JMLR: Workshop and Conference Proceedings 29:1–15, 2014 ACML 2014

Efficient Sample Mining for Object Detection

Olivier Canévet OLIVIER.CANEVET@IDIAP.CH
François Fleuret FRANCOIS.FLEURET@IDIAP.CH
Computer Vision and Learning group, Idiap Research Institute, Martigny, Switzerland
École Polytechnique Fédérale de Lausanne, Lausanne, Switzerland

Abstract

Object detectors based on the sliding window technique are usually trained in two successive steps: first, an initial classifier is trained on a population of positive samples (*i.e.* images of the object to detect) and negative samples randomly extracted from scenes which do not contain the object to detect. Then, the scenes are scanned with that initial classifier to enrich the initial set with negative samples incorrectly classified as positive. This *bootstrapping* process provides the learning algorithm with "hard" samples, which help to improve the decision boundary.

Little work has been done on how to efficiently enrich the training set. While the standard bootstrapping approach densely visits the scenes, we propose to evaluate which regions of scenes can be discarded without any further computation to concentrate the search on promising areas.

We apply our method to two standard object detection settings, pedestrian and face detection, and show that it provides a multi-fold speed up.

Keywords: Computer Vision, Object Detection, Hard Mining, Bootstrapping

1. Introduction

With many real world applications such as surveillance, traffic analysis or content based indexing, object detection remains one of the core problems in computer vision. Sliding-window detectors are based on a binary classifier that scans the image at all possible scales and locations, and outputs a score related to the presence or absence of the object of interest. Multiple detections are then removed by non-maxima suppression to output the final detections for the test image.

This binary classifier is usually trained in two steps, both using positives samples (*i.e.* images of the object) and negative samples (*i.e.* object-free images). In the first step, negative examples are randomly extracted out of scenes that do not contain the object of interest. The training produces a classifier f_1 which already discriminates well between objects and background, but still raises too many false alarms to be used.

The second step is based on a *bootstrapping* procedure, or hard mining, which consists of augmenting the training set with "hard" negative samples. A hard negative sample x is a negative sample on which the binary classifier f_1 has a strongly positive response, or in the case of SVM, that lies inside the margin of the predictor ($f_1(x) \geq -1$, Felzenszwalb et al. 2010b). Object-free scenes are thus densely parsed until enough samples have been found and a second classifier f_2 is trained on this augmented set.

Many works over the past years have focused on proposing new features to improve the performance of the final detector. Haar-wavelets proposed by Viola and Jones (2001) have shown to be extremely efficient combined with AdaBoost to detect faces in images. Histograms of Oriented Gradients (HOGs, Dalal and Triggs 2005) are currently used by almost all state-of-the-art detectors. Walk et al. (2010) proposed features based on the optical flow in videos and on the self-similarity between color channels. Integral or aggregated channel features (Dollár et al. (2009); Dollár et al. (2014) are currently the best and fastest for pedestrian detection when combined with 2-depth trees.

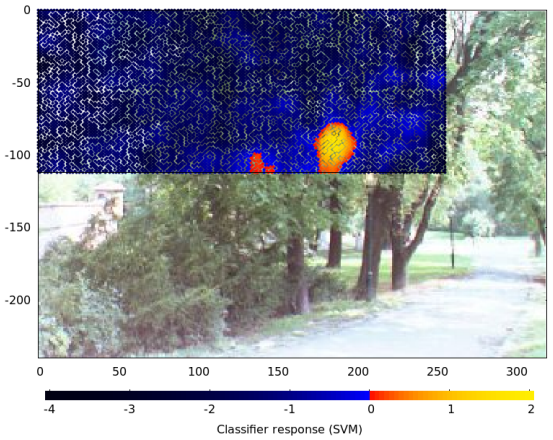
Other works have concentrated on accelerating detection at test time. Making use of cascades (Viola and Jones, 2001; Bourdev and Brandt, 2005; Felzenszwalb et al., 2010a) speeds up detection dramatically by discarding many windows in the early processing steps. Pedersoli et al. (2011) proposed a coarse-to-fine approach that reduces the number of locations to visit by discarding areas at smaller scales that will not be tested at larger scales. Gualdi et al. (2010) introduced a probabilistic framework that exploits the responses of the cascade to efficiently span the image at test time. Regarding deformable part based models, Dubout and Fleuret (2012) have proposed a Fourier-based approach to compute efficiently the convolutions of the parts with the image in the Fourier domain. The fastest detectors are now able to process around 30 frames per second which makes them practical for real-time applications.

The recent work of Henriques et al. (2013) shows that in the particular case of linear classifiers, it is possible to train with the virtual, fully translated versions of the negative samples, because the Gram matrix of the translated samples is block circulant and can be factorized in an efficient way removing redundancies between samples. They thus show that they can avoid several steps of bootstrapping and yet achieve similar final performances.

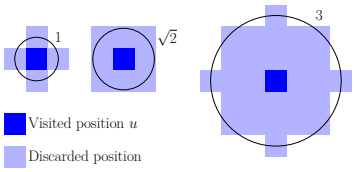
Bootstrapping is used in all the cited works (except the last one) independently of the nature of the classifier (SVM, Boosting), of the detection framework (cascade or not) or of the features. It allows the process to concentrate computation on the most difficult samples and yields a usable sliding-window detector. Dalal and Triggs (2005) use one step of bootstrapping but add all the false positives that are found. Dollár et al. (2009) use 2 of them while other works use more (Walk et al., 2010; Dollár; Dollár et al., 2014) and can go up to 10 (Felzenszwalb et al., 2010b) but limit themselves to a few thousands samples added each time. When training cascades, bootstrapping is used at each new level of the cascade (Viola and Jones, 2001).

Despite the popularity of this approach, little work has been made on how to efficiently parse scenes to find hard samples during the bootstrapping step. The traditional approach sees bootstrapping as a *particular case of detection* in which one uses the non-mature classifier f_1 to increase the training set with difficult samples for f_1. But this cannot scale to huge databases containing millions of images because it inherently requires to test all images at all positions and scales. We propose to still visit images uniformly, but to leverage local consistency of f_1 in the image plan to limit computation to a sparse subset of locations. This should lead to a set of false positives more representative than the one we would obtain by scanning a few images densely.

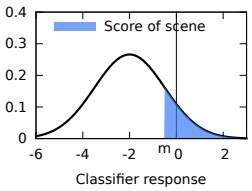
Figure 1(a) shows the responses of a pedestrian detector every other location in the image. The responses suggest a smoothness across the image and that given a response at some position, we can infer the responses of the classifier in its neighborhood.



(a) SVM responses



(b) Various discarding radii



(c) Erf score

Figure 1: Initial pedestrian detector responses (HOG features) — The detector was applied every two pixels. One point on the image corresponds to the top left corner of the 128×64 sliding window. The color of the points corresponds to the value of the detector response. Figure 1(b) shows the neighborhoods which are discarded around a visited location. Figure 1(c) shows how we estimate the score of a scene with *erf* sampling. We model the responses of a scene with a Gaussian model and we estimate the remaining number of false positive samples ($\hat{p}(f_1(x) \geq m)$).

In this context, we propose a simple yet efficient strategy that estimates the largest neighborhood that can be discarded around a position and that does not contain a hard sample. This is faster than the traditional approach because it discards lots of positions, but it is also as efficient because it finds as many hard samples.

Our contribution is three-fold:

- We propose a novel strategy to discard a neighborhood around a sample x based on $f_1(x)$ without additional computation (see § 2.1).
- We propose several methods to sample sequentially the scenes to process (see § 2.2).
- We test our method on pedestrian and face detection and show that it is able to find as many hard samples as the baseline while being significantly faster (see § 3).

2. Proposed strategy

Our purpose is to efficiently build a subset $\mathcal{H}$ of the full set of hard samples $\mathcal{H}^*$. At this point, we assume that we have an initial binary classifier f_1, trained as described in the previous section, and a series of object-free scenes S_n, $n = 1, \ldots, N$. A sample x is considered hard if $f_1(x) \geq m$, where $m \in \mathbb{R}$ is fixed.

Given that we have already selected a series of hard samples, to select a new one, we first choose a scene S_n and then a location $u \in S_n \setminus V_n$, where V_n is the subset of discarded locations in S_n. We extract the sample x at location u and compute $f_1(x)$.

Scene	A scene is an image at a given scale in which negative samples are extracted
$\mathscr{L}$	Set of all the possible sample locations in the scene.
f_1	The initial classifier.
f_2	The classifier trained with the augmented set
x	A $h \times w$ sample on which we can apply f_1 or f_2
m	A threshold above which a sample is considered "hard".
"hard" sample	A sample such that $f_1(x) \geq m$.
S_n	An object-free scene used to find negative and "hard" samples.
V_n	The set of already visited or discarded positions in S_n.
Exhausted	A scene is exhausted when all the locations have been visited or discarded

Table 1: Notations and designations

If $f_1(x) < m$ (*i.e.* x is a true negative), we add u to V_n along with all locations within a circle of radius r^- centered at u, without actually computing f_1 on them. We call this discarding the neighborhood (*i.e.* we will not select these positions in the future). As explained in the following sections, depending on the method we consider, the radius r^- may be a constant, or a function of the response $f_1(x)$. Figure 1(*b*) shows examples of discarded regions for various radii. When all positions in S_n have been either visited or discarded, we say that the scene *exhausted*.

If $f_1(x) \geq m$ (*i.e.* x is a false positive), we add x to $\mathcal{H}$, u to V_n and we discard a small neighborhood of radius r^+ around the false positive: we thus discard positions of redundant samples. In all the methods we present, r^+ is constant.

Finally we repeat the procedure by choosing a scene and a position until we have found enough hard samples or until all the scenes are exhausted.

It is important to note that we do not exhaust a scene before moving to the next scene. Rather, we pick one scene and a location at a time, so that we gradually visit them all. With small data-sets, this is unimportant, especially if the purpose is to exhaust all the scenes (*i.e.* visit all locations). With very large data-sets however, it is imperative to enforce the sampling to be distributed over all the scenes, to avoid generating a set of samples of limited diversity.

We now describe the strategy we propose to parse the scenes efficiently: the key idea is to learn the discarding radius r^- as a function of $f_1(x)$. We illustrate the rest of the section with pictures from the pedestrian INRIA data-set.

2.1. Strategy to discard non-visited locations in a scene

The novel methods we propose in the next sections rely on the relation between the point-wise response of the classifier at a certain location in the scene, and the presence of hard samples in its neighborhood.

Without loss of generality, we assume that all the scenes have the same size and we define the lattice of extractible positions $\mathscr{L} = \{1, \dots, H\} \times \{1, \dots, W\}$[1]. A scene S is a random variable on $[\![0, 255]\!]^{\mathscr{L}^3}$.

For simplicity, given a position $u \in \mathscr{L}$ and a classifier f, we will use the notation $f(u)$ for the response of f on the sample extracted at location u in the scene.

Let U be a position sampled uniformly on $\mathscr{L}$, and R the distance between U and the closest false positive, that is

$$R = \min_{\substack{v \in \mathscr{L} \\ f_1(v) \geq m}} \|v - U\|_2$$

Using $f_1(U)$, our purpose is to predict the *largest* neighborhood that we can discard *without* discarding positions that actually contain hard samples, that is finding a lower bound on the random variable

$$R \mid f_1(U).$$

Figure 2 shows the empirical distribution of $(f_1(U), R)$. We see that for a given negative response $f_1(x)$, there seems to be a non-zero lower bound: the closest false positive from u tends to be farther from a true negative x as $f_1(x)$ increases. By learning this lower bound, we can therefore discard a circular area around u with an *adaptive* radius $r^- = r^-(f_1(x))$.

The standard way to estimate the lower bound of the distribution is to extract the points near the bound and fit a model on them. Many models can be used as well as many ways to learn it: linear or quadratic model, step function, etc.

As we observe on the two data-sets we used, the lower bound is properly modeled as an affine function of the classifier's response. This is the model we use for the *adaptive discarding radius*:

$$r^-(f_1(x)) = -a \times (f_1(x) - m), \quad a \geq 0,$$

where m is the threshold above which x is considered hard. Before building $\mathcal{H}$, we draw the empirical distribution by selecting a few 30×30 sub-lattices out of hundred scenes and compute the distance to the closest hard sample at every location. We then estimate the optimal a^* by averaging over the 10 lowest values of $\frac{r(f_1(x))}{f_1(x)-m}$.

2.2. Strategy to select a scene

In the works of Dalal and Triggs (2005) and Viola and Jones (2001), scenes are visited at all locations either until it no longer fits in memory or to keep a reasonable training time.

However, with the large increase of the data-sets, it becomes impractical to exhaust all scenes. We here present different methods to choose the next scene to visit.

- **Uniform sampling** consists in choosing a scene S_n uniformly with replacement in the subset of non-exhausted scenes, that is with the distribution:

$$\forall\, 1 \leqslant n \leqslant N, \quad p(S_n) = \frac{1}{N}$$

1. The size of the lattice is actually not the size of the scene: if we extract samples of size $h \times w$ in a $H_S \times W_S$ scene, the extractible region is $(H_S - h + 1) \times (W_S - w + 1)$

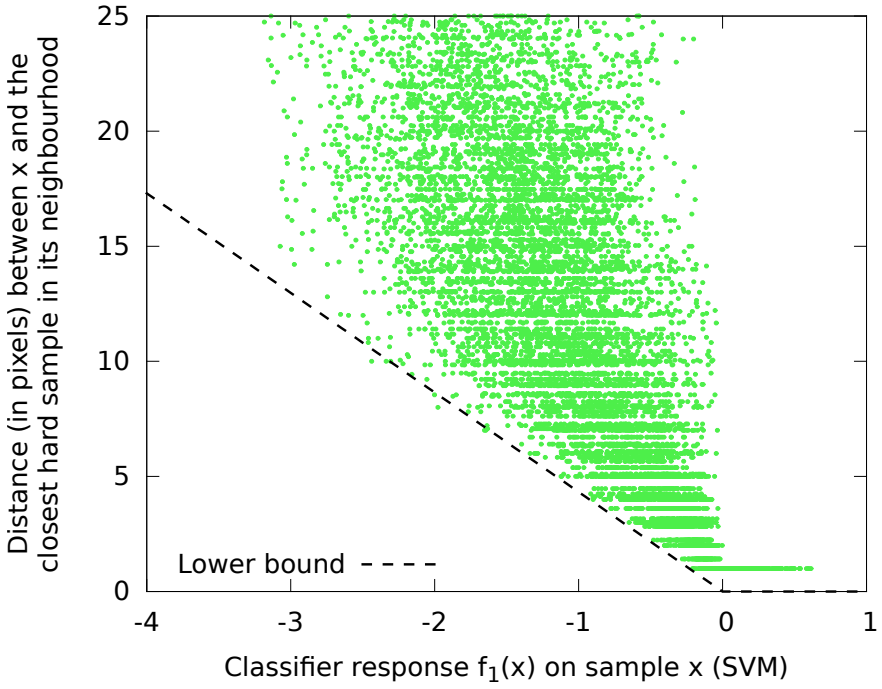


Figure 2: Empirical distribution of $(f_1(U), R)$ as defined in 2.1 with $m = 0$ — One green dot $(f_1(u), r)$ represents the distance $r = ||v - u||_2$ of the closest false positive (at location v) from sample x (at location u) as a function of the response $f_1(x)$ of the initial classifier. For instance, given a true negative sample x^- such that $f_1(x^-) = -2$, we are assured that the closest false positive is farther than ~ 8 pixels, which allows us to discard a disc of radius 8 around u, that is ~ 200 locations that will never be chosen. The horizontal line for $f_1(x) \geq 0$ indicates that empirically, any false positive has a false positive neighbor.

- **Size sampling** consists in choosing a non-exhausted scene according to its size, that is with the distribution

$$\forall\, 1 \leqslant n \leqslant N, \quad p(S_n) = \frac{1}{Z}|S_n|,$$

where Z is a normalization constant and $|S_n|$ the number of extractible positions in the lattice of S_n.

- **Unexplored sampling** consists in choosing a non-exhausted scene according to its unexplored area, that is with the distribution

$$\forall\, 1 \leqslant n \leqslant N, \quad p(S_n) = \frac{1}{Z}(|S_n| - |V_n|),$$

where $|V_n|$ is the number of discarded *or* visited positions in scene S_n.

- ***Erf* sampling** consists in choosing a non-exhausted scene according to the classifier responses on already visited locations in the scene. Let μ_n and σ_n^2 be the mean and variance of the classifier responses in scene S_n at some point in the process. We sample according to the distribution

$$\forall\, 1 \leqslant n \leqslant N, \quad p(S_n) = \frac{1}{Z}\min\left(E_n, \frac{1}{\pi r^{+2}}\right)(|S_n| - |V_n|),$$

where

$$\forall\, 1 \leqslant n \leqslant N, \quad E_n = \int_m^{+\infty} \mathcal{N}(\mu_n, \sigma_n) = \operatorname{erfc}\left(\frac{x - \mu_n}{\sqrt{2}\sigma_n}\right), \quad \text{see } 1(c)$$

This sampling is designed to select a promising scene, that is one which has already provided false positives will have a larger score. The min provides an upper-bound on the score to avoid having too large values of E_n. The constant $1/\pi r^{+2}$ corresponds to the number of samples we would collect if *all* the remaining positions were actually false positives.

2.3. Implementation detail for sampling

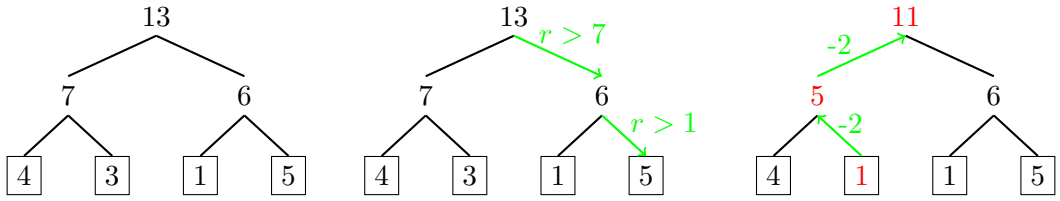
We may need to sample several million times, which can be computationaly prohibitive if done in a naive way. We here provide details of the implementation to select the scene in which to extract the sample.

All sampling methods except the uniform strategy require to sample according to positive weights (image size, unexplored area, or erf score) normalized into a discrete distribution that may evolve in time (for unexplored area and erf score). The basic approach for sampling according to a discrete distribution of size N consists of building the cumulative distribution, drawing a number in $[0, 1]$ and computing the interval in which it falls. However, it has $\mathcal{O}(N)$ complexity, which would be too expensive in our framework.

Moreover we need an efficient way to update the distribution. This rules out the alias sampling method introduced by Walker (1974): Although this methods allows to sample according to a discrete distribution at constant time after a pre-processing step of $\mathcal{O}(N)$,

which reorganizes the distribution in equal-size bins of 2 elements, it does not provide a fast way to update the distribution. The pre-processing step would have to be performed after each update which in our case would also be intractable.

Sampling among N items using a (balanced) binary tree costs $\mathcal{O}(\log N)$ both in sampling and updating the distribution. Figure 3 shows an example for the following distribution $[4, 3, 1, 5]/13$. Given the distribution (in our case, it represents the scores of the scenes), a binary tree is build such that the leaves contain the scores of the images and each internal node have the sum of both its children (see 3(a)). To sample according to this distribution, one starts from the root and recursively draws a number r between 0 and the value of the current node: if r is smaller than the left child, it restarts the sampling with the left child. Otherwise it chooses the right child (see 2.3). The process goes on until reaching a leaf. To update a particular leaf, one just has to update all the nodes from the leaf up to the root (see 3(c)), which also costs $\mathcal{O}(\log N)$.



(a) Sampling tree (b) Sample (c) Updating node 2

Figure 3: Sampling according to a changing discrete distribution — Given a binary tree build on top of a discrete distribution $[4, 3, 1, 5]/13$ (3(a)), we can draw a sample according to this distribution by recursively sampling according to the partial distribution of the children at a given node (2.3). Only the nodes between the leaf to update and the root need to be updated (2.3). Both operations have $\mathcal{O}(\log N)$ complexity.

3. Experiments

We now apply our strategy to train a pedestrian and a face detector. We compare how many false positives are actually found with respect to the method that does not discard a neighborhood and we compare the strategies to select scenes. A hard sample is defined with $m = 0$.

The computational cost of bootstrapping is proportional to the number of visited locations since the time to choose a scene (implemented with a binary tree), then a position, then to add the neighborhood to V_n is negligible in comparison to computing $f_1(x)$. Hence, we report the results as functions of the number of visited locations. The positive discarding radius r^+ is set to 0 for the faces and to 2 for pedestrians.

3.1. Data-sets

We use the INRIA pedestrian data-set (Dalal and Triggs, 2005) to train the pedestrian detector. It consists of 2,416 128×64 images of pedestrians and 1,218 pedestrians-free

scenes for training. The testing set consists of 1,132 pedestrians and 453 scenes. We train the initial detector by using 10,000 128×64 negative images randomly extracted from the training scenes. We use HOG as features and we use the SVM library LIBLINEAR (Fan et al., 2008) with a ℓ_2-regularized ℓ_2-loss.

For faces, we use the standard face data-set of Viola and Jones for training and test it on the MIT-CMU data-set. As negative scenes, we used the Caltech Background data-set (Weber) which provides 900 face-free images (we removed 3 images containing a skull or a baby). We use AdaBoost (Freund and Schapire, 1995) as the learning algorithm and Haar-like features (Viola and Jones, 2001).

3.2. Number of hard samples found

Although our strategy is designed to discard the largest expected neighborhood that does not contain any false positives, some of them might actually be in this discarded area. Moreover we compare our *learnt radius* as described in 2.1 with a manually chosen radius, *i.e.* using $a \gg a^*$ (resp. $a \ll a^*$), that is with a more aggressive (resp. smoother) slope. We therefore compare our strategy to *exhaust* all the scenes with:

- Parsing densely all the scenes *without* discarding any neighborhood (*i.e.* discarding with a radius $r^- = 0$) (red circle with label 0 on figure 4).
- Parsing densely all the scenes discarding a constant radius around true negatives, *i.e.* disregarding $f_1(x)$ (red circles on figure 4).
- Parsing densely all the scenes discarding a manually selected adaptive radius, *i.e.* $r^- = -af_1(x),\ a \neq a^*$ (blue dots on figure 4)

Figure 4 shows the results of the experiment: we plot the number of false positives found as a function of the number of visited locations across all the scenes (recall that we exhaust all scenes).

The plateau formed by the blue dots corresponds to the maximum number of false positives we can actually find in the scenes. With a constant radius (red circles) the parsing discards the false positives which are neighbors of true negatives and as a result finds less hard samples than an adaptive radius.

An adaptive radius does not discard false positives next to negative samples because $r^- = -af_1(x) < 1$. For both data-sets, our simple method finds a learnt adaptive radius which lies in the area where one visits the fewer positions while finding almost all the available hard samples. We argue that any model or estimation method that leads to such ratio between number found and number visited is sufficient to have a significant reduction in the number of visited locations. In this experiment, we visit 30 times (resp. 4) fewer locations on the INRIA data-set (resp. Caltech background).

3.3. Performances

Our ultimate objective is to produce the best detector we can. While the number of samples used for training is an important parameter, it is not the only one. Diversity, and more generally the "representativity" of the sample set, is key. An extreme example would be

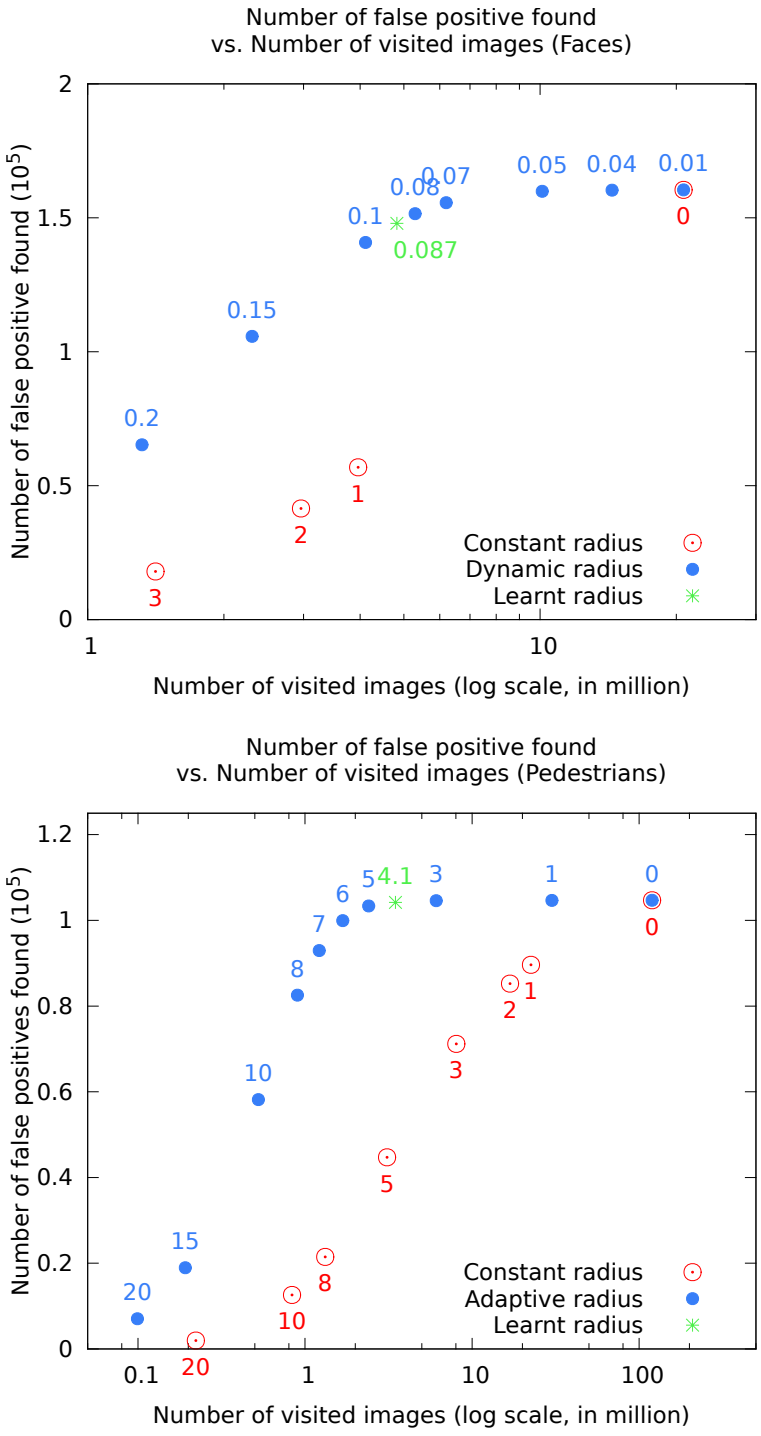
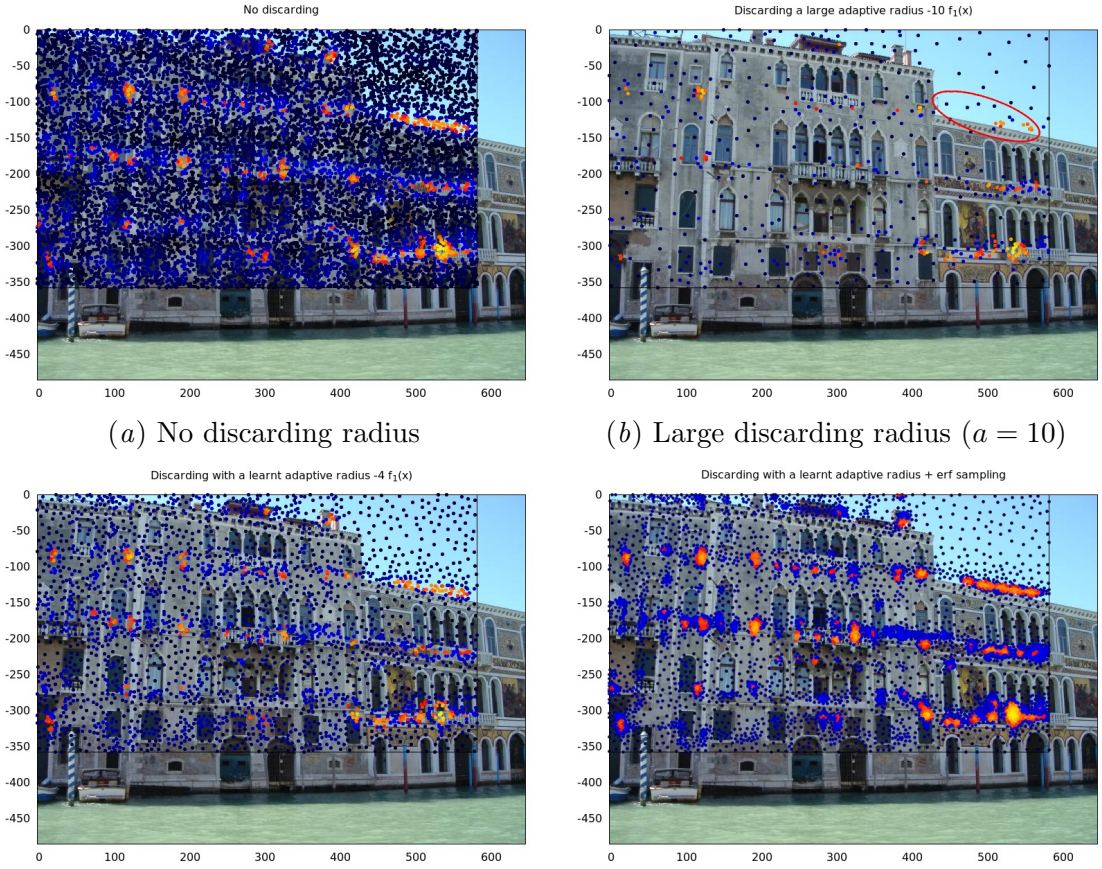


Figure 4: Number of false positives found vs. the number of visited samples for the different strategies (average over 10 runs) — In the experiments, we visite the scenes until *all* are exhausted. The red circles correspond to a strategy which discards a disc of constant radius around a visited sample x, the blue dots to the strategy which uses an adaptive radius $-af_1(x)$. The learnt radius (green star) is in the area where almost all the available false positives are found while visiting a minimum of positions.



(a) No discarding radius

(b) Large discarding radius ($a = 10$)

(c) Learnt discarding radius ($a \simeq 4$)

(d) Erf sampling + learnt discarding radius

Figure 5: Comparison of the visited locations for different strategies — One dot represents the top left corner of the extracted window $a = 10$ tends to discard too wide regions that causes some small areas of false positives to be discarded.

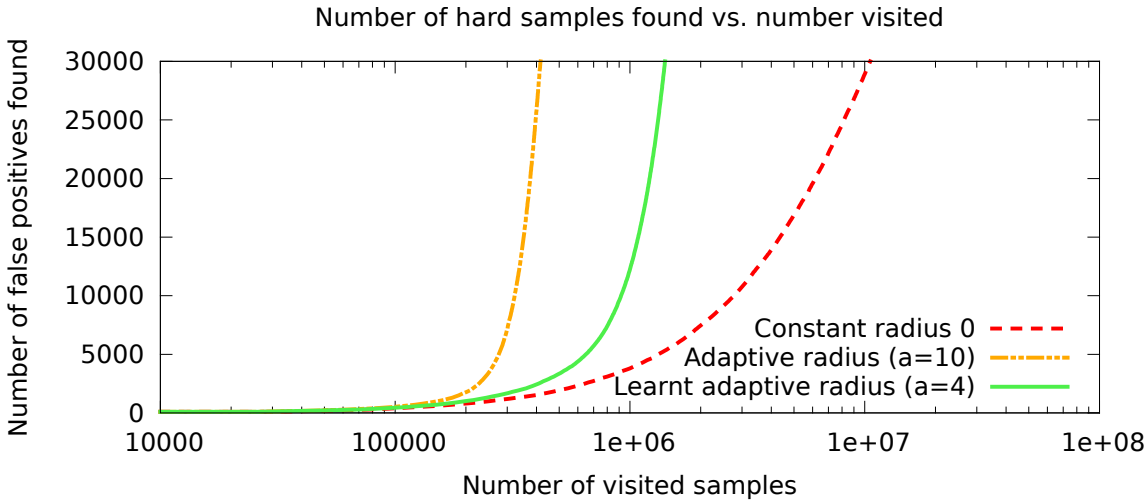


Figure 6: Number of false positives found as a function of the number of visited locations (log scale) — The curves show that in the beginning of the procedure, all three methods find the same number of false positive samples but that after a certain point, the adaptive radii find more. The process has discarded lots of areas and only the false positives remain.

a very large set composed of the same example replicated with very minor pixel noise variation.

Regarding the collection of samples from scenes, if we were only interested in the sheer number of hard samples we collect, we could simply mine exhaustively in the neighborhood of any hard sample we find, since it is often the case that hard samples come in dense clusters.

To assess the efficiency of our method with this aspect taken into account, we must estimate not only the computational cost to mine K false positives, but also how good they are for training. To that purpose, we compare the various strategies after one round of bootstrapping adding $K = 30,000$ hard samples (figure 7).

The left plot of figure 7 presents the performance of the pedestrian detector and figure 5 shows the visited location in a particular scene for various methods.

The red dashed curve is the baseline which does not discard any neighborhood (r^-=0). It needs to sample 10 million positions to find 30,000 hard samples. As we see on figure 5(a), lots of locations are visited. Our proposed strategy (green solid line) needs to sample 6 times fewer positions and yet achieves the same accuracy. We have not noticed much difference between sampling a scene according to its size or to its unexplored area so we report the results for size only. We see on figure 5(c) that our method discards larger neighborhoods around true negatives with large negative response f_1 (dark points) and concentrates its sampling on the areas where the false positives lie.

Our approach with uniform sampling performs slightly worse than the size or unexplored because it unbalances the sampling: small scenes have equal probability of being chosen so $\mathcal{H}$ contains more samples from small scenes.

A too aggressive slope ($a = 10$ here whereas our learnt slope is $\simeq 4$) performs also worse. Discarding too large areas tends to discard plenty of false positives as we can see on figure 5(b). Some "small islands" of false positives were discarded (red circled area). Large "islands" of false positives then tend to be over-sampled that yields a set of redundant samples, as mentioned at the beginning of 3.3.

Erf sampling performs also worse because it tends to select the same scenes so $\mathcal{H}$ is unbalanced (on 5(d) the scene is more visited than if it is sampled according to its size). Without a bound on the erf score, *i.e.* without $1/\pi r^{+2}$, the performance dramatically drops because the hard samples come from very few scenes.

Finding hard samples goes much faster as we visit more locations because we discard more areas 6. This is why is goes 6 times faster when we stop at 30,000 hard samples found (current experiment), and 30 times faster when we look for all of them (right plot of figure 4).

The right plot of figure 7 presents the performance of the face detector. All strategies give the same performance but an adaptive radius reduces considerably the number of visited locations and thus the time.

4. Conclusion

We have presented a simple strategy to mine scenes to find hard samples for object detection. It consists in learning an adaptive radius to discard a neighborhood around a true negative sample as a function of the initial classifier response that avoid to visit these locations in further steps of the parsing. Selecting a scene according to its size has also showed to yield better results.

We have shown that this strategy allows to find as many hard samples as the naive approach that visit all possible locations an order of magnitude faster while achieving similar performances.

Further work will aim at explicitly taking into account the diversity of the samples as estimated through the classifier space. Instead of enforcing this diversity implicitly by choosing one scene selection strategy instead of another, it may be possible to adaptively discard a region not only because we have exhausted its false-positives, but also because we estimate that the remaining ones are redundant with some that we have already sampled.

Acknowledgments

Olivier Canévet was supported by the Swiss National Science Foundation under grant 200021-140912 – DASH

References

Lubomir Bourdev and Jonathan Brandt. Robust object detection via soft cascade. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 2, pages 236–243. IEEE, 2005.

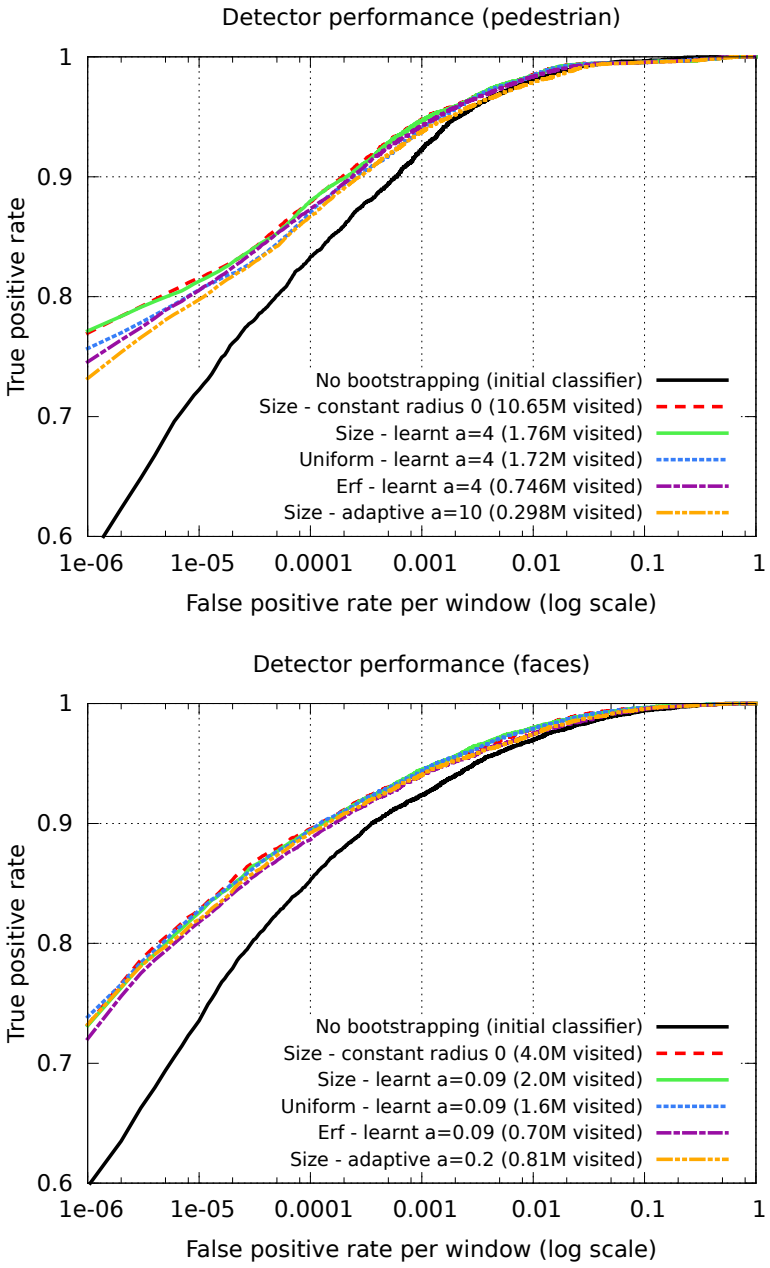


Figure 7: Comparison of the performances on the two test data-sets for different sampling and discarding strategies (averaged over 20 runs) — The initial detector (black curve) is trained as described in section 3.1. Choosing a scene according to its *size* and using the learnt adaptive radius yields better performances than *rand*, *erf* or a more aggressive slope. It also visits 6 times (resp. 2) less samples than the traditional approach and reaches similar performances. We note that for faces, the selection strategy does not matter.

Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 1, pages 886–893. IEEE, 2005.

P. Dollár, Z. Tu, P. Perona, and S. Belongie. Integral channel features. In *BMVC*, 2009.

Piotr Dollár. Piotr's Image and Video Matlab Toolbox (PMT). http://vision.ucsd.edu/~pdollar/toolbox/doc/index.html.

Piotr Dollár, Ron Appel, Serge Belongie, and Pietro Perona. Fast feature pyramids for object detection. 2014.

Charles Dubout and François Fleuret. Exact acceleration of linear object detectors. In *Computer Vision–ECCV 2012*, pages 301–311. Springer, 2012.

Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. LIBLINEAR: A library for large linear classification. *Journal of Machine Learning Research*, 9:1871–1874, 2008.

Pedro F Felzenszwalb, Ross B Girshick, and David McAllester. Cascade object detection with deformable part models. In *Computer vision and pattern recognition (CVPR), 2010 IEEE conference on*, pages 2241–2248. IEEE, 2010a.

Pedro F Felzenszwalb, Ross B Girshick, David McAllester, and Deva Ramanan. Object detection with discriminatively trained part-based models. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 32(9):1627–1645, 2010b.

Yoav Freund and Robert E Schapire. A desicion-theoretic generalization of on-line learning and an application to boosting. In *Computational learning theory*, pages 23–37. Springer, 1995.

Giovanni Gualdi, Andrea Prati, and Rita Cucchiara. Multi-stage sampling with boosting cascades for pedestrian detection in images and videos. In *Computer Vision–ECCV 2010*, pages 196–209. Springer, 2010.

J.F. Henriques, J. Carreira, R. Caseiro, and J. Batista. Beyond hard negative mining: Efficient detector learning via block-circulant decomposition. In *proceedings of the IEEE International Conference on Computer Vision*, 2013.

Marco Pedersoli, Andrea Vedaldi, and Jordi Gonzalez. A coarse-to-fine approach for fast deformable object detection. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 1353–1360. IEEE, 2011.

Paul Viola and Michael Jones. Robust real-time object detection. *International Journal of Computer Vision*, 4:34–47, 2001.

Stefan Walk, Nikodem Majer, Konrad Schindler, and Bernt Schiele. New features and insights for pedestrian detection. In *Computer vision and pattern recognition (CVPR), 2010 IEEE conference on*, pages 1030–1037. IEEE, 2010.

Alastair J Walker. New fast method for generating discrete random numbers with arbitrary frequency distributions. *Electronics Letters*, 10(8):127–128, 1974.

Markus Weber. Caltech background dataset. http://www.robots.ox.ac.uk/~vgg/data/background/background.tar.