, termed windows, are checked for whether they contain an instance of the object class. Although not all windows are classified in practice, a dense sampling of the windows with a small stride is necessary to obtain a high detection accuracy [2]. Since these sliding window approaches are very expensive, several strategies have been proposed for reducing the processing time. Among them, cascading [3, 4] is the most popular approach. It makes use of the fact that some parts of the image can be easily discarded with simple features and classifiers, such that a full processing of these image parts can be avoided. Cascading is specific for a classifier and can significantly reduce the computation time at a small expense of accuracy. When object detection is formulated as an optimization problem, where one searches for instances with the highest detection score in the full image, branch-and-bound techniques can be used to search the space of windows more efficiently [5].

A different concept is followed by approaches that generate window proposals. While sliding window can be regarded as sampling uniformly from the set of windows, one can also learn a distribution over the set of windows that gives



a) b)

Fig. 1. When presented with an image a), our algorithm sees patches which might not contain any pedestrians, but whose visual appearance suggests that there could be some pedestrians walking on the sidewalk below.

a higher probability to parts of the image where objects are expected and a lower probability to parts where objects are not expected. The advantage of the sampling methods is that highly probable candidates are processed first, such that the number of samples can be easily adapted to the available computational resources.

Previous work on proposal generation [6–10] has focused on disregarding windows that do not contain the object. For instance, sky or a building facade are ignored for detecting pedestrians. We propose to randomly sample very few, but large image patches and extract information about the image context from each one of them based on appearance. These nuggets of information are later combined to estimate a probabilistic prior distribution over object location in the image. For instance, instead of ignoring a facade of a building, it is a good indicator for pedestrians and their scales on the sidewalk below (see Fig. 1).

In order to model local probabilistic priors from image context, we learn a regression from large local image patches to its closest annotated object from training data. During testing, we combine the local probabilities from several patches to obtain a distribution over the space of windows. In our experiments, we show that on the challenging Caltech benchmark [11] as little as 0.2% of the windows need to be evaluated to achieve the same performance as processing all windows. Although local priors do not assume that the recording settings for training and testing data are the same, as we will show in a cross-dataset experiment, they can be combined with global priors if such global information is available.

2 Related work

The exhaustive search methods based either on the sliding window approach [2] or on part-based models [4, 12, 13] are well-established approaches for detecting objects and proved to be the state-of-the-art according to recent Pascal VOC challenges [1]. The number of windows examined in an exhaustive search, however, grows at least linearly with the number of image pixels. Since the image

needs to be upscaled for detecting small objects, the time for inspecting all windows can exceed the runtime requirements of real-world applications already for images as small as 320×240 pixels even on modern computers.

As there are usually far fewer windows containing an object of interest than the windows without it, methods were developed based on a cascade of classifiers [3, 14–17]. High-confident negatives are rejected in early stages of the cascade and more computational time is spent on windows that are more difficult to classify. The approach was improved by studying various combinations of features and attained state-of-the-art speed performance at a reasonable accuracy rate for pedestrian detection [18].

While conventional cascade classifiers still have to inspect the whole set of windows, the *Efficient Subwindow Search* (ESS) [5] and its cascaded variant [19] employ a branch-and-bound scheme to inspect only relevant regions. It approximates the response of the original classifier over a region by analytically derived bounding functions. For object detection, the image is first split into large regions and the search then continues recursively in the region ranked highest. The recursion stops when the window with the highest score is found. Since the performance of ESS depends on the tightness of the bounding functions, it has been also proposed to estimate the bounding functions [20].

Other methods focus on an efficient selective search for candidate windows which then serve as input for a classifier. In [7], it was proposed to estimate a likelihood function over the windows based on the response of some classifiers. The method refines the likelihood step-wise using Monte Carlo sampling to specifically draw samples from regions where target objects are more likely. Each refinement stage employs a more accurate and computationally more intensive classifier. This method basically combines coarse-to-fine search with cascading. The work in [6] employed a two-stage procedure. In the first stage, separate classifiers are applied, one for each aspect ratio/scale. The windows are then ranked by a pre-trained ranking classifier. In [21] and based on [22], the authors learn to predict candidate windows from discriminative visual words. A different bottom-up procedure was proposed in [23]: the image is first segmented by an unsupervised technique into locally coherent regions that are later reshaped and combined by the algorithm into rectangles, some of which are discarded as unlikely to contain an object. The approach of [9] begins with over-segmenting the image, proceeds with gradually joining similar regions to construct a hierarchical segmentation and then generates candidate windows based on such a pyramid of segmentations. In [10], a naïve Bayes model is trained to distinguish windows with high "objectness", defined as the probability of a window containing an object of all classes of interest, from windows containing background based on multiple cues. Given a testing image, windows on a regular grid are scored based on saliency. Candidate windows are then sampled from a distribution given by the saliency scores and the trained model. In [8], an initial set of about 10^5 windows is generated based on super-pixel segmentation of the image and a global prior distribution estimated from the training images. The features measuring "objectness" are extracted for the windows and based on these features the final

set of 100 or 1000 windows is selected. The approach of [24] estimates surface orientation and camera viewpoint to predict the scales and positions of the objects in order to improve the accuracy of object detectors. Since making these estimations in a general setting is still a very difficult problem, the method is not applicable for general, time-critical applications.

In contrast to previous approaches for proposal generation that focus on disregarding local image parts that are unlikely to contain an object, we propose a complementary approach where local image parts predict the occurrence of the closest object as shown in Fig. 1. While our approach uses image context to reduce the search space to promising windows, there are various methods that employ scene context or inter-object relations to boost the detection accuracy [25–29]. This is, however, not the focus of the paper.

3 Local Context Priors

Due to the high variation of objects with respect to image position, size, and aspect ratio, the number of image regions, termed windows, that need to be classified in order to detect an object, easily exceeds 1 million for a single image. In particular, detecting pedestrians on a wide range of scales as in the Caltech benchmark [11] is very expensive. In this work, we investigate a method to reduce the number of windows to be evaluated in order to get the same results as commonly used detectors like [2] that process all windows.

The sliding window principle can be formulated as sampling from the set of all windows $\mathcal{W}$, where a window is denoted by $W = (x, y, w, h)$ with (x, y) being the center of the window, w the width, and h the height of the window. The sampling is performed according to a distribution $p(W|I)$, which depends on the image I. Each window is then classified according to a probability $p(c|W, I)$ or a scoring function that gives high values to windows that are tight bounds around an instance of the object class c. Although not all detectors are probabilistic, we use the probabilistic formulation:

$$p(c|W, I)p(W|I). \tag{1}$$

For sliding window detectors, $p(W|I)$ is a uniform distribution over $\mathcal{W}$. We aim to learn a distribution $p(W|I)$ that gives a higher probability to image regions where instances of the object class c can be expected from the context. In our case, the probability is not modeled globally over the full space but locally over local image patches $\mathcal{P}(\boldsymbol{y})$ located at $\boldsymbol{y}$ as illustrated in Fig. 1. In this way, we do not have to process the full image first and can thus compute very efficiently local priors over $\mathcal{W}$ from local image information. Another advantage is that the sampled patches do not need to include the objects of interest. Instead, the context gives information where relevant objects could be. For instance, the building in Fig. 1 gives information that the closest pedestrians are expected

below the building. In order to handle N local priors, we combine them by

$$p(W|I) = \frac{1}{N} \sum_i p(W|\mathcal{P}(\boldsymbol{y}_i)), \tag{2}$$

$$\text{where} \quad \boldsymbol{y}_i = (x_i, y_i) \quad \text{and} \quad p(W|\mathcal{P}(\boldsymbol{y}_i)) = p((x - x_i, y - y_i, w, h)|\mathcal{P}(\boldsymbol{y}_i)). \tag{3}$$

The right hand side of (3) models the relative location of a window with respect to the patch location $\boldsymbol{y}_i$. In this way, the probability becomes invariant to global translations. The context is also not learned explicitly, but implicitly and directly from the image data. Hence, the approach does not require any expensive computation and the prior (2) can be computed within 10ms as we will show in the experiments.

For learning the local priors, we use regression forests [30] that have been previously applied to a variety of regression problems in computer vision [31–34]. In the sequel, we outline the learning procedure of the priors and its application for object detection.

3.1 Training

For learning a local prior $p(W|\mathcal{P})$ given an image patch $\mathcal{P}$, we have to collect some training pairs $(\mathcal{P}_i, W_i)$. To this end, we randomly sample fixed-size patches $\mathcal{P}_i$ from images that contain at least one annotated object. For each patch, we then search for the closest annotated bounding box (x, y, w, h), with (x, y) being the center of the bounding box, to the patch center $\boldsymbol{y}_i$ and use the relative position as window:

$$W_i = (x - x_i, y - y_i, w, h). \tag{4}$$

In some cases, the closest bounding box does not correspond to the closest expected occurrence of an object. For instance, a building on the left hand side of a street might be associated with a pedestrian on the right hand side of the street since the training image captured only a scene where a pedestrian appeared on the right hand side. In order to enforce locality and reduce noise since not all plausible locations and scales of the objects are annotated, we only take patches that have at least some overlap with an annotated bounding box; see Fig. 1.

Having collected the training pairs $(\mathcal{P}_i, W_i)$, we learn a regression forest as in [30]. For each tree in the forest, we select a random subset of our training data and train each tree recursively. To this end, we generate at each node a set of binary tests. Each test t is defined by a random feature and a random threshold and splits the training data arriving at the current node. We evaluate the splitting quality of each test t using the information gain:

$$IG_t = H(A) - \sum_{k=\{0,1\}} \frac{|A_k(t)|}{|A|} H\left(A_k(t)\right) \tag{5}$$

where H denotes the entropy, A the samples arriving at the node and $A_k(t)$ the split sets of A obtained by test t. For efficiency, we use a Gaussian approximation

of the distribution over $\mathcal{W}$ as in [31, 33]:

$$H(A) = 2\left(1 + \log\left(2\pi\right)\right) + \frac{1}{2}\log\left(|\Sigma^W|\right), \tag{6}$$

where Σ^W is the covariance matrix of the windows W in the set A.

Once the optimal split which maximizes the information gain is found, the parameters of the test function are stored at the node and the construction of the tree continues recursively on the two subsets given by the split. As soon as the number of samples arriving at the node is below a threshold or the maximal depth is reached, a leaf node is created. At each leaf, we store the mean $\overline{W}$ and covariance matrix Σ^W of the windows W ending in the leaf during training.

3.2 Testing

For testing, we randomly sample a set of patches $\mathcal{P}(\boldsymbol{y}_i)$ from the image as illustrated in Fig. 2. Each patch is then passed through the random forest consisting of L trees, ending in a leaf l for each tree. Based on the normal distributions $p(W|l) = \mathcal{N}(W;\ \overline{W}_l, \Sigma^W_l)$ stored at the leaves, we compute the average of the leaves as in [30] and obtain the normal distribution given the patch $\mathcal{P}(\boldsymbol{y}_i)$:

$$p(W|\mathcal{P}(\boldsymbol{y}_i)) = \mathcal{N}(W;\ \overline{W}, \Sigma^W), \tag{7}$$

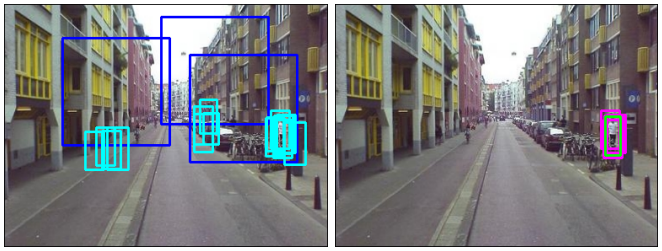
$$\overline{W} = \frac{1}{L}\sum_l (\overline{x}_l + x_i, \overline{y}_l + y_i, \overline{w}_l, \overline{h}_l), \tag{8}$$

$$\Sigma^W = \frac{1}{L^2}\sum_l \Sigma^W_l. \tag{9}$$

Since the regression forest models only the relative location of the windows, we have to add the patch center $\boldsymbol{y}_i = (x_i, y_i)$ to the mean in (8). To obtain a full distribution over the set of windows $\mathcal{W}$, we combine the local priors of the sampled patches $\mathcal{P}(\boldsymbol{y}_i)$ by a sum of Gaussians (2).

In order to use the local priors for object detection, we generate N samples from the distribution and run an object detector on the image regions of the sampled windows. Note that the local priors are specific to an object category, but not to any detector. For each sampled patch $\mathcal{P}(\boldsymbol{y}_i)$, we sample ρ windows from the corresponding normal distribution (7). Keeping the overall number of windows N fixed, the number of sampled patches $\mathcal{P}(\boldsymbol{y}_i)$ is then given by $\frac{N}{\rho}$. The parameter ρ is basically a trade-off between sampling locally based on (7) and exploring the full image by sampling more patches; see Fig. 2.

While (2) weights the Gaussians uniformly, we have also investigated to weight the Gaussians based on the variance. To this end, we weight each Gaussian $p(W|\mathcal{P}(\boldsymbol{y}_i))$ by $w_i \propto \frac{1}{\log(|\Sigma^W|)}$, where $\sum_i w_i = 1$. This is motivated by the observation that leaves with high variance are less confident about the location of the closest objects than leaves with low variance. Therefore, the weighting focuses the local sampling on the patches that are more confident. In our experiments, we observed that the weighting of the Gaussians improves the results



a) b)

Fig. 2. Example illustrating the local prior method. a) Exploration patches (blue) are generated uniformly at random and suggested windows (cyan) are sampled from the estimated distributions. b) Hits (ground truth is painted green, hits are painted purple).

slightly for small sample sizes, but the difference to (2) becomes negligible for larger sample sizes.

4 Experimental results

We focus our evaluation on three challenging datasets. The Caltech pedestrian detection benchmark [11] contains real-world sequences captured with a camera mounted on a car. The publicly available dataset consists of about 1M frames in 640×480 pixel resolution, split over a training and a testing set. A few example images are shown in Fig. 3. We followed the evaluation protocol of [11] and used the settings "reasonable" and "overall". The corresponding sets will be referred to as $caltech_{reasonable}$ and $caltech_{overall}$. For the parameter evaluation, we did not use all the frames, but subsets (every 300th frame; 402 frames in total). We denote them $caltech^{300}_{reasonable}$ and $caltech^{300}_{overall}$. The second dataset for pedestrian detection is taken from [35]. It was recorded from a mobile platform in an urban environment and contains 290 annotated images. We rescaled the images from the resolution 384×288 to 640×480 pixels. We will refer to this dataset as *amsterdam* and use it only for testing. We also evaluated our method on PASCAL VOC 2007 and 2006 [1] to further examine the performance on other object categories and non-urban environments. In the following figures, the mean value and standard deviation are reported over five runs.

4.1 Caltech

We chose patch size to be large relative to the image and fix $\mathcal{P}$ to 256×256 pixels. As image features, we use histograms of gradients [2] (*HOG*) or generalized Haar features [36] which can be efficiently implemented using integral images [37]. We will refer to these features as *Haar* features. As a pre-processing step before feature extraction, patches are down-sampled to the dimension of 128×128 pixels.

The regression forests have been trained with 5 trees, where each tree was trained on a random subset of 450 images containing at least one pedestrian.

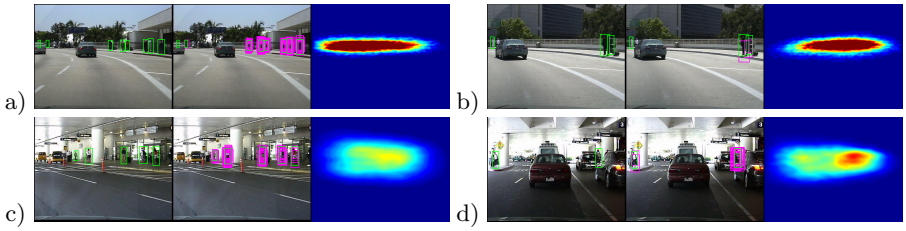


Fig. 3. Results on images from Caltech pedestrian benchmark [11] obtained by our method with $\rho = 20$, Haar features, and 5,000 sampled windows. Each image triple consists of: 1) ground truth (green), 2) suggested windows hitting the ground truth (purple), 3) density over the centers of the windows. a) The sidewalk area is correctly detected. b) 5,000 samples are not enough to obtain good hits for the pedestrians. c) Pedestrians are correctly recognized in an indoor setting. d) The method correctly detects that pedestrians are less likely to appear in the middle of the street. Best viewed in color.

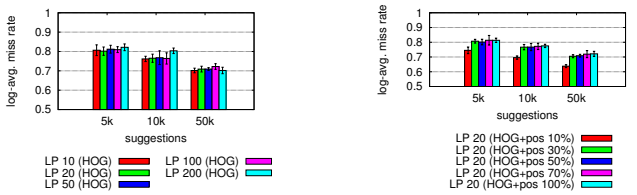
From these images, 50,000 patches $\mathcal{P}$ were extracted for training. For the training of a single node, 30,000 random tests were assessed. The minimal number of patches arriving at a leaf was set to 10 and the maximum tree depth was set to 20.

Fig. 3 shows a few example results from $caltech_{overall}$. Given an image, the method computes the distribution consisting of a sum of Gaussians over the windows $W = (x, y, w, h)$. The figure visualizes only the distribution over (x, y). The distributions change depending on the image. While the distributions in Fig. 3a) and b) focus on the sidewalk which is nearly horizontal in these images, the distributions in c) and d) cover a larger area of the image. Since the sampled windows are limited to 5,000 in this example, some of the pedestrians are missed in b) although the estimated distribution is reasonable.

For quantitative evaluation, we classified the proposed windows using a SVM classifier and histogram of gradients features as in [2]. In our experiments, we used the OpenCV implementation of [2]. We will refer to it as *classifier HOG*. Although our method can generate continuous values for the windows, we limited the results to the set of windows generated by the sliding window approach in order to provide a fair comparison. We used a multiplicative scale stride of 1.05 and positional stride of 4 pixels. For the "reasonable" setting, the image scale range was $[0.5, 2.5]$ and for the "overall" setting, it spanned $[0.5, 5.2]$.

In Fig. 4a), we show the performance with respect to the parameters ρ and the number of sampled windows N from the prior distribution. As measure, we use the log-average miss rate as in [18]. As one can see, the miss rate decreases with an increasing number of sampled windows. The impact of ρ can be said to be negligible up to a certain level. For the rest of the experiments, we used $\rho = 20$ as parameter for the local priors (LP 20).

To evaluate the impact of the patch size, we varied the patch size for LP 20 HOG with 10k windows on $caltech_{reasonable}$. The log-avg. miss rates were: 0.79 (32×32), 0.75 (64×64), 0.74 (128×128), 0.74 (256×256), 0.72 (384×384). This shows that the impact of the particular size is rather small as long as the



a) Parameter ρ b) Weight of the global image coordinates

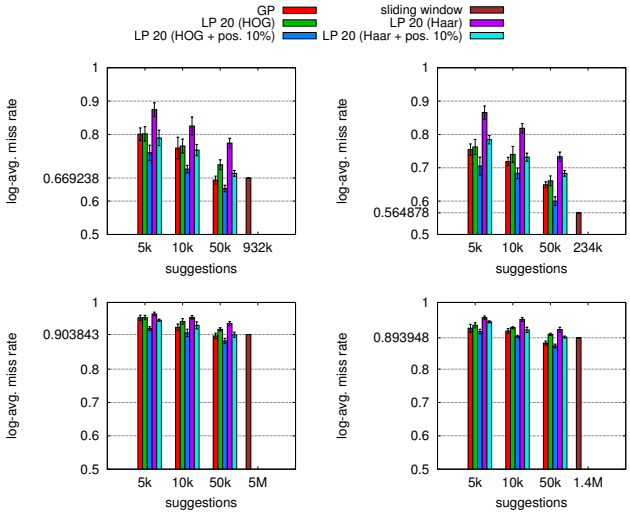
Fig. 4. a) Log-average miss rate with respect to parameter ρ. **b)** Log-average miss rate with respect to the weight of the global image coordinates (pos.). Parameter ρ was fixed to $\rho = 20$. Both experiments were evaluated on $caltech^{300}_{reasonable}$ with HOG features and in conjunction with HOG classifier.

patches are reasonably large. The patch size of 256×256 is used for the rest of the experiments.

In Fig. 5a) and 6, we compare our method with the sliding window approach on the Caltech dataset using classifier HOG. While the sliding window approach processes about 1 million windows in "reasonable" setting, LP 20 with HOG features requires only 10,000 - 50,000 windows to achieve a comparable performance. This corresponds to 1.1%-5.4% of all windows. In the "overall" setting (Fig. 6b), the computational advantage of our method becomes even more pronounced as the search space grows larger with the increasing number of scales: sliding window processes 5 million windows, while our method needs to inspect only 0.2%-1.0% thereof for a comparable performance. Since also detecting small objects is of utter importance for many real-world application like driver assistance systems [11], the "overall" setting is highly relevant.

We also compared HOG features with Haar features for learning the local priors. While the Haar features perform worse than the HOG features, the Haar features are 9 times faster to compute.

We also compare the local priors to a global prior (GP). It has been shown that spatial Gaussian priors are the most important cue for modeling visual attention [38] so we model the global prior as Gaussian over the set of windows given in absolute spatial coordinates and estimated from the training annotations. Since in Caltech dataset the camera was mounted on a car, sampling from the global prior already improves the sliding window approach. Therefore, we combined the global and local priors by adding the absolute (x, y) coordinates of the sampled training patch centers as additional features for learning the regression forests, where the impact of the local image features (HOG) and the global image coordinates (pos.) were weighted. The results for different settings are shown in Fig. 4b). Learning only with image coordinates corresponds to learning a global prior, which performs worse than HOG combined with patch position. Fig. 5 and Fig. 6 show that the global information improves the local prior for this dataset and performs as good or better than the global prior. The miss rate / false positive per image curves computed for a single evaluation run are shown in Fig. 7a).



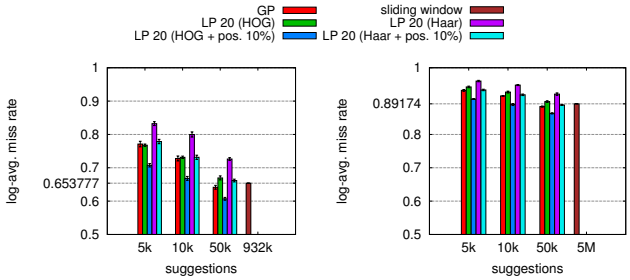
a) Classifier HOG b) Classifier DP

Fig. 5. Comparison between HOG classifier and Deformable parts classifier (DP) performed on $caltech^{300}_{reasonable}$ (upper row) and $caltech^{300}_{overall}$ (bottom row). The global prior (GP) and sliding window serve as base line. The classifier HOG (a) is outperformed by the classifier DP (b).

In order to show that the learned priors are not specific to a detector, we employed a state-of-the-art classifier [4] (referred here as *classifier DP*). The evaluation with the classifier DP is presented in Fig. 5b). When Fig. 5a) and Fig. 5b) are compared, it becomes evident that the classifier DP has a lower miss rate than the classifier HOG, which agrees with the results presented in the literature [11]. The performance of our method changes accordingly.

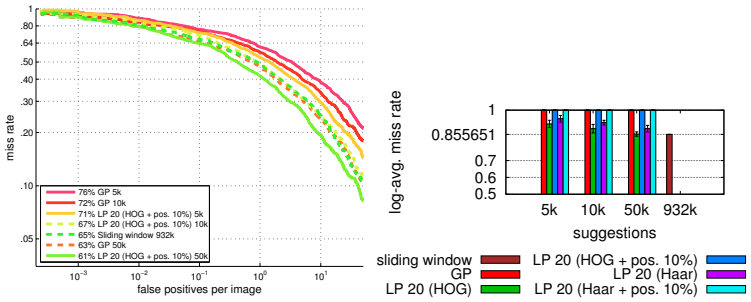
We report here the average runtime of our method in conjunction with the classifier HOG. Sliding window approach took 18s to inspect 932k windows in "reasonable" setting, and 84s in "overall" setting (single-threaded; Intel Core i7-2600K CPU with 4 GB RAM). For benchmark, we used our method to sample 5k windows and run a classifier over them. The generation of windows ($\rho = 20$) with HOG features took 90ms and with Haar features 10ms, respectively. Adding location coordinates to the features did not affect the average runtime. Overall, our approach requires 0.8s for the "reasonable" and 1.3s for the "overall" setting, which corresponds to a runtime reduction by a factor of 23 and 65, respectively. Since the local priors are computed in few milliseconds, they can be used to speed up faster detectors like [18] or to use slower, but more accurate detectors.

Finally, we evaluated on $caltech_{reasonable}$ the benefit of a regression compared to a classification that just discards parts of the image. To this end, a random forest with HOG features was trained to discriminate patches containing objects with the same settings as before. During test time, we ranked patches on a dense grid and thoroughly explored the best ones by sliding window. The inspection stopped at 50k inspected windows. The classification with log-avg. miss rate



a) $caltech_{reasonable}$ b) $caltech_{overall}$

Fig. 6. Performance comparison evaluated on a) $caltech_{reasonable}$ and b) $caltech_{overall}$ using the classifier HOG. The local context priors were trained on corresponding $caltech_{reasonable}$ and $caltech_{overall}$ datasets.



a) $caltech_{reasonable}$ b) $amsterdam$

Fig. 7. a) Comparison of miss rate / false positive per image curves evaluated on $caltech_{reasonable}$ with classifier HOG in a single run (log-avg. miss rate is indicated in the legend). The global prior (GP) and sliding window are used as base lines. The number of suggestions is indicated as suffix. The best performance is achieved by local priors combined with global image coordinates. **b**) Evaluation on *amsterdam* in "reasonable" setting with local priors trained on $caltech_{reasonable}$. Methods relying on global features fail, while the ones based only on local features successfully adapt.

0.78 for 256×256 and 128×128 patch size performs worse than the regression LP 20 HOG (Fig. 6). When we ranked the patches based on the classification, but sampled windows with LP 20 HOG, the performance did not differ significantly for a patch size 256×256. For 128×128, however, the performance improved for 5k windows: 0.71 (5k), 0.70 (10k), 0.66 (50k). This shows that the combination of ranking or discarding image parts with predicting the scale and location of objects using a regression is worth to be explored more in detail in the future.

4.2 Amsterdam

To evaluate the generalizability of the local prior, we have performed a cross-dataset experiment. To this end, the local prior was trained on $caltech_{reasonable}$ and applied to *amsterdam*. The evaluation protocol and the parameters are the same as for the Caltech dataset. The results are presented in Fig. 7b). Due to

variability in camera position, the global prior (GP) completely fails, while the local priors LP 20 without global features still perform well, thus indicating that the local prior does not overfit to the dataset bias in contrast to the global prior.

4.3 PASCAL VOC

In order to demonstrate that the approach also generalizes to other categories than pedestrians in urban environments, we have evaluated the approach on PASCAL VOC 2007 dataset [1].The evaluation protocol "comp3" was followed. The local priors were trained individually for each class on the "trainval" sets and applied to the respective "test" sets. Patch size was set to 128×128. Prior to feature extraction, patches were down-sampled once. Classifier DP [4] was used for the classification. In contrast to the pedestrian datasets, the performance of the Haar features is slightly better than the HOG features. While the HOG features capture shape better than Haar features, which is very important in urban environments, the Haar features also capture color information, which is a useful cue for the general categories. We therefore report results only for Haar features. Table 1a) shows the comparison with [4] as baseline, a related method for window proposal [9] and a branch-and-rank approach [20]. Except for one category, we outperform [20] in terms of average precision (AP), while we match the performance of [9] in many categories, but at significantly lower runtime cost. Our approach requires 7 ± 2ms for proposal generation whereas [9] takes at least 8s for the segmentation.

Due to the subsampling of window space, our approach misses some objects (see Table 1a)), but also removes some false positives. Since [4] is not perfect, subsampling can therefore even increase the AP. As the number of sampled windows increases, the AP converges to the baseline.

We evaluated our method with the same parameter settings on PASCAL VOC 2006 [1] to compare its sampling performance with two other selective search methods [6] and [19]. As in [6, 19], we use the area under overlap-recall curves as measure. Both reference methods proposed 1,000 windows. The results are presented in Table 1b). We outperform [19] on all but two categories and get close to the state-of-the-art [6] on some categories. While [6] and [19] require at least 400ms on average, our method is by more than a factor of 57 faster and requires only 7ms.

5 Conclusion

In this work, we presented a novel method for generating window proposals based on the local context. It sparsely examines the image and incorporates the knowledge extracted from a patch even if it does not contain an object of interest. While the approach generalizes well and can be successfully applied across datasets and for various categories, it can also be adapted to make use of global a priori knowledge. The experiments show that it achieves the detection performance of a computationally expensive exhaustive search in a fraction of the

Table 1. a) Comparison of performance on PASCAL VOC 2007 by average precision (AP). Our method outperforms [20] and [10] and matches [9] at lower runtime cost. **b**) Comparison of performance on PASCAL VOC 2006 in terms of area under overlap-recall curves (AUC). We generally outperform [19] and in some categories come close to state-of-the-art [6]. Our method, however, is at least by a factor of 57 faster.

a) VOC 2007, AP	Bird	Dog	Plant	Boat	Sheep	Cat	Chair	Table	Cow	Bottle	Aeroplane	Sofa	TV/Mon.	Person	Train	Motorbike	Bus	Horse	Car	Bicycle	Mean
LP 20 Haar 1k	10.1	11.3	10.0	11.4	15.6	20.3	14.3	24.7	16.9	11.5	29.7	30.2	27.3	29.6	41.1	39.6	43.3	52.9	39.7	39.7	26.0
LP 20 Haar 3k	10.4	9.8	10.4	11.8	16.2	20.1	17.0	26.2	17.8	15.7	30.8	32.7	33.8	33.5	42.7	41.5	44.7	54.4	43.1	46.0	28.0
[4]	9.9	11.1	12.5	15.0	17.8	19.3	22.6	23.2	24.2	25.4	28.9	34.3	41.8	42.3	44.9	47.9	50.4	56.8	58.1	58.8	32.3
[9]	10.3	12.4	9.9	12.5	18.8	19.1	17.9	23.6	22.9	14.5	31.6	33.2	41.3	29.1	39.7	44.8	44.8	50.9	50.1	52.7	29.0
[20]	0.2	13.2	1.7	0.6	13.0	17.6	3.3	12.2	10.8	9.1	17.3	15.3	18.4	14.2	22.5	27.9	27.6	36.8	29.3	22.4	15.7
[10] (2k windows)	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	22.4

b) VOC 2006, AUC	Bicycle	Bus	Car	Cat	Cow	Dog	Horse	Motorbike	Person	Sheep	Mean
LP 20 Haar 1k	68.5	64.6	52.9	74.9	59.1	71.7	67.7	67.1	51.7	51.5	63.0
LP 20 Haar 3k	73.2	69.1	58.4	78.6	64.0	75.8	72.3	71.7	57.7	56.4	67.7
[6]	70.7	70.6	66.6	73.2	69.9	71.1	71.1	72.8	65.5	67.9	69.9
[19]	62.4	58.8	49.6	76.7	52.5	71.8	63.7	63.4	41.7	44.2	58.5

time. The approach also achieves competitive results compared to state-of-the-art approaches for proposal generation, but at significantly lower runtime cost. In further work, we would like to test the performance of our method with different features and other classifiers as well as to combine it with complementary methods for reducing the runtime, like cascading.

Acknowledgement. This work was partially supported by the EU projects FP7-ICT-24314 Interactive Urban Robot (IURO) and FP7-ICT-248873 Robotic ADaptation to Humans Adapting to Robots (RADHAR).

References

1. Everingham, M., Van Gool, L., Williams, C., Winn, J., Zisserman, A.: The pascal visual object classes (VOC) challenge. IJCV **88** (2010) 303–338
2. Dalal, N., Triggs, B.: Histograms of oriented gradients for human detection. In: CVPR. (2005)
3. Viola, P., Jones, M.: Robust real-time face detection. IJCV **57** (2004) 137–154
4. Felzenszwalb, P., Girshick, R., McAllester, D., Ramanan, D.: Object detection with discriminatively trained part based models. TPAMI **32** (2010)
5. Lampert, C., Blaschko, M., Hofmann, T.: Efficient Subwindow Search: A Branch and Bound Framework for Object Localization. TPAMI **31** (2009) 2129–2142
6. Zhang, Z., Warrell, J., Torr, P.: Proposal generation for object detection using cascaded ranking SVMs. In: CVPR. (2011)
7. Gualdi, G., Prati, A., Cucchiara, R.: Multi-stage sampling with boosting cascades for pedestrian detection in images and videos. In: ECCV. (2010)
8. Rahtu, E., Kannala, J., Blaschko, M.: Learning a category independent object detection cascade. In: ICCV. (2011)
9. van de Sande, K., Uijlings, J., Gevers, T., Smeulders, A.: Segmentation as selective search for object recognition. In: ICCV. (2011)
10. Alexe, B., Thomas, D., Ferrari, V.: What is an object? In: CVPR. (2010)
11. Dollár, P., Wojek, C., Schiele, B., Perona, P.: Pedestrian detection: An evaluation of the state of the art. In: TPAMI. (2011)
12. Zhu, L., Chen, Y., Yuille, A., Freeman, W.: Latent hierarchical structural learning for object detection. In: CVPR. (2010)

13. Pedersoli, M., Vedaldi, A., Gonzàlez: A coarse-to-fine approach for fast deformable object detection. In: CVPR. (2011)
14. Romdhani, S., Torr, P., Schölkopf, B., Blake, A.: Computationally efficient face detection. In: ICCV. (2001)
15. Brubaker, S., Mullin, M., Rehg, J.: Towards optimal training of cascaded detectors. In: ECCV. (2006)
16. Zhang, W., Zelinsky, G., Samaras, D.: Real-time accurate object detection using multiple resolutions. In: ICCV. (2007)
17. Felzenszwalb, P., Girshick, R., McAllester, D.: Cascade object detection with deformable part models. In: CVPR. (2010)
18. Dollár, P., Tu, Z., Perona, P., Belongie, S.: Integral Channel Features. In: BMVC. (2009)
19. Lampert, C.: An efficient divide-and-conquer cascade for nonlinear object detection. In: CVPR. (2010)
20. Lehmann, A., Gehler, P., Van Gool, L.: Branch & rank: Non-linear object detection. In: BMVC. (2011)
21. Vedaldi, A., Gulshan, V., Varma, M., Zisserman, A.: Multiple kernels for object detection. In: ICCV. (2009)
22. Chum, O., Zisserman, A.: An exemplar model for learning object classes. In: CVPR. (2007)
23. Russakovsky, O., Ng, A.: A steiner tree approach to efficient object detection. In: CVPR. (2010)
24. Hoiem, D., Efros, A., Hebert, M.: Putting objects in perspective. IJCV **80** (2008)
25. Torralba, A., Murphy, K., Freeman, W.: Using the forest to see the trees: exploiting context for visual object detection and localization. Commun. ACM **53** (2010) 107–114
26. Desai, C., Ramanan, D., Fowlkes, C.: Discriminative models for multi-class object layout. In: ICCV. (2009)
27. Divvala, S., Hoiem, D., Hays, J., Efros, A., Hebert, M.: An empirical study of context in object detection. In: CVPR. (2009)
28. Sadeghi, M., Farhadi, A.: Recognition using visual phrases. In: CVPR. (2011)
29. Li, C., Parikh, D., Chen, T.: Extracting adaptive contextual cues from unlabeled regions. In: ICCV. (2011)
30. Breiman, L.: Random forests. Machine Learning **45** (2001) 5–32
31. Criminisi, A., Shotton, J., Robertson, D., Konukoglu, E.: Regression forests for efficient anatomy detection and localization in ct studies. In: Medical Computer Vision Workshop. (2010)
32. Gall, J., Yao, A., Razavi, N., Van Gool, L., Lempitsky, V.S.: Hough forests for object detection, tracking, and action recognition. TPAMI **33** (2011) 2188–2202
33. Fanelli, G., Gall, J., Van Gool, L.: Real time head pose estimation with random regression forests. In: CVPR. (2011)
34. Girshick, R., Shotton, J., Kohli, P., Criminisi, A., Fitzgibbon, A.: Efficient regression of general-activity human poses from depth images. ICCV (2011)
35. Leibe, B., Cornelis, N., Cornelis, K., Van Gool, L.: Dynamic 3d scene analysis from a moving vehicle. In: CVPR. (2007)
36. Dollár, P., Tu, Z., Tao, H., Belongie, S.: Feature mining for image classification. In: CVPR. (2007)
37. Crow, F.: Summed-area tables for texture mapping. SIGGRAPH Comput. Graph. **18** (1984) 207–212
38. Judd, T., Ehinger, K., Durand, F., Torralba, A.: Learning to predict where humans look. In: ICCV. (2009)