A Multi-Stage Pedestrian Detection Using Monolithic Classifiers

Giovanni Gualdi
D.I.I. Univ. of Modena and Reggio Emilia, Italy
giovanni.gualdi@unimore.it

Andrea Prati
D.I.S.M.I. Univ. of Modena and Reggio Emilia, Italy
andrea.prati@unimore.it

Rita Cucchiara
D.I.I. Univ. of Modena and Reggio Emilia, Italy
rita.cucchiara@unimore.it

Abstract

Despite the many efforts in finding effective feature sets or accurate classifiers for people detection, few works have addressed ways for reducing the computational burden introduced by the sliding window paradigm. This paper proposes a multi-stage procedure for refining the search for pedestrians using the HOG features and the monolithic SVM classifier. The multi-stage procedure is based on particle-based estimation of pdfs and exploits the margin provided by the classifier to draw more particles on the areas where the classifier’s response is higher. This iterative algorithm achieves the same accuracy than sliding window using less particles (and thus being more efficient) and, conversely, is more accurate when configured to work at the same computational load. Experimental results on publicly available datasets demonstrate that this method, previously proposed for boosted classifiers only, can be successfully applied to monolithic classifiers.

1. Introduction and Related Works

People detection is a foundational problem in surveillance, particularly challenging in images with cluttered backgrounds and in case a good generalization across intra-class variability is required.

Typical people detection normally exploits the sliding window approach (SW hereinafter) and spans the search over the whole image in a serial fashion (e.g. [3]). With SW-based methods, all the possible patches at varying position and scale are extracted and then validated through a binary classifier. The number of windows to be evaluated ($N_{SW}$) is a function of the image size, of the sliding steps (also known as pixel strides) and of the scale search, that is regulated through a scale step (also known as scale stride). This approach has the drawback of brute force methods, that is the high computational load due to the number of windows to check, that grows in each dimension to span over: on one side the computational time is proportional to $N_{SW}$, that should be kept limited because of time constraints; on the other, the detection rate and localization accuracy decrease when the pixel and scale stride increase. As a consequence, a trade-off between efficiency and accuracy is unavoidable. Consequently, several works focus on the reduction of the computational burden, following three main streams: (a) pruning the set of hypotheses by exploiting other cues (e.g. motion [16], depth [4], geometry and perspective [10], or whatever cue that is different from the appearance cue specific of the detector itself); (b) speeding up with hardware-optimized implementations (such as GPUs [18]); (c) efficiently exploring the sub-window space through optimal solution algorithms [1, 7, 11, 14, 17, 19].

Our paper focuses on the last stream of research, where a few related works can be cited: in [11], the authors propose to bypass SW through an efficient sub-window search using a branch and bound technique. However this method has strict requirements over the quality function (i.e., classifier score) that are not met by many classifiers; additionally it detects only one object at a time (i.e., it finds the global maximum of the function), requiring multiple runs to detect multiple objects; in [17] the comparison between a SW approach (called full search by the authors) and a more efficient solution (called speed search) is proposed. Another deterministic coarse-to-fine refinement has been shown in [19], with a deterministic (grid-distributed), multi-stage (coarse-to-fine) detection: successful detections at coarse resolutions yield to refined searches at finer resolutions. Also in [14] a similar approach is presented, adding (loose) prior spatial constraints on the object locations, that limit the generality of the approach. Butko and Movellan in [1] explore the maximization of information gain through a computational approach that simulates a digital fovea. Their approach provides a speed-up w.r.t. SW, however two limitations are suffered: a slight degradation of performances and single-target detection. Conversely, the Multi-Stage detection method described by [7] appears to gain both in efficiency and accuracy w.r.t. SW and is intrinsically multi-target since it models object detection as a multi-modal pdf estimation. This data-driven search assumes that the search can start in parallel, uniformly over the image and iter-
tively focuses on the exploration of the image toward the areas where the target objects are more likely to be found: formally, they provide an incremental estimation of a likelihood function through Monte Carlo sampling, exploiting the confidence of the classifier only. In practice, the confidence is employed to increasingly draw samples on the areas where the objects are potentially present and avoiding to waste search time over other regions. The idea of iterative focusing of the attention on targets is often embraced in pattern recognition tasks, but the authors formalize the approach to work for object detection through ensembles of boosted classifiers.

We extend that proposal introducing some fundamental novelties: indeed, in our paper we demonstrate that the multi-stage method, if fed through appropriate confidence measures, can work not only on ensemble classifiers but on any real-valued confidence classifier.

To this aim, our tests use the popular pedestrian classifier proposed in [3] that apply a SVM classifier to Histogram of Oriented Gradients (HOG): this feature counts occurrences of gradient orientation in localized portions of an image and happens to be particularly suited for the task of pedestrian detection because it shows a high degree of invariance to small body movements. The computation of the histograms is performed on a dense grid of uniformly spaced cells and overlapping local contrast normalization is used for improved accuracy; the SVM classifier can either be linear or use other kernels (e.g., Radial Basis Function); in any case the returned classification margin (i.e., the distance to the class-dividing boundary) extends over \( \mathcal{R} \), making it incompatible with the Multi-Stage method in [7]: indeed, this algorithm requires the classification confidence (or margin, or response) to be in \([0, 1]\). Since several works still recently take inspiration and move forward from the detection architecture proposed by [3] ([6] is among the most successful examples), we believe that demonstrating the feasibility of the Multi-Stage detection onto HOG-SVM classifiers is of particular significance.

In this paper we aim to compare the results of the traditional SW people detection with the Multi-Stage people detection. We call the samples used in such method as particle windows (PWs) in juxtaposition to the uniformly grid-distributed sliding windows (SWs). All tests are performed on public image datasets, with ground truth too in order to facilitate the comparison by other researchers.

### 2. Multi-Stage Density Estimation for Pedestrian Detection

The goal of a multi-stage density estimation for object detection is to provide a non-uniform quantization over the state space and to model the detection as the estimation of a probability density function \( p(\mathbf{X} | \mathbf{Z}) \), where \( \mathbf{X} = (w_x, w_y, w_a) \) is the state and \( \mathbf{Z} \) corresponds to the image; specifically, the aim is estimating the modes of that pdf, where each mode corresponds to the detection of an object [7]. Let us not consider any a priori information in the image (e.g. priors on motion, geometry, scene perspective, etc.), therefore the state pdf can be assumed proportional to the measurement likelihood function, i.e. \( p(\mathbf{X} | \mathbf{Z}) \propto p(\mathbf{Z} | \mathbf{X}) \). This likelihood function is estimated by an iterative refinement based on the observations. Algorithm 1 shows the complete procedure. The initial set \( S_1 \) of PWs is grid-distributed as in SW. This set contains \( N_1 \) samples (see line 5 of the algorithm) and the key point is that this number is one order of magnitude lower than the cardinality of a typical set of SW (\( N_{SW} \)). Each particle represents a window \( w = (w_x, w_y, w_a) \).

#### Algorithm 1 Measurement Step

1: Set \( S = \emptyset, i = 1 \)
2: while true do
3: \hspace{1em} begin
4: \hspace{2em} if \( i = 1 \) then
5: \hspace{3em} Extract \( N_1 \) grid-distributed PWs
6: \hspace{3em} \( S_1 = \{ p_{w_1}^{(j)} | p_{w_1}^{(j)} \sim \text{grid distribution}, \ j = 1, \ldots , N_1 \} \)
7: \hspace{2em} else
8: \hspace{3em} end
9: \hspace{2em} Draw \( N_1 \) PWs from the measurement function \( p_{1}^{(j)}(\mathbf{X}) \):
10: \hspace{3em} \( S_i = \{ p_{w_i}^{(j)} | p_{w_i}^{(j)} \sim p_{1}^{(j)}(\mathbf{X}), \ j = 1, \ldots , N_1 \} \)
11: \hspace{2em} end
12: Assign a Gaussian kernel to each \( p_{w_i}^{(j)} \), \( j = 1, \ldots , N_1 \):
13: \( \mu_{i}^{(j)} = p_{w_i}^{(j)}; \Sigma_{i}^{(j)} = \Sigma_{i} \)
14: Compute the measurement on each \( p_{w_i}^{(j)} \):
15: \( t_{i}^{(j)} = \frac{1}{1 + \exp(a(M(\mu_{i}^{(j)}) + c))} \) with \( M \in (-\infty, +\infty) \)
16: Obtain the measurement density function at step i:
17: \( p_{i}(\mathbf{Z} | \mathbf{X}) = \sum_{j} n_{i}^{(j)} \cdot N(\mu_{i}^{(j)}, \Sigma_{i}^{(j)}) \)
18: where: \( n_{i}^{(j)} = t_{i}^{(j)} / \sum_{k=1}^{N_1} t_{k}^{(k)} \)
19: Retain only the \( p_{w_i}^{(j)} \) with successful margin M:
20: \( \hat{S}_i = \{ p_{w_i}^{(j)} \in S_i | M(\mu_{i}^{(j)}) > \text{thrs}, \ j = 1, \ldots , N_1 \} \)
21: \( S = S \cup \hat{S}_i \)
22: \( i = i + 1 \)
23: if stop condition reached then break while loop;
24: end
25: Assign Gaussian Kernel \( \epsilon \) to each \( p_{w} \in S \)
26: Run the Sequential Kernel Density Approximation [8], and obtain a Mixture of Gaussians \( \mathcal{M} \)
27: \( \mathcal{M} \) is the final likelihood function \( p(\mathbf{Z} | \mathbf{X}) \)
28: Each mode of \( \mathcal{M} \) represents an object detection

Typically, classifiers have a degree of robustness to translation and scaling, meaning that the margin \( M \) has high values not only in exact correspondence of a true positive, but also in its close neighborhood, whose size depends on the classifier, and degrades monotonically when moving further away. This behavior of the margin generates a basin of attraction around true positives, both in location and in scale.
Figure 1. Distribution of PWs across stages in the multi stage approach on a test image. The number of stages employed in this example is $m = 11$, starting with $N_1 = 2820$ and reaching $N_{11} = 1$ through an exponential decay as suggested in [7]. (a-e) show the distribution of the PWs in the first 5 stages; in (a) the particles are grid-distributed on $x$, $y$ and $scale$; circles in (f) represent the particle windows that are retained in $S$ (line 19-21 of Alg. 1), i.e. the set of particles that triggered a successful classification.

Therefore, the rationale is that part of these sparsely distributed PWs of the initial set $S_1$ will fall inside the basins of attraction of the target objects and will provide an initial rough estimation of the measurement function used to sample the PWs at the following stage; the algorithm obtains a progressive refinement of the windows displacement and a growing confidence over the measurements; all this makes possible to decrease, from stage to stage, the number of $N_i$ to sample (as visually depicted in Fig. 1). The final aim is that the total number of particle windows $N_{PW} = \sum_{i=1}^{m} N_i$ ($m$ is the index of the last stage) be definitely lower than the fixed number of windows $N_{SW}$ of the $SW$ set.

At each stage, the $N_i$ samples generate the measurement density function $p_i$ through a Kernel Density Estimation (KDE) approach with Gaussian kernel, producing a mixture of $N_i$ Gaussians (line 17): for each $j$-th component, mean, covariance and weight are defined. The mean $\mu_i^{(j)}$ is set to the $j$-th particle window value $pw_i^{(j)} = (w_x^{(j)}, w_y^{(j)}, w_s^{(j)})$; the covariance matrix $\Sigma_i^{(j)}$ is set to a covariance $\Sigma_i$ (line 13), which, at any given stage $i$, is constant for all particle windows. $\Sigma_i$ is proportional to the size of the basin of attraction of the classifier and in the following stages $\Sigma_i$ decreases: this has the effect of incrementally narrowing the samples scattering, obtaining a more and more focused search over the state space. Finally, the margin $M(pw)$ of the classifier is exploited to determine the weight $\pi_i^{(j)}$ of the $j$-th Gaussian component (line 18). Those particle windows falling on a basin of attraction, i.e., close to the mode/peak of the distribution to estimate, shall receive higher weights with respect to the others, so that the measurement function $p_i$ will drive the sampling of the next stage toward portions of the state space where the classifier yielded high margins. Conversely, sampling must not be wasted over areas with low margin of the classifier. These weights must act as attractors which guide the particle windows toward the peaks. This is accomplished by connecting the weights $\pi_i^{(j)}$ to the margin $M$ of the classifier in the sample location $\mu_i^{(j)}$ (line 15). The method in [7] makes use of ensembles of boosted classifiers only, that naturally fit a KDE architecture because their margin (or response) is in $[0, 1]$. In extending the method also to monolithic classifiers, we need to consider that the classification margin typically ranges over $\mathbb{R}$ or $\mathbb{R}^+$: in such cases KDE cannot be applied, unless an appropriate function translate the range to $[0, 1]$. To this aim hard clipping or soft clipping functions can be used [2]; we propose the use of the latter, through the implementation of a sigmoid function (defined as in line 15), that provides a smooth transition across the margins. The two function parameters $a$ and $c$ can be learned using the Platt algorithm [2] from a training set; the only constraint is $a \in (-\infty, 0)$, to make the sigmoid function monotonically increasing so that the measurement function will be drawn toward the areas of higher margin.

The behavior of this strategy is depicted in Fig. 1, where
the samples at subsequent stages decrease in number but concentrate more and more on the peaks of the distribution.

The measurement density function $p_i(Z_t|X_t)$ (line 17) is directly used to sample the PWs at the next stage, differently from [7] that proposes to combine it linearly with a proposal, that was initialized to a uniform pdf over the state space. Our proposal is to concentrate all the uniformly (grid-distributed) samples at the very first stage and, from the second stage on, just rely on the increasingly-refined measurement function $p_i$. At the end of each stage only the PWs of $S_i$ that triggered a successful object detection (i.e. $M(pw) > thrs$) are retained (line 20) and added to the final set of particle windows $S$ (line 21). $Thrs$ depends on the classifier and is often 0. In [7, 9] the number of total iterations is fixed a priori, instead we propose to adjust it according to a suitable convergence metric (line 23), like the entropy, intended as a measure of uncertainty of a continuous density. Similar ending conditions has been evaluated and discussed for tracking purposes in [12], that exploits the entropy of a pdf to define the appropriate size of the particle filter set. Eventually, recalling that the number of particles to sample decreases from stage to stage, the whole process can also be interrupted when the number of particles to sample ($N_t$) reaches zero.

The above procedure allows multiple detections for each target object, possibly at different scales and positions, and the non-maximal suppression employed here is a sequential kernel density approximation [8] over $S$ (line 26) is exploited. This method approximates an underlying pdf represented by a set of samples ($S$ in our case), with a compact mixture of Gaussians in a time that is linear with the cardinality of the sample set. The detected objects correspond to the means of each single Gaussian components of the resulting mixture (line 28).

It is worth noting that, similarly to [7], this algorithm can be easily extended to the case of videos by exploiting the Bayesian-recursive procedure.

3. Experimental Results

We performed extensive experimentation of the multi-stage particle-window method (MS-PW, hereinafter) exploiting the HOG-SVM classifier, on two public datasets: INRIA [3] and CWSi (Construction Working Sites images)\(^1\). They contain images of variable size (from 333x531 to 1280x960) and complexity, and with people of quite diverse dimensions (from 32x80 to 320x800), which require to consider a large range of scales (in most tests, the biggest is 8 times the smallest). In addition, CWSi contains challenging backgrounds, distractors and people squatted or occluded by pillars or scaffoldings. INRIA consists of 311 images with an average of 1.87 persons per image, while CWSi dataset has 300 images with 2.60 persons per image. Example images of the two datasets are reported in Fig. 2.

The experiments aim at demonstrating two complementary concepts: on one hand, MS-PW exhibits lower detection time when it is configured to operate at the same detection accuracy of SW; on the other hand, MS-PW yields a higher detection accuracy when it is configured to operate with the same detection time of SW. These different operating points are achieved in the following manner: we define the performance (both in speed and accuracy) of the SW approach at several scale strides; we start at 1.05 and increase it gradually; the initial value is suggested by [3], where the authors also recommend a pixel stride of 8, that we maintain fixed throughout all tests, since increasing it would strongly affect performances. As expectable, the higher the scale stride is, the lower is the detection time and the accuracy also. Having defined a few SW operating points, we then configure MS-PW to work at the same speed (and in some cases at the same accuracy) for each of these SW test-cases, tuning the number of particle windows employed. Within this evaluation process, the HOG-SVM classifier is left totally untouched between the two detection approaches.

The accuracy of pedestrian detection is measured at object level in terms of the matching of the bounding box found by the detector with the bounding box in the ground truth, using the measure defined in the PASCAL object detection challenges [15].

In this analysis we employ Detection Error Trade-off (DET) curves [13]. Accuracy can also be roughly evaluated through single-valued scalar quantities, representing cumulative measurements: typical choices are the Area Under the ROC Curve (ROC-AUC) and the Average Precision (AP) [15], that better emphasizes the detection accuracy w.r.t. the detection confidence. We employed the AP, since the PASCAL VOC challenge has preferred it to the ROC-AUC, starting from the 2007 edition [5]. Additionally, we also report the miss rate at the reference value FPPI=1.

The results of our tests are reported in Fig. 3 and Table 1. The PASCAL threshold is set to $T = 0.50$. The SW detections of rows 2 (and 9) in Table 1 are configured according to the recommendations in [3] (i.e. scale stride 1.05 and pixel stride 8). Row 3 shows MS-PW at same speed, where a modest, though not negligible increase in performance of 2.33% over the AP is obtained. On the contrary, row 4 shows MS-PW at same accuracy of SW, demonstrating that MS-PW compute 2.91 times faster than SW, which means almost a third of the time at no cost in accuracy. We prefer to report time speedup instead of time absolute values since these latter measurements would be strongly implementation-dependent, i.e. they would be determined by the employed PC, the software tool-chain and the quality of the implemented code.

Row 5 is obtained with scale stride of 1.125, row 7 with

\(^1\)CWSi dataset and annotation have been downloaded from \url{http://www.openvisor.org}
scale stride of 1.235: as expected, increasing scale stride (i.e. reducing the number of windows) affects accuracy. MS-PW of rows 6 and 8 is configured to operate at the same speed of rows 5 and 7 respectively: the AP gain of MS-PW over SW tends to grow proportionally to the reduction of the windows employed, demonstrating that the multi-stage approach is to be preferred over the regular grid-distributed SW approach when fast object detection is required, that is often the case of video surveillance. In total analogy to what has been observed over INRIA, similar trends are shown on rows 9 to 15 over the CWSi dataset, but the advantage of MS-PW w.r.t. SW is stronger over this challenging dataset: the AP gain obtained using MS-PW over SW goes from 2.33% - 10.01% in the case of INRIA to 3.04% - 17.73% in the case of CWSi. This confirms that the information gain offered by MS-PW across the multiple stages brings benefits also in case the visual conditions are odd.

Fig. 3 shows MS-PW and SW DET curves at the described working conditions. It is possible to appreciate how the DET curves of MS-PW remain almost constantly below the DET curves of SW configured at same speed, confirming the results in Table 1.

### 4. Conclusions

Iterative multi-stage object detection has been applied to ensemble of classifiers. This paper shows that the introduction of proper modifications to the algorithm opens to possibility to successfully apply it to monolithic classifiers. Specifically, we propose the use of a sigmoid function over the margin of the classifier, introduce a dynamic control over the number of iterations and remove the use of a proposal function. Tests are performed on the popular

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Approach</th>
<th># windows</th>
<th>AP (gain)</th>
<th>MRP FPI = 1 (gain)</th>
<th>Speedup</th>
</tr>
</thead>
<tbody>
<tr>
<td>INRIA</td>
<td>MS-PW s.s.</td>
<td>44150</td>
<td>0.771 (n/a)</td>
<td>0.20 (n/a)</td>
<td>n/a</td>
</tr>
<tr>
<td>3</td>
<td>MS-PW s.s.</td>
<td>39800</td>
<td>0.789 (2.33%)</td>
<td>0.20 (0.0%)</td>
<td>n/a</td>
</tr>
<tr>
<td>4</td>
<td>MS-PW s.s.</td>
<td>13150</td>
<td>0.771 (n/a)</td>
<td>0.20 (0.0%)</td>
<td>2.91</td>
</tr>
<tr>
<td>5</td>
<td>SW</td>
<td>23500</td>
<td>0.736 (n/a)</td>
<td>0.22 (n/a)</td>
<td>n/a</td>
</tr>
<tr>
<td>6</td>
<td>MS-PW s.s.</td>
<td>19900</td>
<td>0.775 (5.30%)</td>
<td>0.20 (0.0%)</td>
<td>n/a</td>
</tr>
<tr>
<td>7</td>
<td>SW</td>
<td>14400</td>
<td>0.699 (n/a)</td>
<td>0.28 (n/a)</td>
<td>n/a</td>
</tr>
<tr>
<td>8</td>
<td>MS-PW s.s.</td>
<td>11500</td>
<td>0.769 (10.01%)</td>
<td>0.22 (21.43%)</td>
<td>n/a</td>
</tr>
<tr>
<td>CWSi</td>
<td>SW</td>
<td>51400</td>
<td>0.565 (n/a)</td>
<td>0.40 (n/a)</td>
<td>n/a</td>
</tr>
<tr>
<td>10</td>
<td>MS-PW s.s.</td>
<td>46600</td>
<td>0.582 (3.04%)</td>
<td>0.40 (0.0%)</td>
<td>n/a</td>
</tr>
<tr>
<td>11</td>
<td>MS-PW s.s.</td>
<td>18650</td>
<td>0.565 (n/a)</td>
<td>0.40 (0.0%)</td>
<td>2.32</td>
</tr>
<tr>
<td>12</td>
<td>SW</td>
<td>27200</td>
<td>0.522 (n/a)</td>
<td>0.45 (n/a)</td>
<td>n/a</td>
</tr>
<tr>
<td>13</td>
<td>MS-PW s.s.</td>
<td>23300</td>
<td>0.577 (10.61%)</td>
<td>0.42 (7.45%)</td>
<td>n/a</td>
</tr>
<tr>
<td>14</td>
<td>SW</td>
<td>16850</td>
<td>0.468 (n/a)</td>
<td>0.52 (n/a)</td>
<td>n/a</td>
</tr>
<tr>
<td>15</td>
<td>MS-PW s.s.</td>
<td>13500</td>
<td>0.550 (17.73%)</td>
<td>0.44 (14.60%)</td>
<td>n/a</td>
</tr>
</tbody>
</table>
HOG-SVM classifier and demonstrate the efficiency and efficacy of the algorithm.

References


