Learning to Detect Partially Overlapping Instances

Carlos Arteta[1] Victor Lempitsky[2] J. Alison Noble[1] Andrew Zisserman[1]
[1]Department of Engineering Science, University of Oxford, UK
[2]Skolkovo Institute of Science and Technology, Russia

Abstract

The objective of this work is to detect all instances of a class (such as cells or people) in an image. The instances may be partially overlapping and clustered, and hence quite challenging for traditional detectors, which aim at localizing individual instances.

Our approach is to propose a set of candidate regions, and then select regions based on optimizing a global classification score, subject to the constraint that the selected regions are non-overlapping. Our novel contribution is to extend standard object detection by introducing separate classes for tuples of objects into the detection process. For example, our detector can pick a region containing two or three object instances, while assigning such region an appropriate label. We show that this formulation can be learned within the structured output SVM framework, and that the inference in such model can be accomplished using dynamic programming on a tree structured region graph. Furthermore, the learning only requires weak annotations – a dot on each instance.

The improvement resulting from the addition of the capability to detect tuples of objects is demonstrated on quite disparate data sets: fluorescence microscopy images and UCSD pedestrians.

1. Introduction

Understanding images containing a large-number of objects of interest is a task of great importance for study and monitoring of both the macroworld (e.g. crowds of pedestrians, or animal and plant populations) and within the microscopy domain (cells of in-vitro cultures and developing embryos, blood samples, histopathology images, etc.).

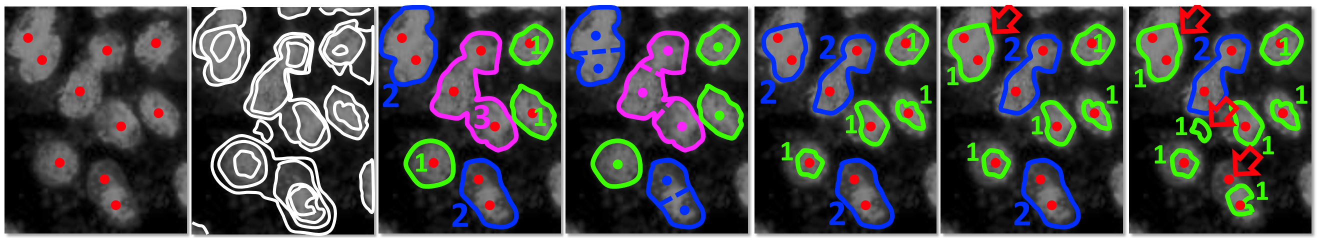
Most computer vision methods for this task fall into two classes. The first is based on individual object detection. Such detection can be based on a sliding window or Hough transform, followed by an appropriate non-maxima suppression procedure [3, 8, 14], stochastic fitting of interacting particles or object models [9, 10, 24], or region-based detection [2, 18, 19]. The second class contains the methods that avoid the detection of individual instances but instead perform analysis based on local or global texture and appearance descriptors, e.g. by recovering the overall real-valued count of objects in the scene [5, 12, 16, 22] or by estimating the local real-valued density of the objects in each location of interest [11, 15].

Depending on the degree of overlap between objects, the first or the second class of methods might be more appropriate. For low object-density images with infrequent overlaps between them, detection methods may perform very well, while regression/density estimation methods can e.g. hallucinate small but non-zero object density/object count spread across the parts of images that do not in fact contain any objects. Furthermore, the localization of individual objects in the detection-based approaches facilitates more intricate analysis by revealing patterns of co-location, providing the possibility for shape and size estimation of individual instances, and allowing the linkage of individual detections through time for video analysis.

For the high-density images, however, detection-based analysis may fail badly, especially when the amount of overlap and inter-occlusion between objects makes the detection of individual instances hard or impossible even for human experts. In such situations, the performance of density/global count estimation methods degrade more gracefully than detection-based methods. The analysis in this case is essentially reduced to texture matching between the test image and the training set, which may be feasible even when individual instances are not distinguishable.

In real life, many applications require the processing algorithm to handle both the high and the low-density scenarios. Furthermore, the two cases may co-exist within the same image. E.g. an image from a surveillance camera may contain multiple individual pedestrians but also few groups of people which are hard to segment from each other [7]. Likewise, a microscopy image may contain both regions of low and high cell density (sometimes corresponding to different morphological parts or different tissues).

In such situations neither of the methods mentioned above will perform optimally and this motivates the approach we present here. This approach generalizes and builds upon our region-based detection method [2]. The main novelty of the proposed method is its ability to parse the input image by detecting groups of objects of different integer sizes (with a "group" of size 1 being a par-



a) input and ground truth; b) pool of nested regions; c) solution with loss=0; d) postproc. of (c); e) solution with loss=0; f) solution with loss=1; g) solution with loss=3

Figure 1: Given an input image (a) our model considers a pool of nested regions (b) and accomplishes detection by picking a non-overlapping subset of regions (c), where each region is assigned a label corresponding to the estimated number of objects (green=1, blue=2, purple=3). Such solution can be further refined to estimate individual object locations (d). The learning in our model is performed based on weak annotation (red dots) and is driven by an *instance count* loss. Solutions with zero loss (c and e) as well as non-zero loss (f and g) are shown. In the latter case, arrows indicate violations from the perfect correspondence between the solution and the ground truth dotting.

ticular case). Via training performed on a set of weakly-annotated training images, the proposed method learns to choose different group sizes depending on the object density. Thus, similarly to local density estimation, it can avoid trying to discern individual objects when they are clumped together. Unlike local density estimation, however, the proposed method is able to enforce the fact that each group has an integer number of objects.

Similarly to our initial approach [2], the parsing process is based on an efficient and exact inference procedure that detects a set of non-overlapping extremal regions delivering a maximum to the parsing functional. The learning is performed in a structured SVM framework and optimizes the (convex upper bound on the) counting loss. We observe that such learning yields a desirable bias to prefer the most detailed explanation, e.g. to choose the groups of the smallest size whenever objects are discernable, as this strategy tends to provide the highest counting accuracy.

We conduct a set of experiments with real and synthetic fluorescence microscopy images, as well a surveillance data from the UCSD pedestrian dataset. We observe that the proposed method achieves very good detection accuracies despite large amount of overlap, and very low effective resolution. For all datasets, the proposed method outperformed other detection methods, including a considerable improvement over the baseline [2], and is comparable with the methods that are trained to count (and do not perform detection).

2. Background and Contributions

Detection of overlapping instances. Our main contribution is to extend standard object detection by introducing separate classes for tuples of objects into the detection process. Thus rather than trying to reason about the boundary and part assignment between several tightly overlapping regions [3, 8, 14, 21], tuples of objects are detected as a whole, making the object detection process more resilient to strong object overlap. At the inference stage, our model automatically chooses the level of granularity (Figure 1-c), i.e. which objects will be detected as singletons and which as part of a group. If necessary, each group can be separated into singletons in a simple post-processing step (Figure 1-d).

Object recognition through extremal regions. Our approach to detection does not use a scanning-window detector. Instead, we follow the observation [18] that good object support hypotheses can be provided by extremal regions of the image, for example *MSER* [17] (Figure 1-b). These regions are well suited to biomedical data [2] and text detection [19]. As an additional contribution, we extend the applicability of this approach by using extremal regions of a derived image (rather than the input image itself). For example, we use the extremal regions of a soft background difference image to generate detection hypotheses for a surveillance image stream (whereas extremal regions of the input images themselves would provide a poor hypotheses set). Note that this approach does not rely on obtaining a *binary* foreground mask from the background difference image (as is commonly done, see e.g. [10, 22]).

Extremal regions with non-overlapping constraint. Our computational model is based on our previous work [2] that also used non-overlapping extremal regions. Whilst that initial model achieves good results on those biomedical datasets where objects are clearly discernable from each other as extremal regions, it struggles to achieve high recall when that is not the case (i.e. when for some object X, any extremal region containing X also contains another overlapping object Y; in this case [2] has no hope of detecting both X and Y as they have to be detected as separate extremal regions). This problem is overcome by our current approach, where spatially-close objects that are not clearly discernable from each other are detected together (as a pair, a triple, a quadruple, etc.).

Our model is thus a strict generalization of [2] and, for the purpose of self-containment, in the following section we describe our complete approach without focusing on the distinctions from [2]. The gains resulting from the generalization will be demonstrated in the experiments section.

3. The Model

For an input image $\mathcal{I}$ containing multiple instances of an object class (some of which may be overlapping) we want to automatically detect the instances and provide an estimate of their location. We start by generating a pool of N *nested* regions, such that for each pair of regions R_i and R_j in the pool, these regions are either nested (i.e. $R_i \subset R_j$ or $R_i \supset R_j$) or they do not overlap ($R_i \cap R_j = \emptyset$). In the simplest case, a pool can comprise extremal regions of the input image (i.e. connected components of the binary images $\mathcal{I} > \tau$ where τ is an arbitrary threshold). More generally, we can transform the input image in various ways, creating a new map $\mathcal{I}$ where higher-value regions correspond to higher probabilities of an object's presence. The pool of candidate regions can then be generated as a set of extremal regions in the transformed image $\mathcal{I}$.

Once the pool of nested regions is generated, each region is scored using a set of classifiers that evaluate the similarity of such region to each of D classes, where each class signifies the integer number of instances of the object that the region contains (i.e. a region of class d contains d instances). During the learning stage, these classifiers are trained in a coordinated fashion within the structured output framework. Given the scores of the classifiers, an inference procedure selects a non-overlapping subset of regions, and assigns each selected region in the subset a class label, thus indicating the number of objects that our system believes this region represents. The choice of the region subset and the class labels are driven by the optimization process that simply maximizes the total classifier score corresponding to selected regions and class labels subject to the non-overlap constraint.

More specifically, let $V_i(d)$ denote the classifier score of a region R_i for class d (the higher the score, the more this region looks like a typical region containing d object centroids). For notational simplicity, we also define $V_i(0) = 0$. We introduce the optimization variables $\mathbf{y} = \{y_i | i = 1 \dots N\}$, where $y_i = 0$ means that the region R_i is not selected, and $y_i = d \in 1 \dots D$ means that the region R_i is selected and assigned class d. We denote with $\mathcal{Y}$ the set of all $\mathbf{y}$ that meet the non-overlap constraint, i.e. such that $\forall i, j$: if $R_i \cap R_j \neq \emptyset$ then $y_i \cdot y_j = 0$. Then the inference is accomplished through the following constrained maximization:

$$F(\mathbf{y}) = \sum_{i=1}^{N} V_i(y_i) \to \max , \text{s.t. } \mathbf{y} \in \mathcal{Y} \tag{1}$$

This maximization of (1) can be performed exactly and efficiently using dynamic programming (since the region pool has a tree structure – this follows from the nestedness property of the regions). The optimization method is described in the implementation details below.

3.1. Learning the model

The model for the evaluation of the regions can learn from weak annotations, i.e. a dot inside each instance of the object on a set of training images (rather than, say, a bounding box around each instance). Such learning is driven by an *instance count loss (IC-loss)* (2) that penalizes all deviations from the one-to-one correspondences between annotation dots and the selected regions (Figure 1).

Suppose we have M training images $\mathcal{I}^j$ indexed by j. Let d_i^j now be the number of dots contained in the region R_i^j, and N^j be the total number of dots in $\mathcal{I}^j$. The *IC-loss* imposed by such annotation on each possible region labeling $\mathbf{y}$ is formulated as:

$$L(\mathbf{y}^j) = \sum_{i=1}^{N^j} [y_i^j > 0] \Delta(d_i^j, y_i^j) + N^j - \sum_{i=1}^{N^j} [y_i^j > 0] d_i^j \tag{2}$$

Here, the first term penalizes the deviations between the assigned class label y_i^j of the selected regions and the true number d_i^j of dots inside of it. The penalty is determined by the function $\Delta(\cdot, \cdot)$, described later in this section. The last two terms correspond to the total number of unmatched (uncovered) dots for the $\mathbf{y}^j$ configuration under the non-overlap constraint, and thus penalize false negatives (missed detections).

Assuming that the properties of each region R_i^j are characterized by the *feature vector* $\mathbf{f}_i^j$, we set the classification scores to be linear functions of these feature vectors: $V_i^j(d) = (\mathbf{w}_d \cdot \mathbf{f}_i^j)$, where $\mathbf{w}_d$ is the parameter vector for the dth class, and has the same dimensionality as the feature vector. The aim of the learning is to find a vector $\mathbf{w} = \left[\mathbf{w}_1^T, \mathbf{w}_2^T, \dots, \mathbf{w}_D^T\right]^T$ so that the inference (1) produces configurations with low IC-loss.

A simple approach for learning $\mathbf{w}$ is to train binary classifiers for each of the D classes, in a one-versus-rest fashion. However, such an approach ignores the inference process and the non-overlap constraint, and we therefore perform learning within a structured output learning framework; specifically, a structured SVM—[23]. Thus, since the loss (2) cannot be optimized directly, a convex upper bound is optimized instead. The minimization objective on $\mathbf{w}$ can then be written as:

$$||\mathbf{w}||^2 + \frac{C}{M} \sum_{j=1}^{M} \max_{\mathbf{y}^j \in \mathcal{Y}^j} \left(L(\mathbf{y}^j) + \mathbf{w} \cdot \left(\Psi(f^j, \mathbf{y}^j) - \Psi(f^j, \bar{\mathbf{y}}^j) \right) \right) \tag{3}$$

where the first term is the regularization on $\mathbf{w}$, the second term is the upper bound on the training error, C is a constant that controls the trade-off between them, $\bar{\mathbf{y}}^j$ is some given "ground-truth" configuration (see later) with the zero

IC-loss, and $\Psi(\mathbf{f}^j, \mathbf{y}^j)$ is the *joint feature representation* defined as follows:

$$\Psi(\mathbf{f}^j, \mathbf{y}^j) = \left[\ \sum_{i=1}^{N^j} [\mathbf{y}^j = 1]\mathbf{f}_i^j, \ldots, \sum_{i=1}^{N^j} [\mathbf{y}^j = D]\mathbf{f}_i^j\ \right]^T \tag{4}$$

The optimization objective (3) can be minimized with a cutting plane algorithm [23], for which an efficient way of computing the most violated constraint is required. Specifically, we need to compute the second term of equation (3) for a fixed $\mathbf{w}$ (*loss-augmented inference*). Fortunately, in our case the loss (2) decomposes in an appropriate way, and the loss-augmented inference corresponds to the following optimization (after removing the terms independent from $\mathbf{y}^j$):

$$\max_{\mathbf{y}^j \in \mathcal{Y}^j} \sum_{i=1}^{N^j} [y_i^j > 0] \left(\Delta(d_i^j, y_i^j) - d_i^j\right) + \mathbf{w} \cdot \left(\Psi(f^j, \mathbf{y}^j)\right) \tag{5}$$

The maximization of (5) is then reduced to the optimization (1) with $V_i^j(y_i^j) = \mathbf{w}_{y_i^j} \cdot f_i^j + \Delta(d_i^j, y_i^j) - d_i^j$ and solved with the same dynamic programming inference.

Penalization function Δ. The direct extension of the penalization function $\Delta(d_i^j, y_i^j)$ in [2] for the case of multiple objects would be $\Delta^u(d_i^j, y_i^j) = |d_i^j - y_i^j|$; where the penalization would have the same behaviour regardless of the estimated class or the true number of dots inside the region. However, when considering the possibility of regions containing multiple objects, we must take into account the increasing intraclass variability (e.g. of region shape) for higher-order classes that would bias the labels assigned to the regions towards low-order classes. In order to counterbalance such effect, we use a re-scaled penalization based on the true number of dots d_i^j inside the region R_i^j. Intuitively, assigning a class 7 to a region that contains 6 instances is not as bad as assigning a class 3 to a region with 2 instances, thus it is not penalized so hard.

After experimentation with several variants of $\Delta(\cdot, \cdot)$, we found the best performing to be $\Delta^r(d_i^j, y_i^j) = \begin{cases} (y_i^j - d_i^j)/(d_i^j + 1), \text{ if } y_i^j \geq d_i^j \\ d_i^j - y_i^j, \text{ if } y_i^j \leq d_i^j \end{cases}$.

Reestimating the "ground truth" configuration. In the derivation above, the "ground truth" configuration $\bar{\mathbf{y}}$ was assumed given for each image; however, only dot-annotations are given at training time (not labeled regions), thus multiple "correct" (i.e. zero-loss) region configurations can be consistent with such annotation (Figure 1c,e). To handle this, we follow a conventional way [25] and add the "ground truth" configuration for each image into the optimization (3) as a latent variable $\mathbf{h}^j \in \mathcal{H}^j$ (where $\mathcal{H}^j$ denotes the set of all labelings with the zero IC-loss). The learning is reformulated as the following optimization:

$$\min_{\mathbf{w}, \mathbf{h}^j \in \mathcal{H}^j} \left\{ ||\mathbf{w}||^2 + \frac{C}{M} \sum_{j=1}^{M} \max_{\mathbf{y}^j \in \mathcal{Y}^j} \left(L(\mathbf{y}^j) + \mathbf{w} \cdot \Psi(f^j, \mathbf{y}^j)\right) - \frac{C}{M} \sum_{j=1}^{M} \mathbf{w} \cdot \Psi(f^j, \mathbf{h}^j) \right\} \tag{6}$$

The new objective can then be optimized by alternation. This implies that we need to provide a way of imputing the latent variable such that the problem is reduced to the standard structural SVM in (3) for each iteration of the alternation algorithm. Specifically, at the beginning of iteration t for each training image j, we need to find $h^j \in \mathcal{H}^j$ that maximizes $\sum_{j=1}^{M} \left(\mathbf{w} \cdot \Psi(f^j, \mathbf{h}^j)\right)$. To achieve this, we run the optimization (1) over $\mathcal{Y}^j$ but set $V_i^j(y_i^j) = \mathbf{w} \cdot \Psi(f^j, \mathbf{h}^j) + d_i^j \cdot v - [y_i^j \neq d_i^j] \cdot N^j\, v$, where v is a very large positive constant. This choice of V_i^j ensures that the maximum in (1) is attained for a zero-loss configuration from $\mathcal{H}$ and that the costs of all such configurations differ from $\sum_{j=1}^{M} \left(\mathbf{w} \cdot \Psi(f^j, \mathbf{h}^j)\right)$ by the same constant $N^j\, v$.

3.2. Implementation details

Dynamic programming for inference. The maximization of (1) can be performed exactly and efficiently by exploiting the nestedness property of the region pool. Indeed, one can consider a tree-structured model, where each node corresponds to a region and where parent-child links correspond to the nestedness property. Namely, the node R_j becomes a parent of the node R_i if R_j is the smallest region in the pool that R_i strictly belongs to. In this way, because of the nestedness, the region pool can be organized into a forest. We can then introduce the auxiliary variables z_i. The auxiliary variables $\mathbf{z}$ are uniquely determined by the initial variables $\mathbf{y}$ in the following sense: $z_i = d > 0$ iff either $y_i = d$ or some y_k such that R_k is an ancestor of R_i in the tree equals d (note that two ancestors of the same region cannot be assigned non-zero labels simultaneously as long as $\mathbf{y} \in \mathcal{Y}$). The optimization (1) can then be rewritten as a pairwise tree-structured MRF on the auxiliary variables:

$$F(\mathbf{z}) = \sum_{i|p(i)\neq 0} W_i(z_i, z_{p(i)}) + \sum_{i|p(i)=0} V_i(z_i) \to \max \tag{7}$$

where $p(i)$ maps region R_i to the number of its parent region ($p(i) = 0$ for root regions in the forest), $W_i(d, d) = 0$, $W_i(d, 0) = V_i(d)$, $W_i(0, d > 0) = -\infty$, and $W_i(d_1, d_2 \neq d_1) = -\infty$ as long as $d_2 > 0$. After such variable change, all $y \in Y$ are one-to-one mapped to z configurations with the finite values of the functional (7) and this mapping preserves F. The optimization task (7) can be accomplished via tree-based dynamic programming [20] (the max-product algorithm). It is then trivial to compute the optimal solution of (1) from the optimal solution of (7).

Postprocessing for inference. Several potential applications and performance measures require the output of the method to be in the form of the sets of individual instances. We use a very simple post-processing in this case. For each selected region R_i we run k-means with $k = y_i$ on the image coordinates of all pixels in that region, thus obtaining an estimate for the set of centroids of individual objects.

Initialization and termination for learning. The initialization of $\mathbf{w}$ for the alternation-based maximization (6) is obtained by learning and concatenating a set of D binary classifiers $\mathbf{w}_1, \mathbf{w}_2, \cdots \mathbf{w}_D$ in a one-versus-rest fashion. The positive training examples for the binary classifier $\mathbf{w}_d$ consist of all regions in the training images that contain d dots. The alternations are stopped once the amount of change in the ground truth configuration with respect to the previous iteration $\frac{||\bar{\mathbf{y}}_t - \bar{\mathbf{y}}_{t-1}||}{M}$ falls below a pre-specified threshold ϵ.

4. Experiments and Results

To show the performance and generality of the method presented, results are reported for two different tasks: cell detection in microscopy images (Figure 2) and pedestrians detection in surveillance videos (Figure 3). We evaluated the performance with two kinds of metrics. Our primary metric is *mean absolute counting error*, which measures the absolute mismatch in the number of objects in an image between the output and the GT. The secondary set of metrics are *precision*, *recall* and the F_1 score of the detection accuracy defined in a standard way. Here, to define true and false positives as well as missed detections for each image, we use the Hungarian algorithm to match (one-to-one) the GT and the predicted set of centroids, with the radius of the smallest object being the threshold for the acceptance of the match. In the case of the pedestrians dataset, where a depth map of the scene is provided, the match threshold was made inversely proportional to the pixel depth. All metrics are average numbers over the test dataset.

4.1. Cell Detection

Detecting cells in microscopy images is a challenging task in many real applications. Among the several difficulties that cellular images present for detection algorithms, partial cell occlusion and image saturation (common in fluorescence microscopy) are particularly challenging as the boundaries between cells tend to disappear, effectively merging different cells into one object in the images. We have selected two datasets to show the applicability of our method for this scenario: a synthetic and a real dataset of fluorescence microscopy. A synthetic dataset has the advantage of containing perfect annotations, making it ideal to evaluate the different components of our method as shown in Table 1.

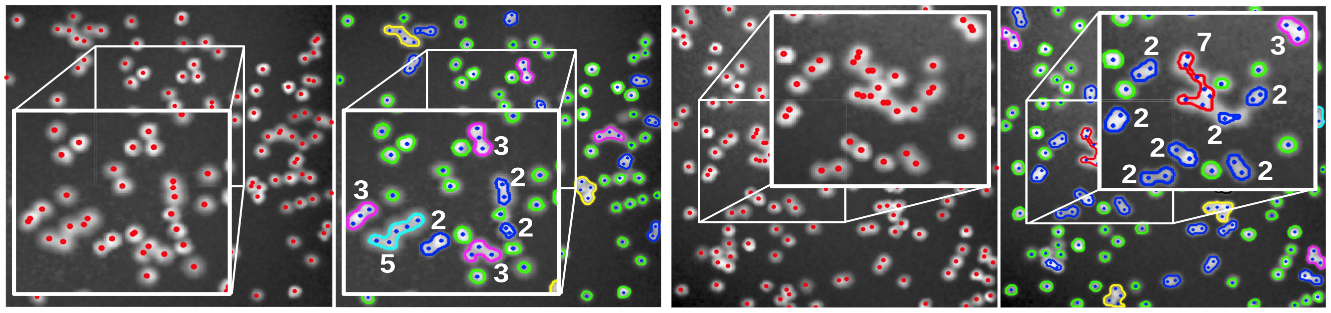
	MCE	Prec.	Rec.
Fiaschi *et al.* [11]	3.2 ± 0.1	-	-
Lempitsky & Zisserman [15]	3.5 ± 0.2	-	-
Barinova *et al.*[3]	6.0 ± 0.5	-	-
Baseline [2]	51.2 ± 0.8	98.87	72.07
This work w$\backslash\Delta^u$, no lat.var.	14.31 ± 0.5	97.79	90.82
This work w$\backslash\Delta^r$, no lat.var.	5.38 ± 0.1	93.45	94.33
This work w$\backslash\Delta^r$ + lat. var.	5.06 ± 0.2	94.58	94.62

Table 1: **Accuracy for the synthetic cell dataset and components evaluation.** The high cell confluency in the synthetic cell dataset [15] poses a difficult challenge for detection algorithms due to very high cell overlap. Therefore, it is expected that counting algorithms such as [11, 15] would outperform detection methods. Nonetheless, our method is able to produce a comparable mean counting error (MCE), while providing estimates of object localization evaluated with precision and recall. The baseline method [2] is unable to detect objects in groups, and thus fails badly in this dataset. The extension to multiple classes (tuples) contributes most to the improvement over [2], especially when the penalization in the cost function is done according to Δ^r, in comparison to Δ^u proposed in [2] (see section 3.1). Further improvement is obtained when including the optimization over the "ground-truth" in included through the use of latent variables.

For both cell datasets, the candidate pools of regions were obtained by taking the (stable) extremal regions of the raw input images by running the MSER detector [17] thus effectively sampling the set of all extremal regions (a very low stability threshold was used, so that several hundred regions were accepted for each image).

The feature vector used to encode each extremal region in the cell images (similar to [2]) consists of the concatenation of the following descriptors: (i) a 150-dimensional binary vector encoding the area (where the area is soft-encoded by putting two non-zeros at the adjacent entries corresponding to the region), (ii) a 12-dimensional histogram of intensities inside the region, (iii) two 8-dimensional histogram of difference of intensities between the boundary of the extremal region and a dilation of it (over two different dilation scales), (iv) a 60-dimensional shape descriptor (see [2]), and finally, (v) a binary vector of the same dimension as the number of classes which encodes the number of leaf regions (i.e. regions without nested regions in the pool) nested within a given region. This last descriptor often indicates the presence of individual objects existing inside the region being encoded.

Synthetic flourescence microscopy. The synthetic dataset of flourescence microscopy from [15] consists of 200 images generated with [13], divided in half for testing and training, with an average number of 171 ± 64 cells per



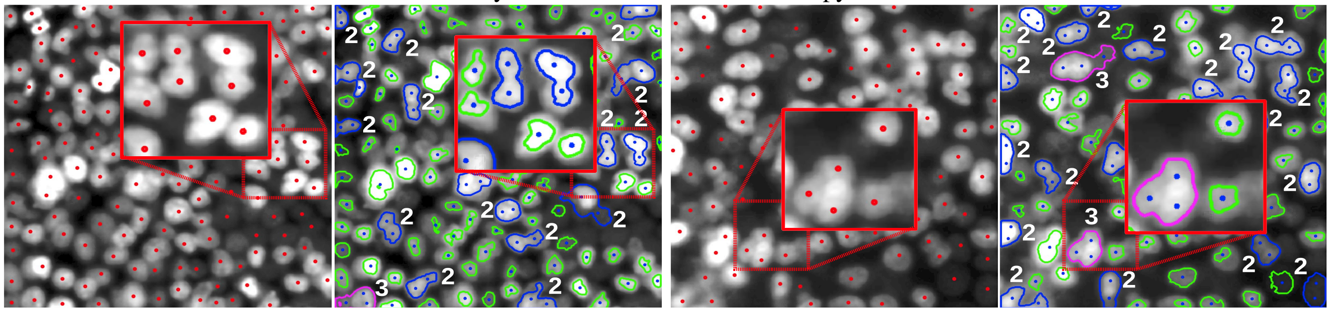


Figure 2: *(best viewed in color)* Results for our method on fluorescence microscopy datasets. Input images are shown with their ground-truth annotations overlaid (red dots) for comparison. The output images show the selected regions, colour-coded according to the estimated number of objects inside of it (green=1, blue=2, purple=3, yellow=4, cyan=5, red=7), also indicated with digits omitting class 1 for clarity. The estimated centroids obtained through postprocessing are shown in blue. Despite large amounts of object overlap, the proposed method is capable of providing quite accurate parsings of the images.

image. In [15], this dataset is further divided (randomly) into different splits in order to vary the size of the training and validation sets. In Table 1 we show the performance of our method and its components for the case of $32 + 32$ training+validation images (same splits as in [15]). The improvement brought by each component presented in this work is compared with the baseline [2], which essentially corresponded to limiting our proposed model to a single class, and with a single ground-truth estimation (same features were used in the evaluation). Moreover, we compare to the counting methods [11, 15] and the detection method [3]. As expected, the counting algorithms can outperform the detection methods in cases of very high object overlap such as this synthetic cells dataset. Nevertheless, our method can achieve comparable results with high precision and recall. The baseline [2], restricted to one object per extremal regions, cannot cope with the level of object clustering in this dataset and thus performs poorly.

Real flourescence microscopy. To show the performance on real microscopy data, we evaluated the method on the small dataset of 9 embryo fluorescence images (about 500×500 pixels; the dataset is derived from one used in [2, 4] by more careful annotation and leaving out 3 images where reliable dot annotation was impossible even for a human).

As before, we evaluated the method of [2] as baseline, and also compare with the recent method of [4] in Table 2.

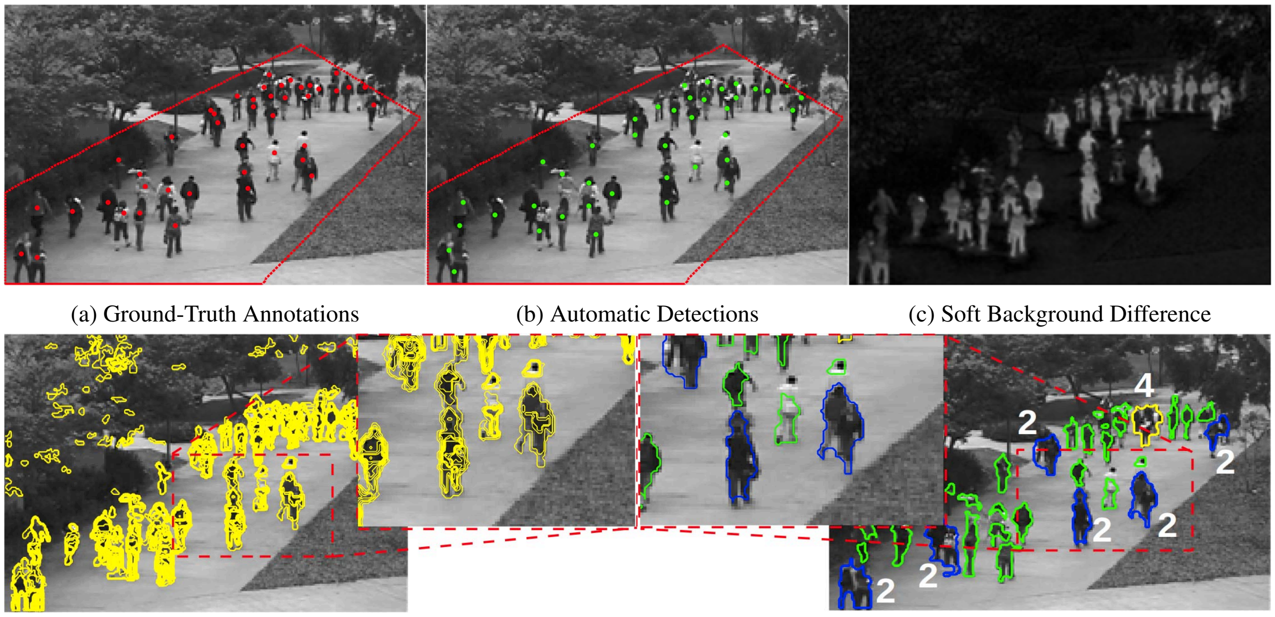
	F1-Score	Count Error
This work	0.87	5.11
Arteta *et al.*[2]	0.86	6.11
Bernardis *et al.*[4]	0.86	8.00

Table 2: **The accuracy for the real cell data set.** The proposed method outperforms the two previous methods both in terms of the detection accuracy and the counting accuracy. Methods were optimized for best counting performance.

In general, the proposed method outperformed both competitors, both in terms of detection accuracy and, more substantially, in terms of the counting error. While we have not restricted the number of classes in our method, we have observed that only classes from 1 to 3 were returned in the results, indicating that the learning procedure has considered the higher-order classes detrimental.

4.2. Pedestrian detection

We apply our method to detect and count pedestrians in the UCSD surveillance camera dataset [6]. It consists of 2000 frames (158×238 pixels) from a video surveillance camera, annotated with a dot on each pedestrian and supplemented with an approximate depth image and the region of



(a) Ground-Truth Annotations (b) Automatic Detections (c) Soft Background Difference

(d) Candidate Extremal Regions (e) Selected Regions

Figure 3: *(best viewed in color)* Results for our methods on the UCSD pedestrian dataset. Due to the amount of overlap and low effective resolution, this dataset poses a big challenge for detection algorithms. Nonetheless, our method is able to produce accurate detection results (b) as compared to (a) the human annotations. Extremal regions are collected from (c) the soft background difference image (see text), and a portion of those regions is shown over the original image (d). The method selects non-overlapping regions (e) and estimates the number of instances of the object that the region contains, which allows the prediction of the location of the individual instances. Digits indicate the estimated number of instances inside the region, and green regions correspond to single objects.

interest. The pedestrians frequently occlude each other and are imaged at a very low resolution (the furthest pedestrians are just a few pixels tall). All this makes detection very hard for this dataset, and although a number of counting methods have been evaluated on it, to the best of our knowledge, we are the first to run detection algorithms.

As pedestrians can correspond to both dark and bright regions, we cannot use the extremal regions of the input images. Instead, to generate the tree of regions for this data, we computed the background image using a simple median filtering of a sparsely sampled set. For each frame, we then simply compute the absolute value of the difference with the background and look for extremal regions in this difference image. To reduce the number of candidate regions to a few hundreds, we applied a mild Gaussian smoothing to the difference image ($\sigma = 1$ pixel).

To describe each region, we have used (i) the histogram of visual words computed with tree codebooks as in [15], (ii) the area feature (as above), (iii) the histograms of intensities for the difference image, (iv) the histograms of Canny edge orientations as in [22], and (v) the nestedness feature (as above). All vectors were concatenated to obtain f_i^j.

We follow the protocol from [22] and split the data into four groups in order to assess accuracy, scalability and practicality. The first split, 'maximal', contains 128 frames out of a segment from the video, the splits 'upscale' and 'downscale' train on the most and least crowded frames respectively, and the 'minimal' split trains on only 10 frames. The counting results are shown in Table 3. In general, the proposed method outperforms the baseline [2]. The counting accuracy of our detection method is comparable with the accuracy of methods that are trained to count and are not able to estimate the locations of individual pedestrians (even for singletons). For this dataset, we have observed that the method produced classes 1 to 5, indicating that discerning individual instances was harder than in the case of the real cell images.

In terms of the detection accuracy, the proposed method has also achieved an improvement over the baseline [2] (Table 4). This is due to the fact that the proposed method, while maintaining a precision similar to the baseline, is able to increase the recall as it has the capacity to handle overlapping objects.

5. Summary and Discussion

We have presented a new model for object detection, which is particularly suitable for images with multiple over-

	'max'	'down'	'up'	'min'
Global count [12]	2.07	2.66	2.78	N/A
Segment+Count [22]	1.53	1.64	1.84	1.31
Density estim. [15]	1.70	1.28	1.59	2.02
Density estim. [11]	1.70	2.16	1.61	2.20
Baseline [2]	2.55	2.25	2.93	2.86
This work	1.98	1.55	2.16	2.35

Table 3: **Mean absolute errors for people counting in the surveillance video [6].** The columns correspond to the four splits ('maximal','downscale','upscale','minimal'). Our detection method approaches the counting accuracy of the counting methods, while outperforming the baseline detection [2] in all splits.

	'max'	'down'	'up'	'min'
Baseline [2]	87.22	87.66	88.30	86.47
This work	89.53	89.99	89.21	86.64

Table 4: **Detection accuracy** in terms of 100*F_1 score for the four splits of the UCSD pedestrian dataset. In this experiment, we varied the bias of the learned classifiers to generate recall-precision curves and picked the point with the highest F1-score on them. Generally, the proposed method resulted in higher optimal F1-score (and also reached the solutions with higher recall) compared to the baseline [2].

lapping object instances. Depending on the difficulty of the detection task, the model has the flexibility to choose groups of variable sizes (including individual instances if the task is easy). The ability to pick the optimal level of granularity (i.e. to determine whether the task is "hard" or "easy") is seamlessly obtained during the learning of the model. The inference in the model is computationally efficient, requiring only a few hundred classifier evaluations followed by tree-based dynamic programming.

The use of the model is particular attractive for biomedical images, where it considerably outperforms the baseline [2] that can only predict individual instances all the time. Thanks to the presented generalization of the region pool generation process, we could also apply the model to object detection in surveillance imagery, obtaining good detection accuracy despite low resolution.

Limitations and extensions. One of the limitations of the proposed method appears when the instances become even denser than in the considered datasets and a higher number of classes is needed to parse such images. In this case, our structured output framework fragments the training data, so that higher-order classes effectively receive less training examples. We are addressing this issue by investigating different learning frameworks for our model. Another obvious possibility for improvement is a more sophisticated postprocessing procedure (e.g. similar to [10]). Finally, it is worth noting that all that is required of the candidate regions is that they are nested. Thus, although we have used extremal regions for candidates, they could instead be generated by hierarchical image segmentation, e.g. [1].

Acknowledgements. Financial support was provided by the RCUK Centre for Doctoral Training in Healthcare Innovation (EP/G036861/1) and ERC grant VisRec no. 228180.

References

[1] P. Arbelaez, M. Maire, C. Fowlkes, and J. Malik. Contour detection and hierarchical image segmentation. *TPAMI*, 33:898–916, 2011.

[2] C. Arteta, V. Lempitsky, J. A. Noble, and A. Zisserman. Learning to detect cells using non-overlapping extremal regions. In *Proc. MICCAI*, 2012.

[3] O. Barinova, V. Lempitsky, and P. Kohli. On the detection of multiple object instances using Hough transforms. *TPAMI*, 2012.

[4] E. Bernardis and S. X. Yu. Pop out many small structures from a very large microscopic image. *Med. Image Analysis*, 2011.

[5] A. Chan and N. Vasconcelos. Bayesian poisson regression for crowd counting. In *Proc. CVPR*, 2009.

[6] A. B. Chan, Z.-S. J. Liang, and N. Vasconcelos. Privacy preserving crowd monitoring: Counting people without people models or tracking. In *CVPR*, 2008.

[7] W. Choi and S. Savarese. A unified framework for multi-target tracking and collective activity recognition. In *ECCV*, 2012.

[8] C. Desai, D. Ramanan, and C. Fowlkes. Discriminative models for multi-class object layout. In *Proc. ICCV*, 2009.

[9] X. Descombes, R. Minlos, and E. Zhizhina. Object extraction using a stochastic birth-and-death dynamics in continuum. *Journal of Mathematical Imaging and Vision*, 2009.

[10] L. Dong, V. Parameswaran, V. Ramesh, and I. Zoghlami. Fast crowd segmentation using shape indexing. In *Proc. ICCV*, 2007.

[11] L. Fiaschi, R. Nair, U. Köethe, and F. Hamprecht. Learning to count with regression forest and structured labels. In *Proc. ICPR*, 2012.

[12] D. Kong, D. Gray, and H. Tao. A viewpoint invariant approach for crowd counting. In *Proc. ICPR*, 2006.

[13] A. Lehmussola, P. Ruusuvuori, J. Selinummi, H. Huttunen, and O. Yli-Harja. Computational framework for simulating fluorescence microscope images with cell populations. *IEEE TMI*, 2007.

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

[15] V. Lempitsky and A. Zisserman. Learning to count objects in images. In *NIPS*, 2010.

[16] A. Marana, S. Velastin, L. Costa, and R. Lotufo. Estimation of crowd density using image processing. In *Image Processing for Security Applications, IEE Colloquium on*, 1997.

[17] J. Matas, O. Chum, M. Urban, and T. Pajdla. Robust wide-baseline stereo from maximally stable extremal regions. *Image and Vision Computing*, 2004.

[18] J. Matas and K. Zimmermann. A new class of learnable detectors for categorisation. In *SCIA*, 2005.

[19] L. Neumann and J. Matas. Text localization in real-world images using efficiently pruned exhaustive search. In *ICDAR*, 2011.

[20] J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kauffman, California, 1988.

[21] H. Riemenschneider, S. Sternig, M. Donoser, P. Roth, and H. Bischof. Hough regions for joining instance localization and segmentation. In *Proc. ECCV*, 2012.

[22] D. Ryan, S. Denman, C. Fookes, and S. Sridharan. Crowd counting using multiple local features. In *Proc. DICTA*, 2009.

[23] I. Tsochantaridis, T. Hofmann, T. Joachims, and Y. Altun. Support vector machine learning for interdependent and structured output spaces. In *Proc. ICML*, 2004.

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

[25] C. Yu and T. Joachims. Learning structural SVMs with latent variables. In *Proc. ICML*, 2009.