SSAP: Single-Shot Instance Segmentation With Affinity Pyramid

Naiyu Gao[1,2], Yanhu Shan[3], Yupei Wang[1,2], Xin Zhao[1,2]*, Yinan Yu[3], Ming Yang[3], Kaiqi Huang[1,2,4]

[1] CRISE, Institute of Automation, Chinese Academy of Sciences
[2] University of Chinese Academy of Sciences [3] Horizon Robotics, Inc
[4] CAS Center for Excellence in Brain Science and Intelligence Technology

{gaonaiyu2017,wangyupei2014}@ia.ac.cn,{xzhao,kaiqi.huang}@nlpr.ia.ac.cn
{yanhu.shan,yinan.yu}@horizon.ai, m-yang4@u.northwestern.edu

Abstract

Recently, proposal-free instance segmentation has received increasing attention due to its concise and efficient pipeline. Generally, proposal-free methods generate instance-agnostic semantic segmentation labels and instance-aware features to group pixels into different object instances. However, previous methods mostly employ separate modules for these two sub-tasks and require multiple passes for inference. We argue that treating these two sub-tasks separately is suboptimal. In fact, employing multiple separate modules significantly reduces the potential for application. The mutual benefits between the two complementary sub-tasks are also unexplored. To this end, this work proposes a single-shot proposal-free instance segmentation method that requires only one single pass for prediction. Our method is based on a pixel-pair affinity pyramid, which computes the probability that two pixels belong to the same instance in a hierarchical manner. The affinity pyramid can also be jointly learned with the semantic class labeling and achieve mutual benefits. Moreover, incorporating with the learned affinity pyramid, a novel cascaded graph partition module is presented to sequentially generate instances from coarse to fine. Unlike previous time-consuming graph partition methods, this module achieves 5× speedup and 9% relative improvement on Average-Precision (AP). Our approach achieves new state of the art on the challenging Cityscapes dataset.

1. Introduction

The rapid development of Convolutional networks [30, 29] has revolutionized various vision tasks, enabling us to move towards more fine-grained understanding of images. Instead of classic bounding-box level object detection [19, 18, 46, 39, 44, 15] or class-level semantic segmen-
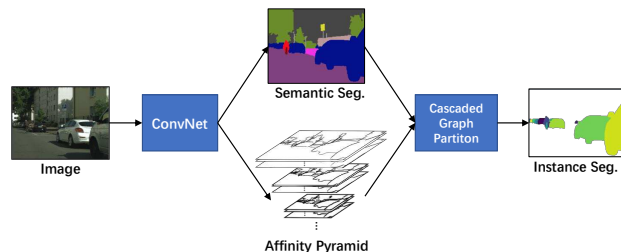

Figure 1. Overview of the proposed method. The per-pixel semantic class and pixel-pair affinities are generated with a single pass of a fully-convolutional network. The final instance segmentation result is then derived from these predictions by the proposed cascaded graph partition module.

tation [41, 6, 49], instance segmentation provides in-depth understanding by segmenting all objects and distinguishing different object instances. Researchers are thus showing increasing interests in instance segmentation recently.

Current state-of-the-art solutions to this challenging problem can be classified into the *proposal-based* and *proposal-free* approaches [34, 28, 40]. The proposal-based approaches regard it as an extension to the classic object detection task [46, 39, 44, 15]. After localizing each object with a bounding box, a foreground mask is predicted within each bounding box proposal. However, the performances of these proposal-based methods are highly limited by the quality of the bounding box predictions and the two-stage pipeline also limits the speed of the systems. By contrast, the proposal-free approach has the advantage of its simple and efficient design. This work also focuses on the proposal-free paradigm.

The proposal-free methods mostly start by producing instance-agnostic pixel-level semantic class labels [41, 6, 8, 49], followed by clustering them into different object instances with particularly designed instance-aware features. However, previous methods mainly treat the two sub-processes as two separate stages and employ multiple mod-

---
*Corresponding author

ules, which is suboptimal. In fact, the mutual benefits between the two sub-tasks can be exploited, which will further improve the performance of instance segmentation. Moreover, employing multiple modules may result in additional computational costs for real-world applications.

To cope with the above issues, this work proposes a single-shot proposal-free instance segmentation method, which jointly learns the pixel-level semantic class segmentation and object instance differentiating in a unified model with a single backbone network, as shown in Fig. 1. Specifically, for distinguishing different object instances, an affinity pyramid is proposed, which can be jointly learned with the labeling of semantic classes. The pixel-pair affinity computes the probability that two pixels belong to the same instance. In this work, the short-range affinities for pixels close to each other are derived with dense small learning windows. Simultaneously, the long-range affinities for pixels distant from each other are also required to group objects with large scales or nonadjacent parts. Instead of enlarging the windows, the multi-range affinities are decoupled and long-range affinities are sparsely derived from instance maps with lower resolutions. After that, we propose learning the affinity pyramid at multiple scales along the hierarchy of an U-shape network, where the short-range and long-range affinities are effectively learned from the feature levels with higher and lower resolutions respectively. Experiments in Table 3 show that the pixel-level semantic segmentation and pixel-pair affinity pyramid based grouping are indeed mutually benefited from the proposed joint learning scheme. The overall instance segmentation is thus further improved.

Then, in order to utilize the cues about global context reasoning, this work employs a graph partition method [26] to derive instances from the learned affinities. Unlike previous time-consuming methods, a cascaded graph partition module is presented to incorporate the graph partition process with the hierarchical manner of the affinity pyramid and finally provides both acceleration and performance improvements. Concretely, with the learned pixel-pair affinity pyramid, a graph is constructed by regarding each pixel as a node and transforming affinities into the edge scores. Graph partition is then employed from higher-level lower-resolution layers to lower-level higher-resolution layers progressively. Instance segmentation predictions from lower resolutions produce confident proposals, which significantly reduce node numbers at higher resolutions. Thus the whole process is accelerated.

The main contributions of this paper are as follows:

- A novel instance-aware pixel-pair affinity pyramid is proposed to distinguish instances, which can be jointly learned with the pixel-level labeling of semantic class. The mutual benefits between the two sub-tasks are explored by encouraging bidirectional inter-

actions, which further boosts instance segmentation.

- A single-shot, proposal-free instance segmentation method is proposed, based on the proposed affinity pyramid. Unlike most previous methods, our approach requires only one single pass to generate instances. On the challenging Cityscapes dataset, our method achieves new state of art with 37.3% AP (val) / 32.7% (test) and 61.1% PQ (val).

- Incorporating with the hierarchical manner of the affinity pyramid, a novel cascaded graph partition module is proposed to gradually segment an image into instances from coarse to fine. Compared with the noncascaded way, this module achieves 5× speedup and 9% relative improvement on AP.

2. Related Work

2.1. Instance Segmentation

Existing approaches on instance segmentation could be divided into two paradigms: proposal-based methods and proposal-free methods.

Proposal-based methods recognize object instances with bounding boxes that generated with detectors [46, 39, 15]. MNC [14] decomposes instance segmentation into a cascade of sub-tasks, including box localization, mask refinement and instance classification. Another work [2, 32] combines the predictions of detection and semantic segmentation with a CRFasRNN [50] to generate instances. FICS [33] develops the position sensitive score map [13]. Mask R-CNN [20] extends Faster R-CNN [46] by adding a segmentation mask predicting branch on each Region of Interest (RoI). Following works extend Mask R-CNN by modifying feature layers [38] or the mask prediction head [7].

Proposal-free methods mainly solve instance segmentation based on the success of semantic segmentation [6, 49, 8]. The segmentation based methods learn instance-aware features and use corresponding grouping methods to cluster pixels into instances. DWT [3] learns boundary-aware energy for each pixel followed by watershed transform. Several methods [5, 17, 43] adopt instance level embeddings to differentiate instances. SGN [37] sequentially groups instances with three sub-networks. Recurrent Neural Networks (RNNs) is adopted in several approaches [47, 45] to generate one instance mask at each time. Graph based algorithm [26] is also utilized for post-processing [31, 28], which segments an image into instances with global reasoning. However, the graph based algorithm is usually time-consuming. To speed up, Levinkov *et al.* [31] down-sample the outputs before the graph optimization while Kirillov *et al.* [28] only derive edges for adjacent neighbors. They all accelerate at the expense of performance. Recently, Yang *et al.* [48] propose a single-shot image parser that achieves a balance between accuracy and efficiency.

2.2. Pixel-Pair Affinity

The concept of learning pixel-pair affinity has been developed in many previous works [36, 23, 1, 4, 42] to facilitate semantic segmentations during training or post-processing. Recently, Liu *et al.* [40] propose learning instance-aware affinity and grouping pixels into instances with agglomerative hierarchical clustering. Our approach also utilizes instance-aware affinity to distinguish object instances, but both the ways to derive affinities and group pixels are significantly different. Importantly, Liu *et al.* [40] employ two models and require multiple passes for the RoIs generated from semantic segmentation results. Instead, our approach is single-shot, which requires only one single pass to generate the final instance segmentation result.

3. Proposed Approach

This work proposes a single-shot proposal-free instance segmentation model based on the jointly learned semantic segmentation and pixel-pair affinity pyramid, which are equipped with a cascaded graph partition module to differentiate object instances. As shown in Fig. 3, our model consists of two parts: (a) a unified network to learn the semantic segmentation and affinity pyramid with a single backbone network, and (b) a cascaded graph partition module to sequentially generate multi-scale instance predictions using the jointly learned affinity pyramid and semantic segmentation. In this section, the affinity pyramid is firstly explained at Subsection 3.1, then the cascaded graph partition module is described at Subsection 3.2.

3.1. Affinity Pyramid

With the instance-agnostic semantic segmentation, grouping pixels into individual object instance is critical for instance segmentation. This work proposes distinguishing different object instances based on the instance-aware pixel-pair affinity, which specifies whether two pixels belong to the same instance or not. As shown in the second column of Fig. 2, for each pixel, the short-range affinities to neighboring pixels within a small $r \times r$ window are learned. In this way, a $r^2 \times h \times w$ affinity response map is presented. For training, the average L2 loss is calculated with the r^2 predicted affinities for each pixel:

$$loss(a, y) = \frac{1}{r^2} \sum_{j=1}^{r^2} \left(y^j - a^j \right)^2, \quad (1)$$

where $a = [a^1, a^2, \ldots, a^{r^2}]$. a^j is the predicted affinity between the current pixel and the j-th pixel in its affinity window, representing the probability that two pixels belong to the same instance. The sigmoid activation is used to let $a^j \in (0, 1)$. Here, $y = [y^1, y^2, \ldots, y^{r^2}]$ and y^j represents the ground truth affinity for a^j. y^j is set to 1 if two pixels are
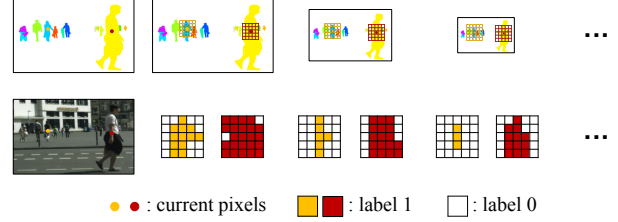

Figure 2. Illustration of affinity pyramid. Pixel-pair affinity specifies whether two pixels belong to the same instance or not. For each current pixel, the affinities to neighboring pixels within a small $r \times r$ window (here, $r = 5$) are predicted. The short-range and long-range affinities are decoupled and derived from instance maps with higher and lower resolutions respectively. In practice, ground truth affinity is set to 1 if two pixels are from the same instance, otherwise 0. Best viewed in color and zoom.

from the same instance, 0 if two pixels are from different instances. Importantly, the training data generated in this way is unbalanced. Specifically, the ground truth affinities are mostly with all 1 as most pixels are at the inner-regions of instances. To this end, 80% pixels with all 1 ground truth affinities are randomly dropped during training. Additionally, we set 3 times loss for pixels belonging to object instances.

Moreover, apart from the short-range affinities above, the long-range affinities are also required to handle objects of larger scales or nonadjacent object parts. A simple solution is to utilize a large affinity window size. However, besides the cost of GPU memories, a large affinity window would inevitably conflict with the semantic segmentation during training, which severely hinders the joint learning of the two sub-tasks. As shown in experiments (see Table 2), jointly learning the short-range affinities with semantic segmentation obtains mutual benefits for the two tasks. However, the long-range affinities are obviously more difficult to be jointly learned with the pixel-level semantic class labeling. Similar observation is also captured by Ke *et al.* [24].

Instead of enlarging the affinity window, we propose to learn multi-scale affinities as an affinity pyramid, where the short-range and long-range affinities are decoupled and the latter is sparsely derived from instance maps with lower resolutions. More concretely, as shown in Fig. 2, the long-range affinities are achieved with the same small affinity window at the lower resolutions. Note that the window sizes can be different, however they are fixed in this work for simplicity. In this way, the 5×5 windows from the $\frac{1}{64}$ resolution can produce affinities between pixels at most 128 pixel-distance. With the constructed affinity pyramid, the finer short-range and coarser long-range affinities are learned from the higher and lower resolutions, respectively. Consequently, multi-scale instance predictions are generated by affinities under corresponding resolutions. As

(a) U-Shape structure to predict Sem. Seg. and Affinity Pyramid

(b) Cascaded Graph Partition

⊕ Elem. Sum

→ Semantic branch

→ Affinity branch

S_i Semantic segmentation at i_{th} resolution

A_i Affinity prediction at i_{th} resolution

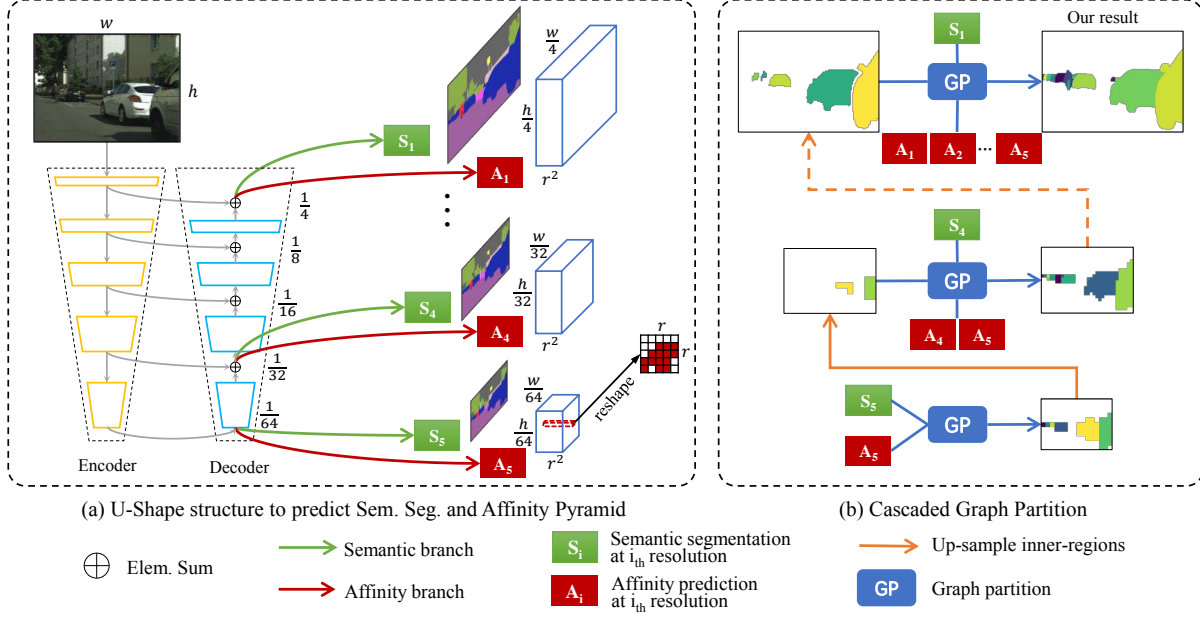→ Up-sample inner-regions

GP Graph partition

Figure 3. Our instance segmentation model consists of two parts: (a) a unified U-shape framework that jointly learns the semantic segmentation and affinity pyramid. The affinity pyramid is constructed by learning multi-range affinities from feature levels with different resolutions separately. (b) a cascaded graph partition module that utilizes the jointly learned affinity pyramid and semantic segmentation to progressively refine instance predictions starting from the deepest layer. Instance predictions in the lower-level layers with higher resolution are guided by the instance proposals generated from the deeper layers with lower resolution. Best viewed in color and zoom.

shown in Fig. 2, the predictions of larger instances are proposed by the lower resolution affinities, and are further detailed by higher resolution affinities. Meanwhile, although the smaller instances have too weak responses to be proposed at lower resolutions, they can be generated by the affinities with higher resolutions.

After that, the affinity pyramid can be easily learned by adding affinity branches in parallel with the existing branches for semantic segmentation along the hierarchy of the decoder network. As shown in Fig. 3 (a), affinities are predicted under $\{\frac{1}{4}, \frac{1}{8}, \frac{1}{16}, \frac{1}{32}, \frac{1}{64}\}$ resolutions of the original image. In this way, the short-range and long-range affinities can be effectively learned at different feature levels in the feature pyramid of the U-shape architecture. The formed affinity pyramid can thus be jointly learned with the semantic segmentation in a unified model, resulting in mutual benefits.

3.2. Cascaded Graph Partition

With the jointly learned semantic segmentation and affinity pyramid, a graph-based partition mechanism is employed in this work to differentiate object instances. In particular, incorporating with the hierarchical manner of the affinity pyramid, a cascaded graph partition module is presented. This module sequentially generates instances with multiple scales, guided by the cues encoded in the deeper-level layers of the affinity pyramid.

Graph Partition With the learned pixel-pair affinity pyramid, an undirected graph $G = (V, E)$ is constructed, where V is the set of pixels and $E \subseteq V^2$ is the set of pixel-pairs within affinity windows. $e_{u,v} \in E$ represents the edge between the pixels $\{u, v\}$. Furthermore, $a_{u,v}, a_{v,u} \in (0, 1)$ are the affinities for pixels $\{u, v\}$, which are predicted at pixels u and v, respectively. The average affinity $\alpha_{u,v}$ is then calculated and transformed into the score $w_{u,v}$ of edge $e_{u,v}$ by:

$$\alpha_{u,v} = (a_{u,v} + a_{v,u})/2, \qquad (2)$$

$$w_{u,v} = \log(\frac{\alpha_{u,v}}{1 - \alpha_{u,v}}). \qquad (3)$$

As the affinities predict how likely two pixels belong to the same instance, the average affinities higher than 0.5 are transformed into positive and negative otherwise. In this way, instance segmentation is transformed into a graph partition problem [11] and can be addressed by solving the following optimization problem [26]:

$$\min_{y \in \{0,1\}} \sum_{e \in E} w_e y_e, \qquad (4)$$

$$s.t. \forall C \, \forall e' \in C : \sum_{e \in C \setminus \{e'\}} y_e \geq y_{e'}. \qquad (5)$$

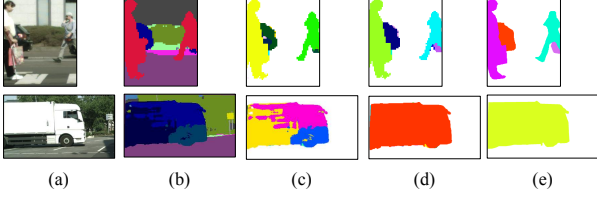Here, $y_e = y_{u,v} \in \{0, 1\}$ is a binary variable and $y_{u,v} = 1$

Figure 4. Influence of segmentation refinement (SR). (a) Input image. (b) Semantic segmentation. (c) Instance segmentation without SR (d) Instance segmentation with SR. (e) Ground truth. SR significantly improves the errors in instance segmentation which are caused by the semantic segmentation failures. Best viewed in color and zoom.

represents nodes u and v belong to different partitions. C is the set of all cycles of the graph G. The objective in formulation 4 is about to maximize the total score of the selected edges, and the inequality 5 constrains each feasible solution representing a partition. A search-based algorithm [26] is developed to solve the optimization problem. However, when this algorithm is employed to segment instances, the inference time is not only long but also rises significantly *w.r.t.* the number of nodes, which brings potential problem for real-world applications.

Cascade Scheme The sizes of instances in the Cityscapes dataset are various significantly. For large instances, the pixels are mostly at the inner-regions which cost long inference time although are easy for segmentation. Motivated by this observation, a cascaded strategy is developed to incorporate the graph partition mechanism with the hierarchical manner of the affinity pyramid. As shown in Fig. 3 (b), the graph partition is firstly utilized on a low resolution where it has fewer pixels and requires a short running time for graph partition. Although only coarse segments for large instances are generated, the inner-regions for these segments are still reliable. In this case, these inner-regions can be up-sampled and regarded as proposals for the higher resolution. At the higher resolution, the pixels in each proposal are combined to generate a node and the remaining pixels are each treated as a node. To construct a graph with these nodes, the edge score w_{t_i,t_j} between nodes t_i and t_j is calculated by adding all pixel-pair edge scores between the two nodes: $w_{t_i,t_j} = \sum_{u \in t_i, v \in t_j} w_{u,v}$. In this way, the proposals for instance predictions are progressively refined. Because the number of nodes decreases significantly at each step, the entire graph partition is accelerated.

Segmentation Refinement In previous steps, the partition is made within each class to speed up. At this step, the cues from both semantic segmentation and affinity branches are integrated to segmentation instances from all the pixels which are classified as foreground. In practice, the average

affinity $\alpha_{u,v}$ for pixels $\{u, v\}$ is refined to $\alpha'_{u,v}$ by:

$$\alpha'_{u,v} = \alpha_{u,v} * \exp[-\mathrm{D_{JS}}(s_u \| s_v)], \qquad (6)$$

$$\mathrm{D_{JS}}(P\|Q) = \frac{1}{2}\Big[\mathrm{D_{KL}}\big(P\|\frac{P+Q}{2}\big) + \mathrm{D_{KL}}\big(Q\|\frac{P+Q}{2}\big)\Big], \qquad (7)$$

$$\mathrm{D_{KL}}(P\|Q) = \sum_i P_i \log \frac{P_i}{Q_i}. \qquad (8)$$

Here, $s_u = [s_u^1, s_u^2, \ldots, s_u^c]$ and $s_v = [s_v^1, s_v^2, \ldots, s_v^c]$ are the semantic segmentation scores for c object classes at the pixel u and v, which represent the classification possibility distributions on the c object classes. The distance between the two distributions can be measured with the popular Jensen-Shannon divergence, as described in Eq. 7-8. After the refinement for the initial affinities, graph partition is conducted for all the foreground pixels at the $\frac{1}{4}$ resolution. By combining the information from semantic segmentation and affinity branches, the errors in instance segmentation which are caused by the semantic segmentation failures are significantly improved, as shown in Fig. 4.

Finally, the class label for each instance is obtained by voting among all pixels based on semantic segmentation labels. Following DWT [3], small instances are removed and semantic scores from semantic segmentation are used to rank predictions.

4. Experiments

Dataset Our model is evaluated on the challenging urban street scenes dataset Cityscapes [12]. In this dataset, each image has a high resolution of 1,024×2,048 pixels. There are 5,000 images with high quality dense pixel annotations and 20,000 images with coarse annotations. Note that only the fine annotated dataset is used to train our model. Cityscapes benchmark evaluates 8 classes for instance segmentation. Together with another 11 background classes, 19 classes are evaluated for semantic segmentation.

Metrics The main metric for evaluation is Average-Precision (AP), which is calculated by averaging the precisions under IoU (Intersection over Union) thresholds from 0.50 to 0.95 at the step of 0.05. Our result is also reported with three sub-metrics from Cityscapes: AP50%, AP100m and AP50m. They are calculated at 0.5 IoU threshold or only for objects within specific distances.

This paper also evaluates the results with a new metric Panoptic Quality (PQ) [27], which is further divided into Segmentation Quality (SQ) and Recognition Quality (RQ) to measure recognition and segmentation performances respectively. The formulation PQ is defined as :

$$\mathrm{PQ} = \underbrace{\frac{\sum_{p,g \in TP} \mathrm{IoU}(p,g)}{|TP|}}_{\text{Segmentation Quality (SQ)}} \times \underbrace{\frac{|TP|}{|TP| + \frac{1}{2}|FP| + \frac{1}{2}|FN|}}_{\text{Recognition Quality (RQ)}}, \quad (9)$$

α	λ	AP (%)	PQ$^{\text{Th}}$ (%)	PQ (%)
0.0003	diff.	27.5	45.0	54.6
0.001	diff.	29.5	48.0	55.8
0.003	diff.	**31.5**	**49.2**	**56.6**
0.003	same	31.0	48.7	56.2
0.01	diff.	31.0	**49.2**	56.3
0.03	diff.	28.1	46.4	53.4

Table 1. Influence of the balancing parameters.

r	AP (%)	PQ$^{\text{Th}}$ (%)	PQ (%)	mIoU (%)
0	-	-	-	74.5
3	30.5	48.5	56.4	75.0
5	**31.3**	**49.0**	**56.5**	75.0
7	31.2	48.1	56.0	**75.1**
9	30.0	46.2	55.0	74.3

Table 2. Influence of affinity window size r. mIoU for semantic segmentation evaluation is also provided. $r = 0$ means to train semantic segmentation only.

Feature	JL	AP (%)	PQ$^{\text{Th}}$ (%)	PQ (%)	mIoU (%)
Single		29.4	46.9	54.9	74.5
(w/o dilation)	$\surd$	30.2	47.6	55.0	74.2
Single		30.6	48.2	55.5	74.5
(w/ dilation)	$\surd$	30.8	48.8	55.8	74.5
Hierarchical		30.0	47.7	55.2	74.5
	$\surd$	**31.3**	**49.0**	**56.5**	**75.0**

Table 3. **JL**: joint learning. Comparing with learning all layers of the affinity pyramid from the single 1/4 resolution feature map, our hierarchical manner with joint learning performs better.

where p and g are the predicted and ground truth segments, while TP, FP and FN represent matched pairs of segments, unmatched predicted segments and unmatched ground truth segments respectively. Moreover, both countable objects (thing) and uncountable regions (stuff) are evaluated in PQ and are separately reported with PQ$^{\text{Th}}$ and PQ$^{\text{St}}$. As the stuff is not concerned in this work, only PQ and PQ$^{\text{Th}}$ is reported.

Implementation Details Our model predicts semantic segmentation and pixel-pair affinity with a unified U-shape framework based on ResNet-50 [21]. The training loss L is defined as:

$$L = \sum_i (L_s^i + \alpha\lambda_i L_a^i), \quad (10)$$

where L_s^i and L_a^i are multi-class focal loss [35] and average L2 loss (see Eq. 1) for semantic segmentation and affinity branches at the i_{th} resolution in $[\frac{1}{4}, \frac{1}{8}, ..., \frac{1}{64}]$ resolutions respectively. To combine losses from each scale, we firstly tune the balancing parameter λ_i to make losses of each scale are in the same order, which are finally set to $[0.01, 0.03, 0.1, 0.3, 1]$ respectively. After that, α is set to 0.003 to balance the losses of affinity pyramid and semantic segmentation. The influence of α and λ_i are shown in Table 1. We run all experiments using the MXNet frame-
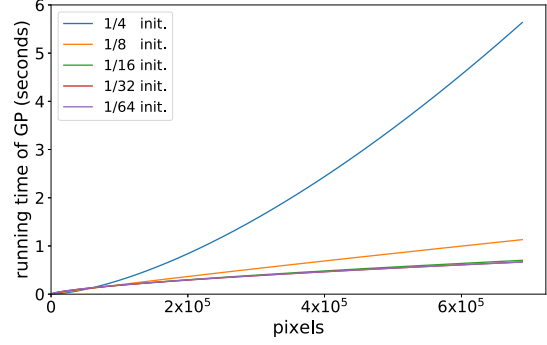

Figure 5. Running time for the cascaded graph partition module under different object sizes. The cascade scheme significantly reduces the time for large objects. Best viewed in color.

work [9]. Our model is trained with Nadam [16] for 70,000 iterations using synchronized batch normalization [22] over 8 TitanX 1080ti GPUs and the batch size is set to 24. The learning rate is initialized to 10^{-4} and divided by 10 at the 30,000 and 50,000 iterations, respectively.

Influence of Joint Learning Our separately trained semantic segmentation model achieves 74.5% mIoU. This result is significantly improved after being jointly trained with the affinity pyramid, as shown in Table 2. However, the performance for both instance and semantic segmentation is affected by the affinity window size. Similar phenomenon is also observed by Ke *et al.* [24] and they explain that small windows and large windows benefit small objects and large objects, respectively. Due to the limitation of GPU memory, the window size is tested from 3 to 9. Among them, 5×5 affinity window balances the conflict and achieves the best performance, which is used in the other experiments. Furthermore, in our proposed model, the semantic segmentation and affinity pyramid are jointly learned along the hierarchy of the U-shape network. We compare this approach with generating all layers of the affinity pyramid from the single $\frac{1}{4}$ resolution feature map with corresponding strides. The employing of dilated convolution [6] is also tested. Table 3 shows our approach performs best, where the mutually benefits of the two tasks are explored and finally improve the performance on instance segmentation.

Influence of Cascaded Graph Partition At this part, the proposed cascaded graph partition module is analyzed by being initialized from each resolution. As shown in Fig. 5, the running time for graph partition increases rapidly w.r.t. the size of object regions when conducting the partition at the $\frac{1}{4}$ resolution directly, without the guidance of instance proposals. However, the time significantly reduces when initializing the cascaded graph partition from lower resolutions, like the $\frac{1}{16}$ resolution, where the graph partition is constructed at the $[\frac{1}{16}, \frac{1}{8}, \frac{1}{4}]$ resolutions sequentially, and the latter two are guided by the proposals from the previous

Init. Res.	GP time (s)	AP (%)	PQ$^{\text{Th}}$ (%)	PQ (%)
1/4	1.26	28.9	45.1	54.9
1/8	0.33	31.3	**49.2**	**56.6**
1/16	0.26	**31.5**	**49.2**	**56.6**
1/32	0.26	30.9	48.8	56.5
1/64	0.26	30.9	48.7	56.5

Table 4. Influence of the initial resolution for the cascaded graph partition. With the decreasing of initial resolution, the GP time (running time for cascaded graph partition per image) keeps decreasing. Comparing with $\frac{1}{4}$ resolution initialization, initializing cascaded graph partition from the $\frac{1}{16}$ resolution achieves $5\times$ speedup with 9% AP improvement.

Affinities Used	AP (%)	PQ$^{\text{Th}}$ (%)	PQ (%)
A_1 only	25.7	41.2	53.2
$+A_2$	29.8	46.5	55.4
$+A_3$	30.8	48.6	56.3
$+A_4$	31.4	49.2	56.5
$+A_5$	31.5	49.2	56.6

Table 5. Effectiveness of the long-range affinities. $[A_1, A_2, ..., A_5]$ are affinities of the $[\frac{1}{4}, \frac{1}{8}, ..., \frac{1}{64}]$ resolutions respectively. Affinities with longer-range are gradually added.

BD	OL	Kernel	AP (%)	PQ$^{\text{Th}}$ (%)	PQ (%)
		3	29.1	46.4	55.8
$\checkmark$		3	30.0	48.8	56.0
$\checkmark$	$\checkmark$	3	31.3	49.0	56.5
$\checkmark$	$\checkmark$	5	31.5	49.2	56.6

Table 6. **BD**: balance the training data by randomly dropping 80% pixels with all 1 ground truth affinities. **OL**: set 3 times affinity loss for pixels belonging to object instances. **Kernel**: kernel size.

Backbone	SR	HF	MS	AP (%)	PQ$^{\text{Th}}$ (%)	PQ (%)
ResNet-50				28.7	45.4	55.1
ResNet-50	$\checkmark$			31.5	49.2	56.6
ResNet-50	$\checkmark$	$\checkmark$		32.8	50.4	57.6
ResNet-50	$\checkmark$	$\checkmark$	$\checkmark$	34.4	50.6	58.4
ResNet-101	$\checkmark$	$\checkmark$	$\checkmark$	37.3	55.0	61.1

Table 7. **SR**: segmentation refinement. **HF**: horizontal flipping test. **MS**: multiscale test.

Method	AP (%)	PQ$^{\text{Th}}$ (%)	PQ (%)	Backbone
Li *et al.* [32]	28.6	42.5	53.8	ResNet-101
SGN [37]	29.2	-	-	-
Mask R-CNN [20]	31.5	49.6[1]	-	ResNet-50
GMIS [40]	34.1	-	-	ResNet-101
Deeperlab [48]	-	-	56.5	Xception-71 [10]
PANet [38]	36.5	-	-	ResNet-50
SSAP (ours)	34.4	50.6	58.4	ResNet-50
SSAP (ours)	**37.3**	**55.0**	**61.1**	ResNet-101

Table 8. Results on Cityscapes **val** set. All results are trained with Cityscapes data only.

Method	PQ [val]	PQ	SQ	RQ	PQ$^{\text{Th}}$	SQ$^{\text{Th}}$	RQ$^{\text{Th}}$	PQ$^{\text{St}}$	SQ$^{\text{St}}$	RQ$^{\text{St}}$
DeeperLab [48]	33.8	34.3	77.1	43.1	37.5	77.5	46.8	29.6	76.4	37.4
SSAP (ours)	36.5	36.9	80.7	44.8	40.1	81.6	48.5	32.0	79.4	39.3

Table 9. Results on COCO **val** ('PQ [val]' column) and **test-dev** (remaining columns) sets. Results are reported as percentages.

stage. The quantitative results are shown in Table 4. Comparing with the $\frac{1}{4}$ resolution initialization (non-cascaded), the $\frac{1}{64}$ resolution initializing scheme achieves $5\times$ acceleration. Importantly, the cascaded approach achieves speeding up without scarifying precisions. As shown in Table 4, initializing from the $\frac{1}{64}$ resolution has 2.0% absolute improvement on AP, which is achieved due to that the proposals from lower resolutions can reduce the disturbing information for prediction. Meanwhile, the $\frac{1}{16}$ resolution initializing approach achieves better performance than the $\frac{1}{64}$ and $\frac{1}{32}$ manner, which indicates proposals from too low resolutions still bring errors for prediction. In the other experiments, cascaded graph partitions are initialized from the $\frac{1}{16}$ resolution.

Quantitative Results Firstly, to show the effectiveness of the long-range affinities, we start with just using the affinities from the 1/4 resolution, and gradually add longer-range affinities. Results are shown in Table 5. Then, the influences of balancing training data, setting larger affinity loss and employing a large kernel are evaluated and shown in Table 6. After that, as shown in Table 7, the segmentation refinement improves the performance with 2.8% AP. With test tricks, our model achieves 34.4% AP and 58.4% PQ on

the validation set. Our model is also trained with ResNet-101, which achieves 37.3% AP and 61.1% PQ, as shown in Table 8. For the test set, our model attains a performance of 32.7% AP, which exceeds all previous methods. Details are in Table 10.

Visual Results The proposals generated from the $\frac{1}{16}$ and $\frac{1}{8}$ resolutions are visualized in Fig 6. A few sample results on the validation set are visualized in Fig 7, where fine details are precisely captured. As shown in the second column, the cars occluded by persons or poles and separated into parts are successfully grouped.

Results on COCO To show the effectiveness of our method in scenarios other than streets, we evaluate it on the COCO dataset. The annotations for COCO instance segmentation are with overlaps, making it unsuitable to train and test a proposal-free method like ours. So our method is evaluated in the panoptic segmentation task. To train on COCO, we resize the longer edge to 640 and train the model with 512×512 crops. The number of iterations is 80,000 and the learning rate is divided by 10 in 60,000 and 70,000 iterations. Other experimental settings are remained the same. The performance of our model (ResNet-101 based) is summarized in Table 9. To the best of our knowledge, DeeperLab [48] is currently the only proposal-free method to report COCO result. Our method outperformes DeeperLab (Xception-71 based) in all sub metrics.

---

[1]This result is reported by Kirillov *et al.* [27].

Method	Training data	AP	AP50%	AP50m	AP100m	person	rider	car	trunk	bus	train	motor	bicycle
InstanceCut [28]	fine+coarse	13.0	27.9	26.1	22.1	10.0	8.0	23.7	14.0	19.5	15.2	9.3	4.7
Multi-task [25]	fine	21.6	39.0	37.0	35.0	19.2	21.4	36.6	18.8	26.8	15.9	19.4	14.5
SGN [37]	fine+coarse	25.0	44.9	44.5	38.9	21.8	20.1	39.4	24.8	33.2	30.8	17.7	12.4
Mask RCNN [20]	fine	26.2	49.9	40.1	37.6	30.5	23.7	46.9	22.8	32.2	18.6	19.1	16.0
GMIS [40]	fine+coarse	27.3	45.6	-	-	31.5	25.2	42.3	21.8	37.2	28.9	18.8	12.8
Neven *et al.* [43]	fine	27.6	50.9	-	-	34.5	26.1	52.4	21.7	31.2	16.4	20.1	18.9
PANet [38]	fine	31.8	**57.1**	46.0	44.2	**36.8**	**30.4**	54.8	27.0	36.3	25.5	**22.6**	**20.8**
SSAP (ours)	fine	**32.7**	51.8	**51.4**	**47.3**	35.4	25.5	**55.9**	**33.2**	**43.9**	**31.9**	19.5	16.2

Table 10. Results on Cityscapes **test** set. All results are trained with Cityscapes data only. Results are reported as percentages.

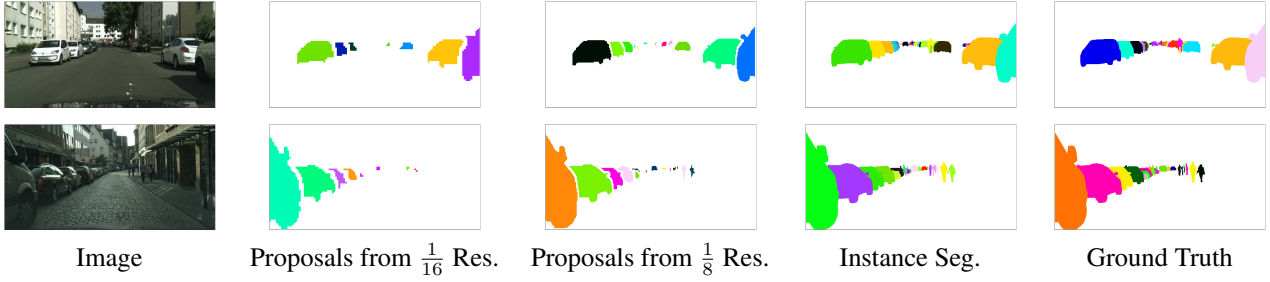| Image | Proposals from $\frac{1}{16}$ Res. | Proposals from $\frac{1}{8}$ Res. | Instance Seg. | Ground Truth |

Figure 6. Visualizations of proposals generated from lower resolutions within the cascaded graph partition module and the final instance segmentation results. Best viewed in color and zoom.
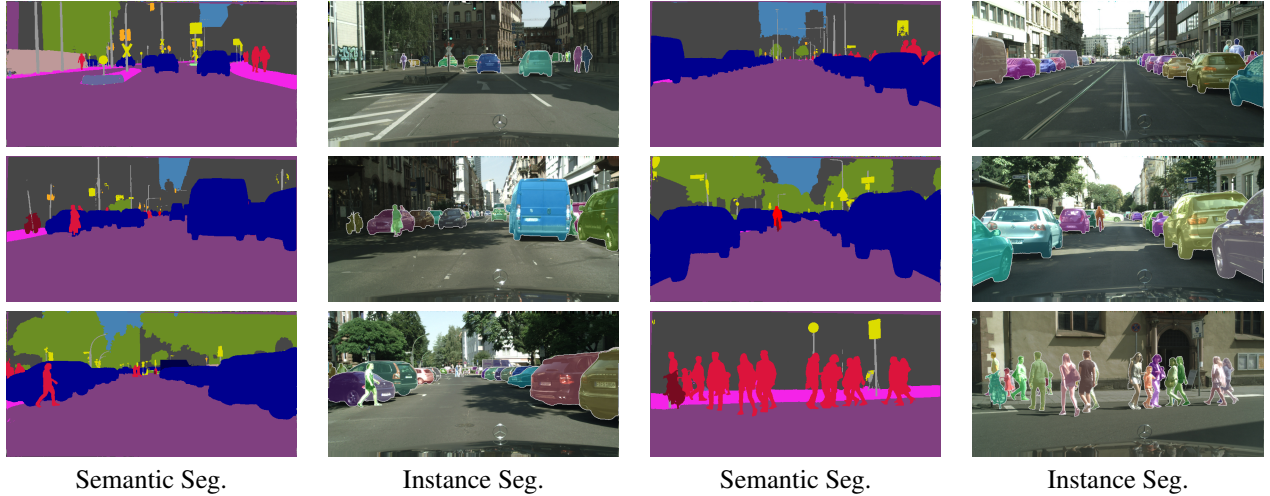

| Semantic Seg. | Instance Seg. | Semantic Seg. | Instance Seg. |

Figure 7. Visualizations of sampled results on the validation set. Best viewed in color and zoom.

5. Conclusion

This work has proposed a single-shot proposal-free instance segmentation method, which requires only one single pass to generate instances. Our method is based on a novel affinity pyramid to distinguish instances, which can be jointly learned with the pixel-level semantic class labels using a single backbone network. Experiment results have shown the two sub-tasks are mutually benefited from our joint learning scheme, which further boosts instance segmentation. Moreover, a cascaded graph partition module has been developed to segment instances with the affinity pyramid and semantic segmentation results. Compar-

ing with the non-cascaded way, this module has achieved $5\times$ speedup and 9% relative improvement on AP. Our approach has achieved a new state of the art on the challenging Cityscapes dataset.

References

[1] Jiwoon Ahn and Suha Kwak. Learning pixel-level semantic affinity with image-level supervision for weakly supervised semantic segmentation. In *CVPR*, 2018. 3

[2] Anurag Arnab and Philip H. S. Torr. Pixelwise instance segmentation with a dynamically instantiated network. In *CVPR*, 2017. 2

[3] Min Bai and Raquel Urtasun. Deep watershed transform for instance segmentation. In *CVPR*, 2017. 2, 5

[4] Gedas Bertasius, Lorenzo Torresani, Stella X Yu, and Jianbo Shi. Convolutional random walk networks for semantic image segmentation. In *CVPR*, 2017. 3

[5] Bert De Brabandere, Davy Neven, and Luc Van Gool. Semantic instance segmentation with a discriminative loss function. *arXiv:1708.02551*, 2017. 2

[6] Liangchieh Chen, George Papandreou, Iasonas Kokkinos, Kevin P Murphy, and Alan L Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *TPAMI*, 40(4), 2018. 1, 2, 6

[7] Liang-Chieh Chen, Alexander Hermans, George Papandreou, Florian Schroff, Peng Wang, and Hartwig Adam. Masklab: Instance segmentation by refining object detection with semantic and direction features. In *CVPR*, 2018. 2

[8] Liang-Chieh Chen, George Papandreou, Florian Schroff, and Hartwig Adam. Rethinking atrous convolution for semantic image segmentation. *arXiv:1706.05587*, 2017. 1, 2

[9] Tianqi Chen, Mu Li, Yutian Li, Min Lin, Naiyan Wang, Minjie Wang, Tianjun Xiao, Bing Xu, Chiyuan Zhang, and Zheng Zhang. Mxnet: A flexible and efficient machine learning library for heterogeneous distributed systems. *arXiv:1512.01274*, 2015. 6

[10] Francois Chollet. Xception: Deep learning with depthwise separable convolutions. In *CVPR*, 2017. 7

[11] Sunil Chopra and M R Rao. The partition problem. *Mathematical Programming*, 59(1), 1993. 4

[12] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *CVPR*, 2016. 5

[13] Jifeng Dai, Kaiming He, Yi Li, Shaoqing Ren, and Jian Sun. Instance-sensitive fully convolutional networks. In *ECCV*, 2016. 2

[14] Jifeng Dai, Kaiming He, and Jian Sun. Instance-aware semantic segmentation via multi-task network cascadeds. In *CVPR*, 2016. 2

[15] Jifeng Dai, Yi Li, Kaiming He, and Jian Sun. R-fcn: Object detection via region-based fully convolutional networks. In *NIPS*, 2016. 1, 2

[16] Timothy Dozat. Incorporating nesterov momentum into adam. 2016. 6

[17] Alireza Fathi, Zbigniew Wojna, Vivek Rathod, Peng Wang, Hyun Oh Song, Sergio Guadarrama, and Kevin P. Murphy. Semantic instance segmentation via deep metric learning. *arXiv:1703.10277*, 2017. 2

[18] Ross Girshick. Fast r-cnn. In *ICCV*, 2015. 1

[19] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *CVPR*, 2014. 1

[20] Kaiming He, Georgia Gkioxari, Piotr Dollar, and Ross Girshick. Mask r-cnn. In *ICCV*, 2017. 2, 7, 8

[21] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 6

[22] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *ICML*, 2015. 6

[23] Tsungwei Ke, Jyhjing Hwang, Ziwei Liu, and Stella X Yu. Adaptive affinity fields for semantic segmentation. In *ECCV*, 2018. 3

[24] Tsung-Wei Ke, Jyh-Jing Hwang, Ziwei Liu, and Stella X. Yu. Adaptive affinity fields for semantic segmentation. In *ECCV*, 2018. 3, 6

[25] Alex Kendall, Yarin Gal, and Roberto Cipolla. Multi-task learning using uncertainty to weigh losses for scene geometry and semantics. In *CVPR*, 2018. 8

[26] Margret Keuper, Evgeny Levinkov, Nicolas Bonneel, Guillaume Lavoue, Thomas Brox, and Bjoern Andres. Efficient decomposition of image and mesh graphs by lifted multicuts. In *ICCV*, 2015. 2, 4, 5

[27] Alexander Kirillov, Kaiming He, Ross Girshick, Carsten Rother, and Piotr Dollar. Panoptic segmentation. In *CVPR*, 2019. 5, 7

[28] Alexander Kirillov, Evgeny Levinkov, Bjoern Andres, Bogdan Savchynskyy, and Carsten Rother. Instancecut: From edges to instances with multicut. In *CVPR*, 2017. 1, 2, 8

[29] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, 2012. 1

[30] Yann Lecun, Leon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11), 1998. 1

[31] Evgeny Levinkov, Jonas Uhrig, Siyu Tang, Mohamed Omran, Eldar Insafutdinov, Alexander Kirillov, Carsten Rother, Thomas Brox, Bernt Schiele, and Bjoern Andres. Joint graph decomposition & node labeling: Problem, algorithms, applications. In *CVPR*, 2017. 2

[32] Qizhu Li, Anurag Arnab, and Philip H.S. Torr. Weakly- and semi-supervised panoptic segmentation. In *ECCV*, 2018. 2, 7

[33] Yi Li, Haozhi Qi, Jifeng Dai, Xiangyang Ji, and Yichen Wei. Fully convolutional instance-aware semantic segmentation. In *CVPR*, 2017. 2

[34] Xiaodan Liang, Yunchao Wei, Xiaohui Shen, Jianchao Yang, Liang Lin, and Shuicheng Yan. Proposal-free network for instance-level object segmentation. *arXiv:1509.02636*, 2015. 1

[35] Tsungyi Lin, Priya Goyal, Ross B Girshick, Kaiming He, and Piotr Dollar. Focal loss for dense object detection. In *ICCV*, 2017. 6

[36] Sifei Liu, Shalini De Mello, Jinwei Gu, Guangyu Zhong, Ming-Hsuan Yang, and Jan Kautz. Learning affinity via spatial propagation networks. In *NIPS*, 2017. 3

[37] Shu Liu, Jiaya Jia, Sanja Fidler, and Raquel Urtasun. Sgn: Sequential grouping networks for instance segmentation. In *ICCV*, 2017. 2, 7, 8

[38] Shu Liu, Lu Qi, Haifang Qin, Jianping Shi, and Jiaya Jia. Path aggregation network for instance segmentation. In *CVPR*, 2018. 2, 7, 8

[39] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott E Reed, Chengyang Fu, and Alexander C Berg. SSD: Single shot multibox detector. In *ECCV*, 2016. 1, 2

[40] Yiding Liu, Siyu Yang, Bin Li, Wengang Zhou, Jizheng Xu, Houqiang Li, and Yan Lu. Affinity derivation and graph merge for instance segmentation. In *ECCV*, 2018. 1, 3, 7, 8

[41] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *CVPR*, 2015. 1

[42] Michael Maire, Takuya Narihira, and Stella X Yu. Affinity cnn: Learning pixel-centric pairwise relations for figure/ground embedding. In *CVPR*, 2016. 3

[43] Davy Neven, Bert De Brabandere, Marc Proesmans, and Luc Van Gool. Instance segmentation by jointly optimizing spatial embeddings and clustering bandwidth. In *CVPR*, 2019. 2, 8

[44] Joseph Redmon and Ali Farhadi. Yolo9000: Better, faster, stronger. In *CVPR*, 2017. 1

[45] Mengye Ren and Richard S Zemel. End-to-end instance segmentation with recurrent attention. In *CVPR*, 2017. 2

[46] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *NIPS*, 2015. 1, 2

[47] Bernardino Romeraparedes and Philip H S Torr. Recurrent instance segmentation. In *ECCV*, 2016. 2

[48] Tien-Ju Yang, Maxwell D. Collins, Yukun Zhu, Jyh-Jing Hwang, Ting Liu, Xiao Zhang, Vivienne Sze, George Papandreou, and Liang-Chieh Chen. Deeperlab: Single-shot image parser. *arXiv:1902.05093*, 2019. 2, 7

[49] Hengshuang Zhao, Jianping Shi, Xiaojuan Qi, Xiaogang Wang, and Jiaya Jia. Pyramid scene parsing network. In *CVPR*, 2017. 1, 2

[50] Shuai Zheng, Sadeep Jayasumana, Bernardino Romeraparedes, Vibhav Vineet, Zhizhong Su, Dalong Du, Chang Huang, and Philip H S Torr. Conditional random fields as recurrent neural networks. In *ICCV*, 2015. 2