Learning to Co-Generate Object Proposals with a Deep Structured Network

Zeeshan Hayder[1,2], Xuming He[2,1]
[1]Australian National University & [2]NICTA *
{zeeshan.hayder, xuming.he}@anu.edu.au

Mathieu Salzmann[1,3]
[3]CVLab, EPFL, Switzerland
mathieu.salzmann@epfl.ch

Abstract

Generating object proposals has become a key component of modern object detection pipelines. However, most existing methods generate the object candidates independently of each other. In this paper, we present an approach to co-generating object proposals in multiple images, thus leveraging the collective power of multiple object candidates. In particular, we introduce a deep structured network that jointly predicts the objectness scores and the bounding box locations of multiple object candidates. Our deep structured network consists of a fully-connected Conditional Random Field built on top of a set of deep Convolutional Neural Networks, which learn features to model both the individual object candidates and the similarity between multiple candidates. To train our deep structured network, we develop an end-to-end learning algorithm that, by unrolling the CRF inference procedure, lets us backpropagate the loss gradient throughout the entire structured network. We demonstrate the effectiveness of our approach on two benchmark datasets, showing significant improvement over state-of-the-art object proposal algorithms.

1. Introduction

Generating object proposals has recently become one of the key components of modern object detection techniques [7, 9, 10, 6]. By filtering out the irrelevant portions of the input image, these proposals hugely reduce the search space of object detectors, which has proven beneficial for both speed an accuracy.

Existing object proposal methods [1, 4, 15, 18, 22, 25, 12], however, all generate candidate detections one by one, independently of each other. By contrast, in a parallel line of research, object co-detection [2, 8, 10, 11] has emerged as an effective approach to leveraging the information jointly contained in multiple images to improve detection accuracy. Unfortunately, to model the similarity of multiple objects, existing methods rely on either handcrafted features [2, 8, 20, 11], or features learned for object recognition [10]. As a consequence, they are ill-suited to handle general object proposals, whose appearance is subject to much larger variations than specific object classes.

In this paper, we introduce an approach to co-generating object proposals in multiple images. To this end, we propose a deep structured network that lets us learn features to model (i) the appearance of individual object candidates; and (ii) the similarity between multiple object candidates. As a result, our model is able to leverage the collective power of multiple object candidates, while coping with the large appearance variability of general object proposals.

More specifically, given an initial pool of object candidates, our model consists of a fully-connected Conditional Random Field (CRF) built on top of a set of deep Convolutional Neural Networks (CNNs), one for each candidate. The CNN module of each candidate predicts (i) an objectness score; (ii) a bounding box location; and (iii) a low-dimensional feature vector employed in the pairwise term of the CRF. This pairwise term takes the form of a Gaussian kernel, which allows us to perform inference efficiently [13], even for large numbers of candidates. Altogether, the resulting deep structured model jointly produces improved objectness scores for multiple candidates and refined locations for the foreground objects.

We introduce an end-to-end learning algorithm to estimate the weights of our deep structured network. To this end, we follow a stochastic gradient descent procedure using mini-batches on which we define the CRF. By unrolling the iterations of our CRF inference strategy, we can backpropagate the gradient of our loss function throughout the entire structured network. This lets us learn the similarity of pairs of candidates, thus effectively benefitting from multiple candidates to co-generate high-quality object proposals.

We demonstrate the effectiveness of our approach on two benchmark datasets for object proposal generation: Pascal VOC 2007 [5] and MS COCO [17]. Our experiments evidence the benefits of leveraging multiple images for object proposal generation over state-of-the-art methods that generate the proposals individually.

*NICTA is funded by the Australian Government as represented by the Department of Broadband, Communications and the Digital Economy and the ARC through the ICT Centre of Excellence program. The Tesla K40 used for this research was donated by the NVIDIA Corporation.

2. Related Work

Objectness has essentially lead to a paradigm shift in object detection. Instead of the traditional sliding window approach, objectness facilitates detection by proposing a smaller number of interesting candidate regions. These object proposals have now become ubiquitous in state-of-the-art detectors [7, 9, 6]. While object proposals have also been considered in the context of depth images [24], below, we focus on methods designed for RGB images, which are more common for large-scale object detection.

Most objectness methods rely on well-engineered handcrafted features. For instance, Alexe et al. [1] introduced a generic objectness measure using four image cues, including multi-scale saliency, color contrast, edge density and superpixel straddleness. Instead of performing exhaustive search over the image, the Selective Search method of Uijlings et al. [22] utilizes an image over-segmentation. This method achieves high accuracy, and is therefore widely used as an initial step for detection. Krähenbühl et al. [15] introduced a fast method based on the geodesic distance transform, which can be computed in near-linear time and generates object proposals at different scales. To date, the fastest method to generate object proposals is that of Cheng et al. [4]. To achieve speed, this method relies on a binary representation of gradient-based features. This speed, however, comes at some loss in object localization accuracy. By contrast, Zitnick et al. [25] exploited the edges and edge groups at the object boundaries to better localize the objects and generate good-quality proposals.

Recently, Pinheiro et al. [18] proposed to go beyond handcrafted features for object proposal generation. In particular, [18] leverages the representation power of deep networks to learn a discriminative CNN that generates boxes and segmentation proposals. Ultimately, this method achieves state-of-the-art accuracy and competitive speed.

While effective, all existing objectness methods essentially generate one proposal at a time, without considering interactions between the proposals beyond simple exclusion via non-maximum suppression. By contrast, object co-detection [2] attempts to simultaneously exploit the similarity between pairs of objects to perform detection jointly in multiple images. Most existing co-detection methods rely on simple handcrafted features to model object similarity [2, 8, 20, 11]. By contrast, in our previous work [10], we performed feature selection using pre-trained CNN features. In both cases, however, the resulting techniques are ill-suited to produce general object proposals, because the employed features are tuned to the problem of detecting specific objects. Here, instead, we co-generate object proposals from multiple images by introducing a deep structured network that lets us learn general object features, as well as pairwise features to model proposal similarity. To the best of our knowledge, leveraging the power of multiple images has never been achieved in the context of objectness.

The idea of deep structured neural networks has nonetheless been exploited in the past and can be traced back to the 90s [16]. More recently, such structured networks have been exploited for the task of object recognition and semantic segmentation [3, 19, 23]. In this context, Schwing et al. [19] and Zheng et al. [23] have also proposed to exploit the efficient mean-field inference procedure of [13]. However, these approaches, beside tackling a different problem than ours, still rely on simple handcrafted features in their pairwise term. By contrast, here, in addition to the unary features, we also learn pairwise features that are back-propagated throughout the entire network. As demonstrated by our experiments, by learning this similarity, we effectively exploit the joint information of multiple object candidates to produce high-quality object proposals.

3. Co-Generating Object Proposals

We tackle the problem of jointly predicting a set of object proposals from multiple images[1]. Ultimately, our goal is to leverage the collective information contained in a set of object candidates to obtain high-quality object proposals. To this end, we start from an initial pool of object candidates and develop a deep structured network that improves their ranking and localization. This deep structured network models both the appearance of each object candidate and the interactions between these candidates.

Formally, let $\mathbf{X} = \{\mathbf{x}_1, \cdots, \mathbf{x}_N\}$ be an initial pool of object candidates generated from a set of images $\mathcal{I}$, typically by an existing object proposal method, such as Bing or Selective Search. Each object candidate $\mathbf{x}_i$ denotes an image window cropped from one of the images in $\mathcal{I}$. For each $\mathbf{x}_i$, we introduce (i) a binary variable y_i, which indicates whether $\mathbf{x}_i$ is a foreground object ($y_i = 1$) or background clutter ($y_i = 0$); and (ii) a continuous real-valued vector $\mathbf{t}_i = (t_{i,x}, t_{i,y}, t_{i,w}, t_{i,h})^T$ containing the offset to the true object location if $\mathbf{x}_i$ is a foreground object and 0 otherwise.

We formulate the co-generation of object proposals for image set $\mathcal{I}$ as a multi-label prediction problem, in which we simultaneously predict the labels $\mathbf{Y} = \{y_1, \cdots, y_N\}$ and the location offsets $\mathbf{T} = \{\mathbf{t}_1, \cdots, \mathbf{t}_N\}$ of the object candidate set $\mathbf{X}$. To this end, we develop a deep structured network that defines a joint distribution over $\mathbf{Y}$ and $\mathbf{T}$ given $\mathbf{X}$, denoted by $P(\mathbf{Y}, \mathbf{T}|\mathbf{X})$. Our deep structured network consists of two components: One CNN for each object candidate, with weights shared across all candidates, which provides a general object representation, and one fully-connected CRF, which captures the similarity between every pair of object candidates.

Specifically, the joint distribution defined by the deep

[1]Note that our approach also applies to the multiple proposals of a single image.

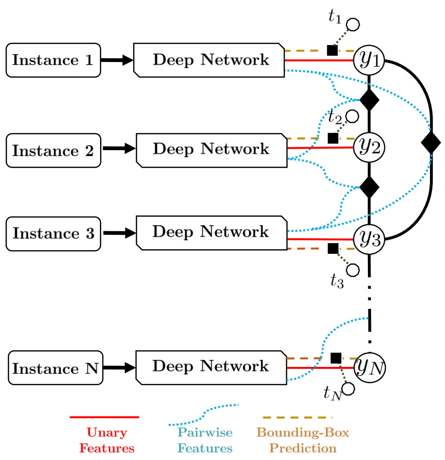

Figure 1: **Overview of our deep structured network for object proposal co-generation.** Our model consists of one deep CNN module per object candidate, linked by a fully-connected CRF.

structured network can be written as

$$P(\mathbf{Y}, \mathbf{T}|\mathbf{X}) = \frac{1}{Z(\mathbf{X})} \exp\left(-\sum_{i=1}^{N} \phi(y_i, \mathbf{t}_i|\mathbf{x}_i) - \sum_{i=1}^{N}\sum_{j>i} \psi(y_i, y_j|\mathbf{x}_i, \mathbf{x}_j) \right), \quad (1)$$

where $Z(\cdot)$ is the partition function, and ϕ, ψ are the unary and pairwise potential functions, respectively. The unary potential ϕ encodes how likely a candidate $\mathbf{x}_i$ is to be assigned label y_i with location offset $\mathbf{t}_i$, while the pairwise potential ψ is a symmetric term encouraging any two similar candidates to have the same label assignment.

To fully leverage the representative power of CNNs, we make both the unary and pairwise potentials depend on the CNN modules of our deep structured network. Intuitively, and as illustrated by Fig. 1, a CNN module corresponding to candidate $\mathbf{x}_i$ produces features for the unary term of $\mathbf{x}_i$ and features for the pairwise terms involving $\mathbf{x}_i$. Furthermore, the pairwise term connects multiple CNN modules to form a structured prediction model. In the remainder of this section, we describe our network architecture and potential functions in details.

3.1. Deep CNNs for Individual Object Candidates

The network architecture constituting the CNN module for each individual object candidate $\mathbf{x}_i$ is depicted by Fig. 2. As mentioned above, this CNN module produces (i) a unary term consisting of an objectness score and of a refined object location; and (ii) a feature vector for the pairwise term. To this end, the output of this network consists of three sibling layers. The first one relies on a softmax layer to predict the foreground/background probabilities; the second one makes use of a regression layer that outputs the four real-valued coordinates of the foreground location offset; and the third one employs a fully-connected layer to generate a low-dimensional feature vector for the CRF pairwise terms.

More specifically, let us denote by $\mathbf{f}_i^{net}$ the output of the $\mathbf{FC}_7$ layer of the CNN for object candidate $\mathbf{x}_i$. The three network outputs are computed as

$$P_u(y_i) \propto \exp(\mathbf{w}_{u,y_i}^T \mathbf{f}_i^{net}) \quad (2)$$

$$\tilde{\mathbf{t}}_i = \mathbf{W}_r^T \mathbf{f}_i^{net} [\![y_i = 1]\!] \ , \quad (3)$$

$$\mathbf{h}_i^p = \mathbf{W}_p^T \mathbf{f}_i^{net}, \quad (4)$$

where $\mathbf{w}_{u,y_i}$, $\mathbf{W}_r$ and $\mathbf{W}_p$ denote the fully-connected weights to generate the label probabilities, the estimated object location offsets and the features for the pairwise potential, respectively. By $[\![y_i = 1]\!]$, we mean that the estimated offset will be $\mathbf{W}_r^T \mathbf{f}_i^{net}$ if the predicted label $y_i = 1$, and 0 otherwise. This lets us write our unary potential as

$$\phi(y_i, \mathbf{t}_i|\mathbf{x}_i) = -\mathbf{w}_{u,y_i}^T \mathbf{f}_i^{net} + \|\mathbf{t}_i - \mathbf{W}_r^T \mathbf{f}_i^{net} [\![y_i = 1]\!]\|^2, \quad (5)$$

which measures the cost for a candidate $\mathbf{x}_i$ to belong to the foreground/background class and have offset $\mathbf{t}_i$.

3.2. Fully-connected CRF for Candidate Similarity

On top of the deep CNN modules, we construct a fully-connected CRF, which models inter-candidate similarity. To this end, we define the pairwise potential ψ as a data-dependent smoothing term that encourages similar object candidates to share the same label. As in [13], we restrict the data-dependent weight in the pairwise potential to take the form of a Gaussian kernel. This yields

$$\psi(y_i, y_j|\mathbf{X}_i, \mathbf{X}_j) = \mu(y_i, y_j) k(\mathbf{h}_i^p, \mathbf{h}_j^p) = \mu(y_i, y_j) k(\mathbf{W}_p^T \mathbf{f}_i^{net}, \mathbf{W}_p^T \mathbf{f}_j^{net}) \ , \quad (6)$$

with

$$k(\mathbf{h}_i^p, \mathbf{h}_j^p) = \exp\left(-\frac{1}{2} \|\mathbf{h}_i^p - \mathbf{h}_j^p\|^2) \right) \ , \quad (7)$$

where $\mathbf{h}^p$ is defined in Eq. 4, and μ is a label compatibility function. While a general compatibility function can be learned [14], in practice, we found that a Potts model, *i.e.*, $\mu(y_i, y_j) = [\![y_i \neq y_j]\!]$, was already effective.

Our deep structured model differs from the existing fully-connected CRFs for semantic labeling in several ways. First, since our nodes correspond to object proposals, our CRF implements multiple tasks, including labeling object candidates and refining their locations. In addition, we do not rely on the traditional bilateral kernels, which use manually selected features. Instead, we learn a low-dimensional feature representation that, when used in a Gaussian kernel, is able to encode candidate similarity. As a side effect, we do not need to define a covariance matrix for the kernel, since the weights $\mathbf{W}_p$ implicitly handle this. This simplifies the end-to-end learning of the full network.

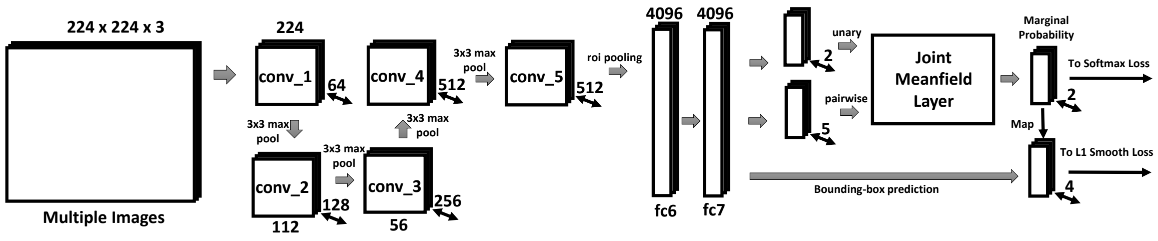

Figure 2: **Detailed architecture of our deep structured network for object proposal co-generation.** Each input image first goes through a series of convolutional layers, followed by Region-of-Interest (RoI) pooling corresponding to the different candidates in the image. Each candidate then passes through several fully-connected layers to predict unary features, pairwise features and bounding box location offsets. The features are finally employed in a fully-connected CRF. During training, our model makes use of a multi-task loss.

3.3. Efficient Object Proposal Co-Generation

Given our deep structured model, we co-generate object proposals by taking a set of initial object candidates as input, and jointly inferring the MAP estimates of the objects' labels and location offsets, as well as the posterior marginal probabilities of the labels. Note that the MAP estimates of the labels and location offsets are decoupled. Indeed, for any label assignment y_i, the minimizer of the second term in Eq. 5 is given by

$$\mathbf{t}_i^* = \mathbf{W}_r^T \mathbf{f}_i^{net} [\![y_i = 1]\!] \ , \tag{8}$$

since $\mathbf{t}_i$ does not appear in the pairwise terms. Therefore, we can first compute the MAP label assignment and obtain the location offsets from Eq. 8.

To compute the MAP and posterior marginals of the label variables, we make use of the same efficient mean-field inference as in [13]. Specifically, we approximate the joint label probability by a factorized distribution $Q(\mathbf{Y}) = \prod_i \mathbf{q}_i(y_i)$. The mean-field inference updates approximate the marginals iteratively as

$$\mathbf{q}_i^{(t)}(y_i) = \frac{1}{Z_i} \exp\big(\mathbf{w}_{u,y_i}^T \mathbf{f}_i^{net} \tag{9}$$
$$- \sum_{j \neq i} \sum_{y_j} \mu(y_i, y_j) k(\mathbf{h}_i^p, \mathbf{h}_j^p) \mathbf{q}_j^{(t-1)}(y_j)\big) \ ,$$

where t denotes the iteration step, and Z_i is the approximate partition function (*i.e.*, the normalizer). These updates can be computed efficiently for a large number of object candidates using a fast Gaussian filtering technique. After performing inference, the MAP estimate of the object label is approximated by $y_i^* = \arg\max_{y_i} \mathbf{q}_i(y_i)$. We use the (approximate) marginal probabilities of the foreground class as scores to generate the final ranking of the object proposals.

4. Learning our Deep Structured Network

We now develop an end-to-end learning method to estimate the parameters of our multi-output deep structured network. To this end, let $\mathcal{D} = \{\mathbf{X}, \hat{\mathbf{Y}}, \hat{\mathbf{T}}\} = \{(\mathbf{x}_i, \hat{y}_i, \hat{\mathbf{t}}_i)\}_{i=1}^N$ be a set of object candidates extracted from training images with ground-truth object bounding boxes. Note that this training set contains both foreground objects and background clutter, obtained with the same objectness algorithm as at test time. The offsets of these candidates are normalized by the size of the bounding boxes (scale-invariant) and expressed in log-space. Furthermore, to handle the iterative nature of mean-field inference, we unroll its iterations and denote the output of the final one by $\{\mathbf{q}_i^{(m)}\}_{i=1}^N$.

To train our deep structured network, we employ a multi-task loss L, which combines a classification loss L_y for the object labels and a regression loss L_r for the box offsets. This loss can be expressed as

$$L(\mathcal{D}) = L_y(Q^m, \hat{\mathbf{Y}}) + \lambda L_r(\mathbf{T}, \hat{\mathbf{T}}) \tag{10}$$
$$= -\sum_i \Big[\log \mathbf{q}_i^{(m)}(\hat{y}_i) + \lambda [\![\hat{y}_i = 1]\!] \|\mathbf{t}_i - \hat{\mathbf{t}}_i\|_{SL_1} \Big] \ ,$$

where $\mathbf{T} = \{\mathbf{t}_i\}$ is the offset predictions from the network, and λ is a hyperparameter balancing the two tasks. For offset regression, Eq. 10 makes use of a smoothed L_1 loss, denoted by $\|\cdot\|_{SL_1}$ and defined as

$$\|\mathbf{z}\|_{SL_1} = \sum_k 0.5 z_k^2 [\![|z_k| < 1]\!] + (|z_k| - 0.5) [\![|z_k| \geq 1]\!] \ .$$

Our deep structured network has four sets of parameters, including the network weights $\mathbf{W}_{cnn}$ before the $\mathbf{FC}_7$ layer of the CNN module, the unary term weights $\mathbf{W}_u$, the regression weights $\mathbf{W}_r$ and the pairwise feature weights $\mathbf{W}_p$. We use stochastic gradient descent (SGD) to train those weights in an end-to-end manner. The overall training procedure consists of two steps: We first pre-train the CNN modules and then train the full structured model using mini-batches.

4.1. Pre-training the Deep CNN Module

In a first stage, we train the CNN module corresponding to the individual object candidates. In other words, in this stage, we ignore the pairwise features and the dense CRF layer. The CNN module has two outputs: the object label probability $P_u(y_i)$ and the bounding box location offset $\mathbf{t}_i$. We train it using the same procedure as the Fast RCNN [6].

More specifically, we start from a convolutional neural network (VGG-16 [21]) pre-trained on ImageNet, which gives us an initialization for the weights $\mathbf{W}_{cnn}$. We then define our training loss as

$$L_c = -\sum_i \left[\log P_u(\hat{y}_i) + \lambda[\![\hat{y}_i > 0]\!]\|\mathbf{t}_i - \hat{\mathbf{t}}_i\|_{SL_1}\right], \quad (11)$$

and adopt the same strategy of mini-batch sampling and back-propagation through RoI pooling layers as in [6]. This pre-training step initializes the unary and regression weights ($\mathbf{W}_u$ and $\mathbf{W}_r$), and fine-tunes the network ones ($\mathbf{W}_{cnn}$).

4.2. End-to-end Learning with Mini-batches

In a second stage, given the weight initializations described above, we learn our complete deep structured network. To this end, we follow an SGD procedure using mini-batches on which we define the fully-connected CRF. The main challenge of this procedure is to derive the gradient of the loss function $L(\mathcal{D})$ with respect to the weight parameters and to compute this gradient efficiently. Below, we will focus on the gradient of L_y, since the second term L_r, not involving the CRF, can be handled in the same manner as in Fast RCNN (as in the pre-training of Section 4.1).

We derive a mean-field gradient method which computes the parameter gradients of our deep structured model recursively, similarly to [14]. Note that, unlike [14], we need to compute the gradient w.r.t. the unary *and pairwise* weights $\mathbf{W}_u$ and $\mathbf{W}_p$, as well as the network features $\mathbf{f}_i^{net}$ in order to backpropagate the gradient to the CNN modules.

As before, let us denote the marginals by $\mathbf{q} = (\mathbf{q}_1^T, \cdots, \mathbf{q}_N^T)^T$. The gradient of the loss L_y w.r.t. a parameter $\mathbf{w}$ can be written as

$$\frac{\partial L_y}{\partial \mathbf{w}} = \frac{\partial L_y}{\partial \mathbf{q}}\frac{\partial \mathbf{q}^T}{\partial \mathbf{w}}. \quad (12)$$

Let $\mathbf{u} = (\mathbf{u}_1^T, \cdots, \mathbf{u}_N^T)^T$ be the unary term for the label variables, where $\mathbf{u}_i = -[\mathbf{w}_{u,0}, \mathbf{w}_{u,1}]^T\mathbf{f}_i^{net}$, and $\hat{\Psi}$ be the matrix form of the pairwise term, *i.e.*, $\hat{\Psi} = \mathbf{K} \otimes \mu$, where $\mathbf{K} = [k_{ij}]_{N\times N}$ is the kernel matrix and $k_{ij} = k(\mathbf{h}_i^p, \mathbf{h}_j^p)$. Following [14], the gradient can be recursively computed as

$$\frac{\partial L_y(\mathbf{q}^m(\mathbf{w}))}{\partial \mathbf{w}} = \sum_{t=1}^{m} \mathbf{b}^{(t)T}\left(\frac{\partial \mathbf{u}}{\partial \mathbf{w}} + \mathbf{q}^{(t-1)}\frac{\partial \hat{\Psi}}{\partial \mathbf{w}}\right), \quad (13)$$

where $\mathbf{b}^{(t)} = (\mathbf{b}_1^{(t),T}, \cdots, \mathbf{b}_N^{(t),T})^T$ is the normalized loss gradient at iteration t. This normalized loss gradient is defined recursively as

$$\mathbf{b}^{(m)} = A^{(m)}\left(\nabla L_y(\mathbf{q}^{(m)})\right)^T \quad (14)$$

$$\mathbf{b}^{(t)} = A^{(t)}\hat{\Psi}\mathbf{b}^{(t+1)}, \ t = 1, \cdots, m-1, \quad (15)$$

where $A^{(t)}$ is a block diagonal matrix with blocks $A_i^{(t)} = \mathbf{q}_i^{(t)}\mathbf{q}_i^{(t)^T} - \text{diag}(\mathbf{q}_i^{(t)})$. We now derive the two partial derivatives in Eq. 13 for different weights and features.

Unary weights and features. The unary weight matrix $\mathbf{W}_u$ only appears in the first term of Eq. 13. This term can be computed as

$$\frac{\partial \mathbf{b}^T\mathbf{u}}{\partial \mathbf{W}_u} = \sum_i \mathbf{b}_i\mathbf{f}_i^{net\,T}. \quad (16)$$

The gradient of the unary term w.r.t. the deep network features $\mathbf{f}_i^{net}$ can be computed similarly as

$$\frac{\partial \mathbf{b}^T\mathbf{u}}{\partial \mathbf{f}_i^{net}} = \mathbf{W}_u^T\mathbf{b}_i. \quad (17)$$

Note that this feature gradient will be combined with the feature gradient of the pairwise term derived below.

Pairwise weights and features. For the pairwise term, we apply the chain-rule and first compute the gradient w.r.t. the pairwise features $\mathbf{h}_i^p$. This yields

$$\begin{aligned}\frac{\partial \mathbf{b}^T\hat{\Psi}\mathbf{q}}{\partial \mathbf{h}_i^p} &= \frac{\partial}{\partial \mathbf{h}_i}\left(\mathbf{b}^T\mathbf{K}^{(m)} \otimes \mu^{(m)}\mathbf{q}\right)\\ &= \sum_j(\mathbf{h}_j^p - \mathbf{h}_i^p)\mathbf{q}_j^T\mathbf{K}_{ij}^{(m)}\mu^{(m)}\mathbf{b}_i \quad (18)\\ &= \sum_j \mathbf{h}_j^p\mathbf{q}_j^T\mathbf{K}_{ij}^{(m)}\mu^{(m)}\mathbf{b}_i - \mathbf{h}_i^p\mathbf{b}_i^T\sum_j\mathbf{K}_{ij}^{(m)}\mu^{(m)}\mathbf{q}_j.\end{aligned}$$

This gradient can be computed efficiently using high-dimensional filtering on the permutohedral lattice [13].

The gradient w.r.t. the pairwise weight matrix $\mathbf{W}_p$ and the CNN features $\mathbf{f}_i^{net}$ can be computed as

$$\frac{\partial \mathbf{b}^T\hat{\Psi}\mathbf{q}}{\partial \mathbf{W}_p} = \sum_i \frac{\partial \mathbf{b}^T\hat{\Psi}\mathbf{q}}{\partial \mathbf{h}_i^p}\frac{\partial \mathbf{h}_i^{p\,T}}{\partial \mathbf{W}_p} = \sum_i \frac{\partial \mathbf{b}^T\hat{\Psi}\mathbf{q}}{\partial \mathbf{h}_i^p}\mathbf{f}_i^{net\,T} \quad (19)$$

$$\frac{\partial \mathbf{b}^T\hat{\Psi}\mathbf{q}}{\partial \mathbf{f}_i^{net}} = \frac{\partial \mathbf{b}^T\hat{\Psi}\mathbf{q}}{\partial \mathbf{h}_i^p}\frac{\partial \mathbf{h}_i^{p\,T}}{\partial \mathbf{f}_i^{net}} = \mathbf{W}_p^T\frac{\partial \mathbf{b}^T\hat{\Psi}\mathbf{q}}{\partial \mathbf{h}_i^p}. \quad (20)$$

Altogether, we can compute all the gradients in a single forward and backward pass over our full deep structured network. The overall mean-field gradient computation framework is summarized in Algorithm 1. In practice, we use different step sizes for the unary, pairwise, regression and CNN weights, which helps the learning procedure focus on the pairwise and regression weights, while only fine-tuning the rest of the network.

5. Experiments

In this section, we demonstrate the effectiveness of our method on the problem of large-scale proposal generation for generic objects. To this end, we evaluate our approach on two challenging datasets with multiple object classes, *i.e.*, PASCAL VOC 2007 and Microsoft COCO, and compare our results with those of the state-of-the-art object proposal methods.

Algorithm 1 Mean-field Gradient for Classification Loss

```
Input: Features $\{\mathbf{f}_i^{net}\}_{i=1}^N$, Initial Weights $\mathbf{W}_u$, $\mathbf{W}_p$, Mean-field
    Iterations $m$, Loss Function $L_y$
Output: Feature and Weight Gradients: $\mathbf{g}_w^u$, $\mathbf{g}_w^p$, $\mathbf{g}_{f,i}^{net}$
Phase 1: Joint Inference
 1: procedure (Forward pass)
 2:     $\mathbf{u}_i = \mathbf{q}_i^0 = -\mathbf{W}_u^T \mathbf{f}_i^{net}$
 3:     $\mathbf{h}_i^p = \mathbf{W}_p^T \mathbf{f}_i^{net}$
 4:     for t = 1 ... $m$ do
 5:         $\mathbf{q}_i^{(t)} = \frac{1}{Z} \exp\left(\mathbf{u}_i - \sum_{j \neq i} k(\mathbf{h}_i^p, \mathbf{h}_j^p) \mu \mathbf{q}_j^{(t-1)}\right)$
 6:     end for
 7: end procedure
Phase 2: Gradient Computation
 8: procedure (Backward pass)
 9:     $\mathbf{g}_w^u \leftarrow 0$
10:     $\mathbf{g}_w^p \leftarrow 0$
11:     $\mathbf{g}_f^{net} \leftarrow 0$
12:     $\mathbf{A}_i^{(m)} = \mathbf{q}_i^{(m)} \mathbf{q}_i^{(m)^T} - \mathrm{diag}(\mathbf{q}_i^{(m)})$
13:     $\mathbf{b}^{(m-1)} = \mathbf{A}^{(m)} \left(\nabla L_y(\mathbf{q}^{(m)})^T\right)$
14:     for t = $m - 1$ ... 1 do
15:         $\mathbf{A}_i^{(t)} = \mathbf{q}_i^{(t)} \mathbf{q}_i^{(t)^T} - \mathrm{diag}(\mathbf{q}_i^{(t)})$
16:         $\mathbf{g}_w^u \leftarrow \mathbf{g}_w^u + \frac{\partial}{\partial \mathbf{W}_u}\left(\mathbf{b}^{(t)^T} \mathbf{u}\right)$            ▷ Eq. 16
17:         $\mathbf{g}_{f,i}^{net} \leftarrow \mathbf{g}_{f,i}^{net} + \frac{\partial}{\partial \mathbf{f}_i^{net}}\left(\mathbf{b}^{(t)^T} \mathbf{u}\right)$            ▷ Eq. 17
18:         $\mathbf{g}_w^p \leftarrow \mathbf{g}_w^p + \frac{\partial}{\partial \mathbf{W}^p}\left(\mathbf{b}^{(t)^T} \hat{\Psi} \mathbf{q}^{(t)}\right)$            ▷ Eq. 19
19:         $\mathbf{g}_{f,i}^{net} \leftarrow \mathbf{g}_{f,i}^{net} + \frac{\partial}{\partial \mathbf{f}_i^{net}}\left(\mathbf{b}^{(t)^T} \hat{\Psi} \mathbf{q}^{(t)}\right)$            ▷ Eq. 20
20:         $\mathbf{b}^{(t-1)} = \mathbf{A}^{(t)} \hat{\Psi} \mathbf{b}^{(t)}$
21:     end for
22: end procedure
```

5.1. Datasets and Setup

The Pascal VOC 2007 dataset [5] comprises 5011 training-validation (trainval) images and 4952 test images, and the Microsoft COCO 2014 validation dataset [17] contains 82783 training images and 40504 validation images. For Pascal VOC 2007, we used all the trainval data to learn our deep structured model, and evaluated it using all the test data. For Microsoft COCO, we also used all the training images to learn our model, but, following [18], used only the first 5000 validation images for evaluation purpose.

In our experiments, we evaluated several techniques to generate the initial set of candidates. The choice of the particular initial proposal generation methods we use was motivated by the fact that [12], whose protocol we follow, and other existing methods used them for these datasets. In particular, we used Bing [4] for both datasets, as well as Selective Search [22] for Pascal and Edge Box [25] for MS COCO, which represent the most commonly-used methods for each dataset, respectively. For training, we obtained the mini-batches by randomly sampling 2 images from the training set and taking 512 candidates per image. At test time, given the initial candidates of all the test images, we extracted the unary features, bounding box location offsets and pairwise features for each candidate using the deep CNN, and then performed inference in the fully-connected CRF using all the candidates. This allows us to truly exploit the similarities across all the test data, and, thanks to the efficient inference procedure, remains fast (e.g., 1.4 sec. for roughly 10k Bing candidates per image).

The standard error measures to evaluate object proposal quality rely on Average Recall (AR). It has been shown that, for a fixed number of proposals per image, AR correlates well with the mean Average Precision of the object detector applied to the proposals [12]. We therefore report our results using the average recall metrics defined by Hosang et al. [12] and the COCO-style metrics [17] used in [18].

5.2. Results on VOC 2007

In Table 1, we provide a quantitative comparison of our approach with state-of-the-art objectness methods according to the criteria of [17] on the Pascal VOC 2007 dataset. The results of these baselines were directly reproduced from their respective papers. Note that our approach yields state-of-the-art results; for 10 and 100 proposals, when using Bing candidates, and for 1000 proposals, when using Selective Search candidates. Table 1 also shows that, when selecting 100 proposals, our co-generation approach yields consistent improvement across different sizes of objects. In terms of runtime, our method remains highly competitive. Altogether, we believe that these results clearly evidence the benefits of co-generating the object proposals.

The results of our approach and of the baselines according to the criteria defined by Hosang et al. [12] are shown in Fig. 3[2]. In particular, the top row of Fig. 3 depicts the recall as a function of the Intersection over Union (IoU) threshold when using the 10, 100, 1000 & 10000 highest-scoring bounding boxes per image, respectively. The bottom row of Fig. 3 shows the AR and the recall as a function of the number of proposals for IoU thresholds of 0.5, 0.7 and 0.8, respectively. Again, these curves clearly show that co-generating the proposals yields to much better object bounding boxes. In particular, it is interesting to note that, even though the AR of the initial Bing bounding boxes is quite low, it is boosted to state-of-the-art results after our deep structured co-generation process. Note also that, while initially better, the Selective Search candidates still benefit from our approach. Interestingly, at high IoU, our results with Bing candidates tend to outperform our results with Selective Search candidates.

To evidence that our approach is not simply learning the

[2]Note that we do not have access to the code, or the bounding boxes, of [18], and were thus unable to compute these curves for their approach.

PASCAL VOC07	AR@10	AR@100	AR@1000	AR@Small	AR@Medium	AR@Large	Time (sec)
Bing	0.141	0.262	0.344	0.000	0.083	0.369	0.003
EdgeBoxes	0.203	0.407	0.601	0.035	0.159	0.559	0.25
Geodesic	0.121	0.364	0.596	-	-	-	1.7
Selective Search	0.085	0.347	0.618	0.017	0.134	0.364	10
MCG	0.232	0.462	0.634	0.073	0.228	0.618	30
Deep-Mask	0.337	0.561	0.690	-	-	-	1.2
Ours Co-Obj (Sel. Search)	0.325	0.509	**0.745**	0.114	0.321	0.629	1.1
Ours Co-Obj (Bing)	**0.430**	**0.602**	0.675	**0.453**	**0.517**	**0.654**	1.4

Table 1: **AR analysis on the PASCAL VOC 2007 test set:** We compare our method with state-of-the-art object proposal baselines according to the criteria of [17]. The results of our approach are provided in Rows 7-8 when using Bing and Selective Search to generate the initial candidates, respectively. The AR for small, medium and large objects were computed for 100 proposals. Note that our co-generation approach outperforms the state-of-the-art baseline in all metrics. The difference in speed between two versions of our approach is due to the fact that Bing yields a larger candidate pool than Selective Search.

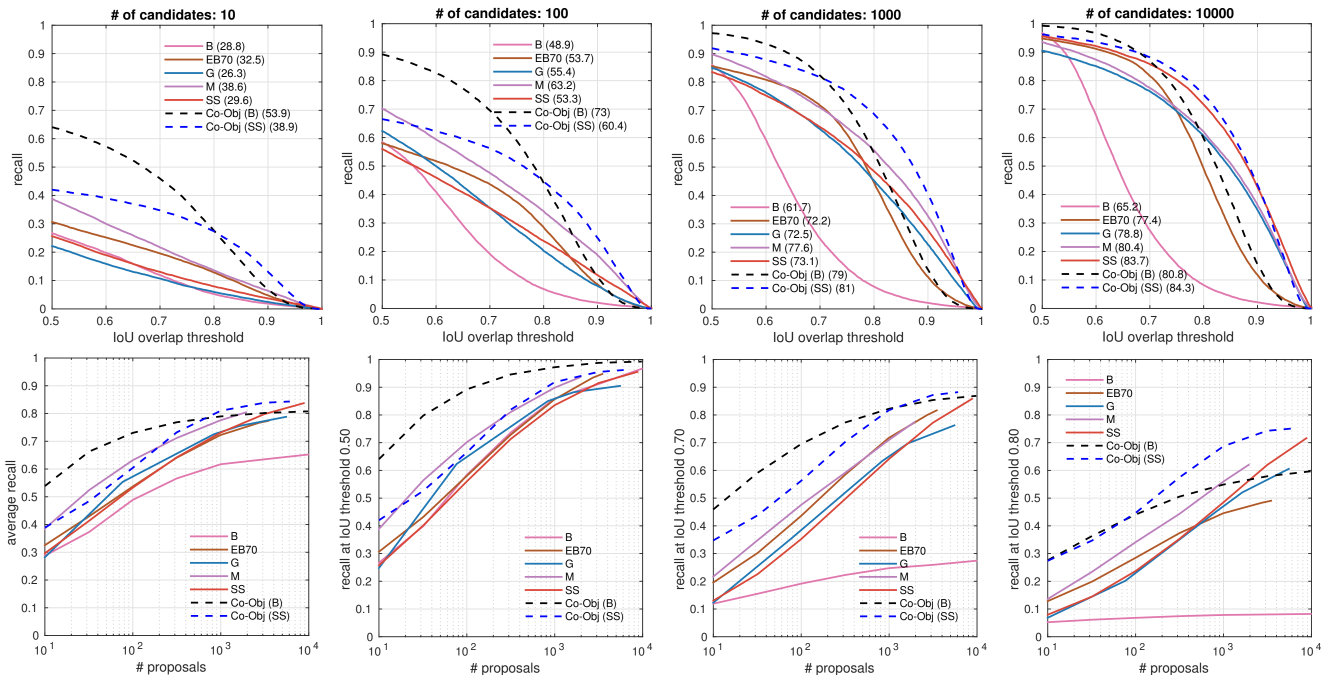

Figure 3: **Pascal VOC 2007 test:** We compare our method with state-of-the-art object proposal baselines according to the criteria of [12]. **Top: Recall v.s. IoU threshold.** These recall curves were generated using the highest-scoring 10, 100, 1000 and 10000 object proposals, respectively. **Bottom: Recall v.s. Number of Proposals.** The first plot shows the AR, and the remaining recall curves were generated using IoU thresholds of 0.5, 0.7 and 0.8, respectively. In all the plots, the dashed lines correspond to our co-generation results, in blue when using Selective Search candidates (**Co-Obj (SS)**) and in black when using Bing candidates (**Co-Obj (B)**). The baselines correspond to Bing (**B**), EdgeBoxes (**EB70**), Geodesic (**G**), MCG (**M**) and SelectiveSearch (**SS**). These results clearly evidence the benefits of our co-generation approach.

behavior of a particular proposal method, we evaluated it with different proposal generation techniques during training and test time. As shown in Table 2, our method still outperforms the initial proposal generation methods, thus evidencing that it truly learns the relevant context for the object candidates themselves. We acknowledge, however, that the best results are obtained when using the same method at training and test time.

PASCAL VOC07	Train	Test	AR@10	AR@100	AR@1000
Bing	-	-	0.141	0.262	0.344
MCG	-	-	0.232	0.462	0.634
Ours Co-Obj	MCG	MCG	0.365	0.573	**0.709**
Ours Co-Obj	MCG	Bing	0.350	0.481	0.558
Ours Co-Obj	Bing	MCG	0.392	0.517	0.641
Ours Co-Obj	Bing	Bing	**0.430**	**0.602**	0.675

Table 2: **Using different initial proposal methods:** Rows 1-2 show the baseline object proposal methods. Rows 3-6 show our results using various candidate generation options at training and test time.

Microsoft COCO 2014	AR@10	AR@100	AR@1000	AR@Small	AR@Medium	AR@Large
Bing	0.042	0.100	0.189	0.001	0.063	0.319
EdgeBoxes	0.074	0.178	0.338	0.015	0.134	0.502
Geodesic	0.040	0.180	0.359	-	-	-
Selective Search	0.052	0.163	0.357	0.012	0.132	0.466
MCG	0.101	0.246	0.398	0.008	0.119	0.530
DeepMask	0.153	0.313	0.446	-	-	-
Ours Co-Obj (Bing)	0.183	0.340	0.423	**0.111**	0.438	0.590
Ours Co-Obj (Edge Boxes 70)	**0.189**	**0.366**	**0.492**	0.107	**0.449**	**0.686**

Table 3: **AR analysis on the MS COCO validation set:** We compare our method with state-of-the-art object proposal baselines according to the criteria of [17]. The results of our approach are provided in Rows 7-8 when using Bing and EdgeBox to generate the initial candidates, respectively. The AR for small, medium and large objects were computed for 100 proposals. Note that our co-generation approach outperforms the state-of-the-art baseline in all metrics.

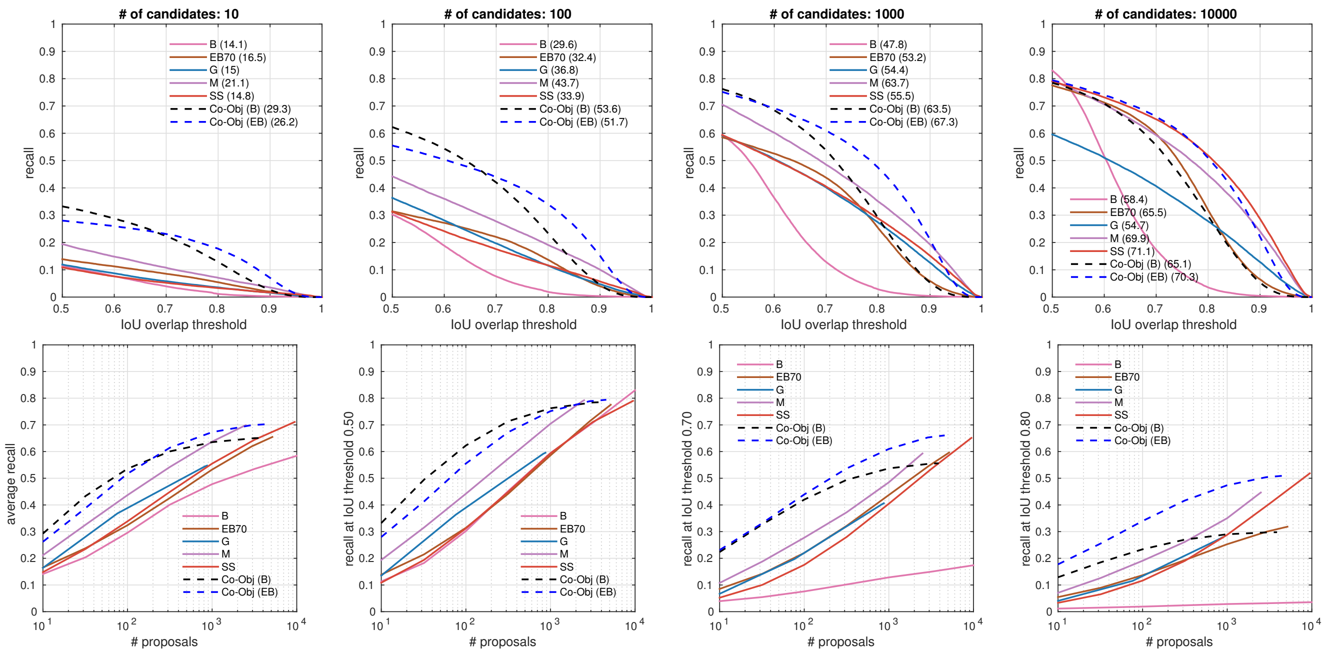

Figure 4: **MS COCO validation:** We compare our method with state-of-the-art object proposal baselines according to the criteria of [12]. **Top: Recall v.s. IoU threshold.** These recall curves were generated using the highest-scoring 10, 100, 1000 and 10000 object proposals, respectively. **Bottom: Recall v.s. Number of Proposals.** The first plot shows the AR, and the remaining recall curves were generated using IoU thresholds of 0.5, 0.7 and 0.8, respectively. In all the plots, the dashed lines correspond to our co-generation results, in blue when using EdgeBox candidates (**Co-Obj (EB)**) and in black when using Bing candidates (**Co-Obj (B)**). The baselines correspond to Bing (**B**), EdgeBoxes (**EB70**), Geodesic (**G**), MCG (**M**) and SelectiveSearch (**SS**). These results again evidence the benefits of our co-generation approach.

5.3. Results on Microsoft COCO

The results on Microsoft COCO, using the same metrics as before, are provided in Table 3 and Fig. 4, respectively. The same conclusions as before can be drawn from this analysis: Co-generating object proposals clearly is beneficial over generating the proposals independently. Our approach with EdgeBox initial candidates yields state-of-the-art results, with our Bing-based approach still outperforming the baselines for all metrics, with the exception of AR@1000. The improvement due to our approach is again consistent across all object sizes. The runtimes of our method on the MS COCO dataset are 1 and 0.8 sec per image when using Bing and EdgeBox, respectively.

6. Conclusion

We have introduced a framework to jointly generate object proposals from multiple images, thus leveraging the collective power of multiple object candidates. Our method is based on a deep structured network that jointly predicts the objectness scores and the bounding box locations of multiple object candidates by extracting features that model the individual object candidates and the similarity between them. Our experiments have demonstrated the benefits of our approach over the state-of-the-art methods that generate object proposals individually. In the future, we intend to exploit our high-quality object proposals to improve the accuracy of object (co-)detection.

References

[1] B. Alexe, T. Deselaers, and V. Ferrari. What is an object? In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2010. 1, 2

[2] S. Bao, Y. Xiang, and S. Savarese. Object co-detection. In *The European Conference on Computer Vision (ECCV)*, 2012. 1, 2

[3] L. Chen, A. G. Schwing, A. L. Yuille, and R. Urtasun. Learning deep structured models. In *International Conference on Machine Learning (ICML)*, 2014. 2

[4] M. Cheng, Z. Zhang, W. Lin, and P. H. S. Torr. BING: binarized normed gradients for objectness estimation at 300fps. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014. 1, 2, 6

[5] M. Everingham, L. V. Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The pascal visual object classes (voc) challenge. *The International Journal of Computer Vision (IJCV)*, 2010. 1, 6

[6] R. B. Girshick. Fast R-CNN. In *IEEE International Conference on Computer Vision (ICCV)*, 2015. 1, 2, 4, 5

[7] R. B. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014. 1, 2

[8] X. Guo, D. Liu, B. Jou, M. Zhu, A. Cai, and S. F. Chang. Robust Object Co-detection. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2013. 1, 2

[9] B. Hariharan, P. Arbeláez, R. Girshick, and J. Malik. Simultaneous detection and segmentation. In *The European Conference on Computer Vision (ECCV)*. 2014. 1, 2

[10] Z. Hayder, X. He, and M. Salzmann. Structural kernel learning for large scale multiclass object co-detection. In *IEEE International Conference on Computer Vision (ICCV)*, 2015. 1, 2

[11] Z. Hayder, M. Salzmann, and X. He. Object co-detection via efficient inference in a fully-connected crf. In *The European Conference on Computer Vision (ECCV)*. 2014. 1, 2

[12] J. H. Hosang, R. Benenson, P. Dollár, and B. Schiele. What makes for effective detection proposals? *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2015. 1, 6, 7, 8

[13] P. Krähenbühl and V. Koltun. Efficient inference in fully connected crfs with gaussian edge potentials. In *The Conference on Neural Information Processing Systems (NIPS)*, 2011. 1, 2, 3, 4, 5

[14] P. Krähenbühl and V. Koltun. Parameter learning and convergent inference for dense random fields. In *International Conference on Machine Learning (ICML)*, 2013. 3, 5

[15] P. Krähenbühl and V. Koltun. Geodesic object proposals. In *The European Conference on Computer Vision (ECCV)*, 2014. 1, 2

[16] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 1998. 2

[17] T. Lin, M. Maire, S. Belongie, L. D. Bourdev, R. B. Girshick, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft COCO: common objects in context. In *The European Conference on Computer Vision (ECCV)*, 2014. 1, 6, 7, 8

[18] P. O. Pinheiro, R. Collobert, and P. Dollár. Learning to segment object candidates. In *The Conference on Neural Information Processing Systems (NIPS)*, 2015. 1, 2, 6

[19] A. G. Schwing and R. Urtasun. Fully connected deep structured networks. *CoRR*, 2015. 2

[20] J. Shi, R. Liao, and J. Jia. CoDeL: An efficient human co-detection and labeling framework. In *IEEE International Conference on Computer Vision (ICCV)*, 2013. 1, 2

[21] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. In *International Conference on Learning Representations (ICLR)*, 2015. 5

[22] J. Uijlings, K. van de Sande, T. Gevers, and A. Smeulders. Selective search for object recognition. *The International Journal of Computer Vision (IJCV)*, 2013. 1, 2, 6

[23] S. Zheng, S. Jayasumana, B. Romera-Paredes, V. Vineet, Z. Su, D. Du, C. Huang, and P. Torr. Conditional random fields as recurrent neural networks. In *IEEE International Conference on Computer Vision (ICCV)*, 2015. 2

[24] S. Zheng, V. A. Prisacariu, M. Averkiou, M.-M. Cheng, N. J. Mitra, J. Shotton, P. H. Torr, and C. Rother. Object proposal estimation in depth images using compact 3d shape manifolds. *German Conference on Pattern Recognition (GCPR)*, 2015. 2

[25] C. L. Zitnick and P. Dollár. Edge boxes: Locating object proposals from edges. In *The European Conference on Computer Vision (ECCV)*, 2014. 1, 2, 6