This CVPR 2021 paper is the Open Access version, provided by the Computer Vision Foundation.
Except for this watermark, it is identical to the accepted version;
the final published version of the proceedings is available on IEEE Xplore.

BoxInst: High-Performance Instance Segmentation with Box Annotations

Zhi Tian, Chunhua Shen*, Xinlong Wang, Hao Chen
The University of Adelaide, Australia

Figure 1: Some qualitative results of BoxInst with the ResNet-101 based model achieving 33.0% mask AP on COCO `val2017`. The model is trained without any mask annotations and can infer at 17 FPS on a V100 GPU. Best viewed on screen.

Abstract

We present a high-performance method that can achieve mask-level instance segmentation with only bounding-box annotations for training. While this setting has been studied in the literature, here we show significantly stronger performance with a simple design (e.g., dramatically improving previous best reported mask AP of **21.1%** *[13]* **to 31.6%** *on the COCO dataset). Our core idea is to redesign the loss of learning masks in instance segmentation, with no modification to the segmentation network itself. The new loss functions can supervise the mask training without relying on mask annotations. This is made possible with two loss terms, namely, 1) a surrogate term that minimizes the discrepancy between the projections of the ground-truth box and the predicted mask; 2) a pairwise loss that can exploit the prior that proximal pixels with similar colors are very likely to have the same category label.*

Experiments demonstrate that the redesigned mask loss can yield surprisingly high-quality instance masks with only box annotations. For example, without using any mask annotations, with a ResNet-101 backbone and 3× training schedule, we achieve 33.2% mask AP on COCO test-dev split (vs. 39.1% of the fully supervised counterpart). Our excellent experiment results on COCO and Pascal VOC indicate that our method dramatically narrows the performance gap between weakly and fully supervised instance segmentation.

Code is available at: https://git.io/AdelaiDet

1. Introduction

Instance segmentation requires the algorithm to predict the pixel-wise masks and categories of instances of interest, and is one of the most fundamental tasks in computer vision. The performance of instance segmentation has been significantly advanced by a number of recent successful methods [5, 6, 11, 15, 28, 30, 31]. These methods have almost made the previously much more challenging instance segmentation task be as simple and fast as bounding-box object detection. For example, built on the detector FCOS [29], CondInst [28] only adds very compact dynamic mask heads to predict instance masks, and thus only introduces less than 10% computation time, compared to FCOS. Instance segmentation is able to provide more accurate and finer *mask-*

*Corresponding author.

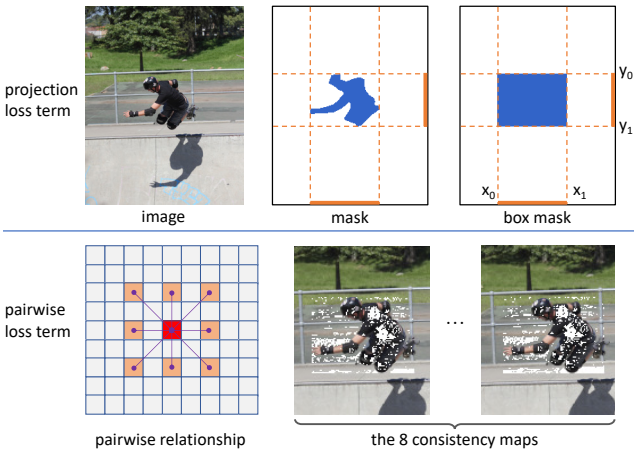

Figure 2: The two proposed loss terms. Top row: the projections onto x-axis and y-axis of the mask and the box, and the projections should be the same, where (x_0, y_0) and (x_1, y_1) are the two corners of the box. Bottom row: the pairwise term. For each pixel, we compute the pairwise label consistency between the pixel and its 8 neighbours (with dilation rate 2). Thus each pixel has 8 edges and we have 8 consistency maps in the right. The white locations in the right figure are the edges we have the supervision derived from the color similarity, and other edges are discarded in the loss computation.

level object location than detection. Thus, given that the extra computation cost is negligible, instance segmentation should be preferred over bounding box detection in many cases. For example, if a robot wants to grasp an object, an accurate mask will be much more helpful than a box. Now the main obstacle that impedes instance segmentation replacing box detection is the significantly heavier pixel-wise mask annotations. Compared to box-level annotations required by object detection, annotating pixel-level masks is notoriously time-consuming, as shown in [3,9,19]. Here we aim to eliminate this obstacle by training instance segmentation using box annotations only.

A few works [2, 7, 13, 17, 19, 22, 24, 26] attempted to obtain (semantic or instance-level) mask prediction with box-level annotations. Among them, most methods such as BoxSup [7] and Box2Seg [19] rely on the region proposals that are generated by MCG [23] or GrabCut [25]. One drawback might be the slow training procedure since these algorithms are hard to be parallelized by modern GPUs. Moreover, in order to achieve good performance, some methods often require iterative training, resulting in a complicated training pipeline and more hyper-parameters. Most importantly, *none of these methods is able to show strong weakly-supervised performance on large benchmarks such as COCO [21].* Thus almost all of them are only evaluated on small datasets such as Pascal VOC [9].

In this work, we propose a simple, single-shot and high-performance box-supervised instance segmentation method, built upon the recent fully convolutional instance segmentation framework—CondInst [28]. Our core idea is to replace the original pixel-wise mask losses in CondInst with a carefully designed mask loss consisting of two terms. The first term minimizes the discrepancy between the horizontal and vertical projections of the predicted mask and the ground-truth box (see Fig. 2 top). *This essentially ensures that the tightest box covering the predicted mask matches the ground-truth box.* Since the ground-truth mask and ground-truth box have the same projections on the two axes[1], this can be also viewed as a surrogate term that minimizes the discrepancy between the projections of the predicted mask and ground-truth mask. This loss term can be computed when we only have box annotations. Clearly, with this projection term, multiple masks can be projected to a same box. Therefore the projection loss alone would not suffice. Thus, we introduce the second loss term, supervising the *pairwise label consistency* in proximal pixels, i.e., if two pixels have the same labels or not (Fig. 2 bottom). At first glance, supervising the pairwise consistency still requires mask annotations. With only box annotations available, in principle this pairwise supervision signal is inevitably noisy. However, *an important observation is that the proximal pixels with similar colors are very likely to have the same label.* Thus, we show that it is empirically plausible to determine a color similarity threshold such that only confident pairs of pixels having a same label are used in the loss computation (the white regions in the bottom right of Fig. 2), thus largely eliminating supervision noises. Using these two loss terms, we achieve stunning instance segmentation results *without using any mask annotations*. Some qualitative results are shown in Fig. 1.

Even though ideas that are relevant to either of our two observations mentioned above were studied more or less in the literature, ranging from non-deep learning methods such as CRF [18] and GrabCut [25] to deep learning-based methods such as Box2Seg [19] and BBTP [13], none of these works effectively incorporates them into a simple and appropriate framework. As a result, and more importantly, performance of existing methods on large challenging datasets (*e.g.*, COCO) is far away from that of the full potential of box-supervised instance segmentation that is achievable, as we are going to reveal here. In summary, our method, termed **BoxInst**, enjoys the following advantages.

- The proposed method can achieve instance segmentation with box supervision by introducing two loss terms to the instance segmentation framework CondInst [28]. BoxInst is *simple* as it does not modify the network model of CondInst at all, only using different loss terms. This means that the inference process of the proposed BoxInst is exactly the same as

[1]This may not hold if the instance mask consists multiple disjointed regions.

CondInst, thus naturally inheriting all desirable properties of CondInst.

- BoxInst attains excellent instance segmentation performance on the large-scale benchmark COCO. With the ResNet-101 backbone and 3× training schedule, our BoxInst achieves 33.2% mask AP on COCO with no mask annotations used in training, *outperforming a few recent fully supervised methods using the same backbone and trained with mask annotations*, including YOLACT [4] (31.2% AP) and PolarMask [32] (32.1% AP). We empirically show that in the semi-supervised setting, mask AP of BoxInst can be further improved, as expected (§3.6).
- Since instance masks can provide much more precise localization than boxes, we envision that BoxInst can be used in many downstream tasks to boost their performance without extra effort of annotating ground-truth masks. For example, we can obtain text masks using BoxInst (see the supplementary), which often help text recognition. BoxInst can also help annotate the mask-level training data for the fully-supervised settings.

Instance segmentation has long been believed to be much more challenging to solve than bounding box detection. Our strong performance of instance segmentation using only box supervision shows that it may not necessarily be the case.

1.1. Related Work

Box-supervised Semantic Segmentation. A few works attempted to obtain semantic masks using box annotations. For example, BoxSup [7] uses the region proposals from MCG as the pseudo labels to train an FCN, and an iterative training algorithm is used to refine the estimated masks. The recent Box2Seg [19] method employs the masks generated by GrabCut to supervise training of the mask prediction model. In addition, a per-class attention map is also predicted by the model to make the per-pixel cross entropy loss focus on foreground pixels and refine the segmentation boundaries. This method shows excellent performance on Pascal VOC [9]. Authors of [26] propose to use the unsupervised CRF [18] to generate the segment proposals. Additionally, a class-wise filling rate loss to supervise the models for training, resulting in improved segmentation performance. One of the crucial steps in these methods is to employ the pseudo labels generated by unsupervised segmentation methods such as MCG [23] or GrabCut [25]. This is because these method all rely on pixel-wise mask loss functions, thus not being able to work without mask annotations. In this work, we remove the dependency on pixel-wise mask losses, as a result, eliminating the region proposals. Our new loss functions ensure that mask prediction can still be *imperfectly* supervised without using any mask annotations.

Box-supervised Instance Segmentation. In the context of deep learning, instance segmentation with box annotations has not explored too much yet. SDI [17] might be the first instance segmentation framework with box annotations. Similar to the methods for semantic segmentation, SDI also relies on the region proposals generated by MCG. Then they make use of an iterative training procedure to further refine the segmentation results. Recently, BBTP [13] formulates the box-supervised instance segmentation into a multiple instance learning (MIL) problem. BBTP is built on Mask R-CNN and samples the positive and negative bags according to the ROIs on CNN feature maps. In contrast, our method is built on ROI-free CondInst [28] and employs the proposed projection loss term to supervise the mask learning, eliminating the need for sampling. BBTP also makes use of the pairwise term. However, their pairwise term is defined on the set containing all neighboring pixel pairs with the oversimplified assumption of spatially neighboring pixel pairs being encouraged to have the same label, inevitably introducing heavily noisy supervision. The crucial prior derived from proximal pixels' colors was not exploited in BBTP. Our experiments in Table 1a show that, the heavily noisy supervision can have a negative impact on accuracy.

As a result, we significantly outperform the mask AP of BBTP on COCO by an absolute 10%.

2. Approach

Conditional Convolutions for Instance Segmentation (CondInst) Here, we briefly describe CondInst [28]. The main goal of CondInst is to solve instance segmentation in an RoI-free fully convolutional way. In CondInst, they believe that the most challenging piece in instance segmentation is that the prediction of each pixel varies according to the instance to be predicted. For example, when the model is predicting the mask for instance A, the pixels of instance B should be predicted as background. However, when the target instance is B, the pixels of B turn to be foreground. This poses the main challenge for the FCNs' application on instance segmentation because the traditional FCNs can only make deterministic prediction for each pixel.

CondInst [28] proposes to employ dynamic filters [16] to address the above issues. Instead of using a fixed mask head as in Mask R-CNN, according to the instance to be predicted, CondInst dynamically adapts the weights of the mask heads, as shown in Fig. 3, thus bypassing the above issue. With the instance-aware mask heads, CondInst can obtain the mask of each instance in the fashion of FCNs, eliminating the RoI operations. Notably, CondInst can generate the full-image instance masks, not only the masks within RoIs as in Mask R-CNN, as shown in Fig. 3. *This has played a crucial role in the box-supervised settings*. Also, the mask head only predicts class-agnostic masks. The in-

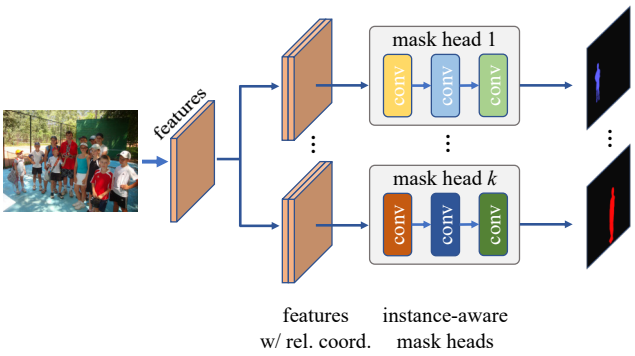

Figure 3: Illustration of CondInst [28]. CondInst employs the instance-aware dynamically-generated mask heads to obtain the full-image instance masks. We refer readers to [28] for more details.

stance's category is determined by the detector's classification branch.

2.1. Projection and Pairwise Affinity Mask Loss

Projection loss term. As mentioned before, the first term supervises the horizontal and vertical projections of the predicted mask using the ground-truth box annotation, which ensures that the tightest box covering the predicted mask matches the ground-truth box. Formally, let $\boldsymbol{b} \in \{0,1\}^{H\times W}$ be the mask generated by assigning 1 to the locations in the ground-truth box and 0 otherwise, as shown in Fig. 2 (top-right). Then we have

$$\text{Proj}_x(\boldsymbol{b}) = \boldsymbol{l}_x, \;\; \text{Proj}_y(\boldsymbol{b}) = \boldsymbol{l}_y, \tag{1}$$

where $\text{Proj}_x\colon \mathbb{R}^{H\times W} \to \mathbb{R}^W$ and $\text{Proj}_y\colon \mathbb{R}^{H\times W} \to \mathbb{R}^H$ indicate that projecting the mask onto x-axis and y-axis, respectively. $\boldsymbol{l}_x \in \{0,1\}^W$ denotes the 1-D segmentation mask on x-axis and the same applies to $\boldsymbol{l}_y$. The process of projection is illustrated in Fig. 2 (top row).

The projection operation can be implemented by a $\max$ operation along with each axis. Formally, we define

$$\begin{aligned} \text{Proj}_x(\boldsymbol{b}) &= \max_y(\boldsymbol{b}) = \boldsymbol{l}_x, \\ \text{Proj}_y(\boldsymbol{b}) &= \max_x(\boldsymbol{b}) = \boldsymbol{l}_y, \end{aligned} \tag{2}$$

where $\max_y$ and $\max_x$ are the $\max$ operations along with y-axis and x-axis, respectively.

Let $\tilde{\boldsymbol{m}} \in (0,1)^{H\times W}$ be the network predictions for the instance mask, which can be viewed as the probabilities being foreground (*i.e.*, the label is 1). We apply the same projection operations of Eq. (2) to the mask predictions and obtain the corresponding projections $\tilde{\boldsymbol{l}}_x$ and $\tilde{\boldsymbol{l}}_y$. We then compute the loss between the projections of the ground-truth box and the predicted mask. The projection loss term is defined as:

$$\begin{aligned} L_{proj} &= L(\text{Proj}_x(\tilde{\boldsymbol{m}}), \text{Proj}_x(\boldsymbol{b})) + L(\text{Proj}_y(\tilde{\boldsymbol{m}}), \text{Proj}_y(\boldsymbol{b})) \\ &= L(\max_y(\tilde{\boldsymbol{m}}), \max_y(\boldsymbol{b})) + L(\max_x(\tilde{\boldsymbol{m}}), \max_x(\boldsymbol{b})) \\ &= L(\tilde{\boldsymbol{l}}_x, \boldsymbol{l}_x) + L(\tilde{\boldsymbol{l}}_y, \boldsymbol{l}_y), \end{aligned} \tag{3}$$

where $L(\cdot,\cdot)$ is the Dice loss as in CondInst[2]. Note that *all the operations in the last equation are (sub-)differentiable*. This loss function is applied to all the instances in a training image and the final loss is their average. As shown in our experiments, by using this projection loss term, we can already obtain decent instance segmentation results without using any mask annotations.

Pairwise affinity loss term. In almost all instance segmentation frameworks such as Mask R-CNN and CondInst, they supervise the predicted masks in a per-pixel fashion. The pixelwise supervision becomes unavailable if we do not have the mask annotations. Here, we attempt to supervise the mask in a *pairwise* way, and we will show this supervision can be *partially available* even if we do not have any mask annotations.

Now, assume we have the ground-truth masks. Consider an undirected graph $G = (V, E)$ built on an image, where V is the set of the pixels in the image, and E is the set of the edges. Each pixel is connected with its $K \times K - 1$ neighbours (the dilation trick may be applied), as shown in Fig. 2 (bottom left). Then we define $y_e \in \{0,1\}$ be the label for the edge e, where $y_e = 1$ means the two pixels linked by the edge have the same ground-truth label and $y_e = 0$ means their labels are different. Let pixels (i,j) and (l,k) be the two endpoints of the edge e. The network prediction $\tilde{\boldsymbol{m}}_{i,j} \in (0,1)$ can be viewed as the probability of pixel (i,j) being foreground. Then the probability of $y_e = 1$ is

$$P(y_e = 1) = \tilde{\boldsymbol{m}}_{i,j} \cdot \tilde{\boldsymbol{m}}_{k,l} + (1 - \tilde{\boldsymbol{m}}_{i,j}) \cdot (1 - \tilde{\boldsymbol{m}}_{k,l}), \tag{4}$$

and $P(y_e = 0) = 1 - P(y_e = 1)$. By convention, the probability distribution from the network prediction can be trained with the binary cross entropy (BCE) loss. Thus, the loss function is

$$\begin{aligned} L_{pairwise} = -\frac{1}{N} \sum_{e\in E_{in}} & y_e \log P(y_e = 1) \\ & + (1 - y_e) \log P(y_e = 0), \end{aligned} \tag{5}$$

where E_{in} is the set of the edges containing at least one pixel in the box. Using E_{in} instead of E here can prevent the loss from being dominated by a large number of the pixels outside the box. N is the number of the edges in E_{in}.

If only the pairwise loss is used to supervise the mask learning (in the fully-supervised setting), ideally, two possible solutions may be obtained. The first one is the same as

[2]One can also use the cross-entropy loss here.

the ground-truth mask $\boldsymbol{m}$, which is desirable. The second solution is the inverse $1 - \boldsymbol{m}$. Fortunately, the second solution can be easily eliminated as long as we have a resolved label for any pixel. This can be achieved by the projection loss term because it ensures that the pixels outside the box is background. Note that the edges in E_{in} still involve some pixels outside the box, which are of great importance to help the model get rid of the undesirable solutions. Overall, the total loss for mask learning can be formulated as

$$L_{mask} = L_{proj} + L_{pairwise}. \tag{6}$$

We will show in experiments that the redesigned mask loss can have similar performance to the original pixelwise one in the fully-supervised settings.

2.2. Learning without Mask Annotations

So far, we have shown that we can employ Eq. (6) to supervise the masks. In Eq. (6), the first term L_{proj} is always valid no matter we have box or mask annotations. At first glance, the second term $L_{pairwise}$ still requires the mask annotations to compute the edge's label y_e. However, an important observation is that if two pixels have similar colors, they are very likely to have the same labels as well (*i.e.*, the corresponding edge's label is 1). Thus, we may determine a color similarity threshold τ such that the edge's label is 1 with a high probability if its color similarity is above τ. Formally, let us define the color similarity as

$$S_e = S(\boldsymbol{c}_{i,j}, \boldsymbol{c}_{l,k}) = \exp\left(-\frac{||\boldsymbol{c}_{i,j} - \boldsymbol{c}_{l,k}||}{\theta}\right), \tag{7}$$

where S_e be the color similarity of the edge e, and $\boldsymbol{c}_{i,j}$ and $\boldsymbol{c}_{l,k}$ are, respectively, the color vectors of the two pixels (i, j) and (l, k) linked by the edge. Here we use the LAB color space as it is closer to human perception. θ is a hyper-parameter, being 2 in this work.

In order to confirm our hypothesis above, we visualize the proportion of the positive edges in all the edges with color similarity above the threshold τ on the COCO `val2017` split. Fig. 4 (blue curve) shows that the proportions of the positive edges in all the edges with $S_e \geq \tau$ as the threshold τ increases. As shown in figure, if the threshold is 0.1, more than 98% of the edges are positive. The proportion can be further improved if we continue to increase τ, but an overlarge threshold would reduce the number of the supervised edges (red curve in Fig. 4). In experiments, we found that the threshold is not sensitive to the final performance.

Given the high proportion of positive edges, during training, we can safely assume that all the edges with $S_e \geq \tau$ are positive and then only compute the pairwise loss for them. Other edges are discarded during the loss computation. As a result, the pairwise loss becomes

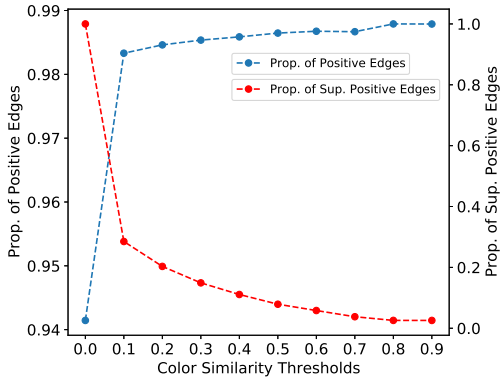

Figure 4: The relationship between the edges' labels and the color similarity thresholds. 'blue curve': the proportion of the positive edges in all the edges with color similarity above the threshold. 'red curve': the proportion of the supervised positive edges in all the positive edges. The number of positive edges are computed with the ground-truth masks of the COCO `val2017` split.

$$L_{pairwise} = -\frac{1}{N}\sum_{e \in E_{in}} \mathbb{1}_{\{S_e \geq \tau\}} \log P(y_e = 1), \tag{8}$$

where $\mathbb{1}_{\{S_e \geq \tau\}}$ is the indicator function, being 1 if $S_e \geq \tau$ and 0 otherwise. Eq. (8) only involves the term in Eq. (5) for positive edges because we can only infer that an edge e is positive if $S_e \geq \tau$. If $S_e < \tau$, the label is agnostic. As we only have positive labels in Eq. (8), one may note this would result in two possible trivial solutions, *i.e.*, the masks of all the pixels being 0 or 1. However, the masks with all pixels being 0 do not meet the projection term; and the masks of all pixels being 1 almost never appear since the pairwise term encourages the pixels near the box boundaries to be negative if their colors are similar to that of the negative pixels outside the box.

3. Experiments

We conduct experiments on COCO [21] and Pascal VOC [9]. For COCO, the models are trained with `train2017` (115K images) and the ablation experiments are evaluated on `val2017` (5K images). Unless specified, *only box annotations* are used during training. Our main results are reported on `test-dev`. For Pascal VOC, following previous works [13, 17], we train the models on the augmented Pascal VOC 2012 dataset [10] with 10, 582 training images, and evaluate them on Pascal VOC 2012 `val` split with 1, 449 images.

3.1. Implementation Details

Our proposed method only requires to change the mask loss in CondInst. Other training and testing details are kept

	prop.	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
fully-sup.	-	35.4	55.9	37.6	17.0	38.8	50.7
$\tau = 0$	94.1%	9.4	30.3	3.3	7.6	10.3	11.4
$\tau = 0.1$	98.3%	**30.7**	52.2	**31.1**	13.8	**33.1**	**45.7**
$\tau = 0.2$	**98.4%**	30.6	**52.6**	30.9	**13.9**	32.8	45.5

(a) Varying the color similarity threshold τ.

size	dilation	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
3	1	29.7	52.0	29.6	13.4	32.3	44.4
3	2	**30.7**	52.2	**31.1**	**13.8**	**33.1**	**45.7**
5	1	30.5	**52.3**	30.7	13.7	33.0	**45.7**
5	2	29.9	51.9	30.0	13.8	32.1	45.0

(b) Varying the size and dilation of the local patches (with $\tau = 0.1$).

Table 1: Ablation study on the hyper-parameters in the proposed mask loss on the COCO `val2017` split. "prop." is the proportion of the positive edges in the edges with $S_e \geq \tau$. "fully-sup.": fully-supervised results. As shown in Table 1a, by using $\tau = 0.1$, BoxInst can achieve 30.7 mask AP with only box annotations, which is close the fully-supervised mask AP (35.4%) and significantly better localization precision than boxes (10.6% mask AP as shown in Table 3). Table 1b shows the experiment results by varying the neighbours of each pixel.

mask loss	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
Dice loss	35.6	56.3	37.8	16.9	38.9	51.0
proposed	35.4	55.9	37.6	17.0	38.8	50.7

Table 2: The projection and pairwise affinity mask loss vs. the original pixelwise one in the fully-supervised settings. As we can see here, they attain very similar mask AP on the COCO split `val2017`.

L_{proj}	$L_{pairwise}$	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
box mask		10.6	32.2	4.6	5.7	11.3	15.6
✓		21.2	45.2	17.7	10.0	21.4	32.5
✓	✓	**30.7**	**52.2**	**31.1**	**13.8**	**33.1**	**45.7**

Table 3: The mask AP on COCO `val2017` by applying the different loss terms. "box mask": using the masks generated by boxes. If both terms are not used, the model can only provide the box-level localization precision (10.6% mask AP).

as similar as possible to the original CondInst. On COCO, unless specified, we use the following training details. The models are trained for 90K iterations with batch size 16 on 8 V100 GPUs (2 images per GPU). The initial learning rate is set to 0.01 and reduced by a factor of 10 at step 60K and 80K, respectively. ResNet-50 [12] is used as the backbone and is initialized with the ImageNet [8] pre-training weights. The newly added layers are initialized as in [28]. We use exactly the same data augmentation as in CondInst (*e.g.*, left-right flipping and multi-scale training). For the dynamic mask heads, we use 3 conv. layers as in CondInst, but we increase the channels of the mask heads from 8 to 16, which can result in better performance with negligible extra computational overheads, and the compared baselines are adjusted accordingly. Following CondInst, the output mask is up-sampled to $1/4$ resolution of the input image. For the pairwise loss term, we compute the pairwise relationship within 3×3 patches with the dilation rate being 2. On Pascal VOC, following [13], we use batch size 8 and the number of iterations is 20K. The learning rate is reduced by a factor of 10 at step 15K. Only left-right flipping is used as the data augmentation during training. Other settings are the same as on COCO. The inference is the same as the original CondInst on both benchmarks. The performance is evaluated with the COCO-style mask AP.

3.2. Projection and Pairwise Affinity Loss for Mask Learning

We first demonstrate that the redesigned mask loss can have similar performance to the original pixelwise mask loss in the fully-supervised settings. The experiments are conducted on COCO. We replace the original Dice loss for mask training in CondInst with the proposed one, and keep other settings exactly the same. As shown in Table 2, the proposed mask loss can have similar performance (35.4% vs. 35.6% mask AP), which suggests that using the proposed loss for mask learning is feasible.

3.3. Box-supervised Instance Segmentation

The key advantage of the proposed mask loss is that it can still supervise the predicted masks with only box annotations. We confirm this here and conduct experiments to investigate the hyper-parameters in the proposed mask loss.

Varying the threshold of color similarity. As mentioned before, we use a color similarity threshold τ to determine the edges that will be used to compute the pairwise loss. Here, we conduct experiments by varying τ. When $\tau = 0$, all of the edges defined by the size of neighborhood are considered positive and used in the loss computation, as shown in Eq. (8). As shown in Table 1a, in this case, 94.1% of the edges are truly positive, and $\sim$6% of the edges are actually negative. Thus, the loss computation would introduce $\sim$6% noisy labels and all of the truly negative edges are wrongly labelled positive. Unsurprisingly, this experiment yields a trivial solution with poor performance (9.4% mask AP) that almost all pixels in the box are predicted as foreground. If we increase τ to 0.1, the proportion of the truly positive edges are improved to 98.3%, and only less than $\sim$2% of the edges are wrongly labelled. As a result, the model can yield high-quality instance masks, achieving 30.7% mask AP (vs. fully-supervised counterpart 35.4%). This result is even better than that of some fully-supervised methods such as YOLACT and PolarMask. Some qualitative results are shown in Fig. 1. If we further increase the threshold τ

method	backbone	aug.	sched.	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
fully supervised methods:									
Mask R-CNN [11]	ResNet-50-FPN	✓	3×	37.5	59.3	40.2	21.1	39.6	48.3
CondInst [28]	ResNet-50-FPN	✓	3×	37.8	59.1	40.5	21.0	40.3	48.7
Mask R-CNN	ResNet-101-FPN	✓	3×	38.8	60.9	41.9	21.8	41.4	50.5
YOLACT-700 [4]	ResNet-101-FPN	✓	4.5×	31.2	50.6	32.8	12.1	33.3	47.1
PolarMask [32]	ResNet-101-FPN	✓	2×	32.1	53.7	33.1	14.7	33.8	45.3
CondInst	ResNet-101-FPN	✓	3×	39.1	60.9	42.0	21.5	41.7	50.9
box-supervised methods:									
BBTP† [13] (prev. best)	ResNet-101-FPN		1×	21.1	45.5	17.2	11.2	22.0	29.8
BoxInst†	ResNet-101-FPN		1×	31.6	54.0	31.9	13.9	34.2	48.2
BoxInst	ResNet-50-FPN	✓	3×	32.1	55.1	32.4	15.6	34.3	43.5
BoxInst	ResNet-101-FPN	✓	1×	32.5	55.3	33.0	15.6	35.1	44.1
BoxInst	ResNet-101-FPN	✓	3×	33.2	56.5	33.6	16.2	35.3	45.1
BoxInst	ResNet-101-BiFPN [27]	✓	3×	33.9	57.7	34.5	16.5	36.1	46.6
BoxInst	ResNet-DCN-101-BiFPN [34]	✓	3×	**35.0**	**59.3**	**35.6**	**17.1**	**37.2**	**48.9**

Table 4: Comparisons with state-of-the-art methods on the COCO `test-dev` split. "†" means that the results are on the COCO `val2017` split. BBTP only reported the results on the `val2017` split. Our BoxInst outperforms the previous best reported mask AP by over absolute 10% mask AP. Ours even outperforms two recent fully supervised methods, YOLACT and PolarMask, and is close to state-of-the-art fully-supervised results. 'DCN': deformable convolutions [34]. '1×' means 90K iterations.

to 0.2, the performance slightly drops to 30.6% mask AP. This might be because the number of the supervised positive edges decreases as we increase the threshold, as shown in Fig. 4.

Varying the neighborhood of the pixels. We conduct experiments with the different neighbours for each pixel. The size (*i.e.*, K) defines how many surrounding pixels of each pixel are used to compute the pairwise loss with the pixel. Additionally, we may use the dilation trick to enlarge the scope (as in dilated convolutions). As shown in Table 1b, by increasing the size from 3×3 to 5×5, the performance is boosted from 29.7% to 30.5%. This suggests that a relatively long-distance pairwise relationship is important to the final performance. However, using 5×5 requires more computational overheads in the training. Thus, we apply the dilation rate 2 to the 3×3 patches. This can capture the long-distance relationship without increasing the computational overheads and achieves similar performance (30.7%). The performance cannot be further improved by applying the dilation trick to the 5×5 patches because the assumption, two pixels with similar colors probably have the same label, might not hold if the two pixels are far from each other.

The contribution of each loss term. Table 3 shows the contribution of each loss term. Even if only the first projection term is used, we can also achieve decent performance (21.2% mask AP), which can already provide much higher localization precision than boxes (10.6% mask AP). By further using the proposed pairwise term, high-quality instance masks can be obtained and the performance is largely improved to 30.7%.

Failure cases. We also show some failure cases in Fig. 5. BoxInst sometimes fails to predict the correct masks (shown in the red boxes) because there are still multiple masks satisfying the constraints imposed by the two loss terms. It is not trivial to eliminate these wrong masks, which will be left for a future research.

Figure 5: Some failure cases on COCO. The incorrect parts are in the red boxes.

3.4. Comparisons with State-of-the-art on COCO

We compare BoxInst with state-of-the-art fully/box supervised instance segmentation methods on the COCO dataset. As shown in Table 4, with the same backbone and training settings, BoxInst significantly surpasses the previous best reported result [13] by absolute 10.5% mask AP (*e.g.*, from 21.1% to 31.6%). *BoxInst, without using any mask annotations, performs even better than some recent fully-supervised methods such as PolarMask [32] and YOLACT [4]*, with the same backbones as well as similar training and testing settings (32.5% with R-101 1× vs. PolarMask 32.1% R-101 2× and YOLACT-700 31.2% R-101 4.5×). BoxInst also demonstrates competitive performance with top-performing fully-supervised instance segmentation methods. For example, with the same backbone ResNet-50-FPN and 3× training schedule, BoxInst achieves 32.1% mask AP (vs. 37.8% of the fully-supervised CondInst [28]). With BiFPN [27] and DCN [34], the performance can be further boosted to 35.0% mask AP. Some qualitative results are shown in Fig. 1. The excellent perfor-

method	backbone	AP	AP_{50}	AP_{75}
GrabCut [25]	ResNet-101	17.8	37.8	15.5
SDI [17]	VGG-16	-	44.8	16.3
BBTP [13]	ResNet-101	23.1	54.1	17.1
BBTP w/ CRF	ResNet-101	27.5	59.1	21.9
BBTP*	ResNet-101	20.5	51.1	14.3
BBTP* w/ CRF	ResNet-101	25.0	56.9	18.9
BoxInst	ResNet-50	32.2	58.1	31.0
BoxInst	ResNet-101	**34.4**	**60.1**	**34.6**

Table 5: Results on Pascal VOC `val2012`. Here, BBTP* denotes the results after we fix the issue [1] in its Matlab evaluation code. Clearly, BoxInst achieves significantly improved mask AP, outperforming previous best by about 10%. Here, the GrabCut obtains the instance masks by taking as input the boxes generated by BoxInst. Thus, the only difference between the GrabCut and BoxInst is the way to obtain the masks.

mance shows that BoxInst dramatically narrows the performance gap between the fully supervised and box-supervised instance segmentation, and for the first time, the great potential of box-supervised instance segmentation is revealed.

3.5. Experiments on Pascal VOC

We also conduct experiments on Pascal VOC. As shown in Table 5, BoxInst achieves state-of-the-art instance segmentation with only box annotations. With the same backbone and training settings, BoxInst outperforms BBTP both in AP_{50} and AP_{75} by a large margin. Notably, the AP_{75} is improved by more than relative 200% (14.3% vs. 34.6% mask AP), which suggests BoxInst can produce the masks of much higher quality. BoxInst is even much better than the BBTP with CRF. Additionally, BoxInst also performs much better than SDI [17]. We also compare BoxInst with the traditional unsupervised segmentation method GrabCut [25]. In the experiment, GrabCut takes as input the bounding-boxes predicted by the ResNet-101 based FCOS in BoxInst. Thus the only difference between BoxInst and GrabCut is the way of obtaining instance masks. As shown in Table 5, BoxInst is far better than GrabCut (17.8% vs. 34.4% mask AP). Moreover, BoxInst is fully convolutional and can benefit from the highly-efficient GPUs, thus inferring tens of times faster than GrabCut.

3.6. Semi-supervised Instance Segmentation

In this section, we show that our method can also help the model generalize to unseen categories in the semi-supervised setting where only partial classes have the mask annotations. Following previous works [14, 20, 33] in this setting, we conduct the experiments on the COCO dataset and split the 80 classes in COCO into two groups – 20 classes present in Pascal VOC and 60 classes not in Pascal VOC. Then the models are trained with the mask annotations of the classes in one group, and another group of classes only have the box annotations. The generalization ability is evaluated with the mask AP averaged over the group of classes without mask annotations (*i.e.*, unseen classes).

L_{proj}	$L_{pairwise}$	all 80 classes			60 unseen classes		
		AP	AP_{50}	AP_{75}	AP	AP_{50}	AP_{75}
		24.7	44.6	24.2	19.9	38.3	18.5
✓		31.8	52.5	33.2	29.7	49.3	31.0
✓	✓	**32.5**	**53.0**	**34.0**	**30.9**	**50.1**	**32.4**
box supervised		30.7	52.2	31.1	29.6	49.7	30.4

Table 6: BoxInst for semi-supervised instance segmentation. These models are trained with the 20 classes mask annotations and the other 60 classes (*i.e.*, unseen classes) are only with box annotations.

L_{proj}	$L_{pairwise}$	all 80 classes			20 unseen classes		
		AP	AP_{50}	AP_{75}	AP	AP_{50}	AP_{75}
		32.1	51.6	33.9	25.5	45.5	25.1
✓		33.1	53.8	34.3	31.6	57.4	30.0
✓	✓	**33.8**	**54.3**	**35.7**	**35.9**	**60.9**	**36.3**
box supervised		30.7	52.2	31.1	29.6	49.7	30.4

Table 7: BoxInst for semi-supervised instance segmentation. The models are trained with the 60 classes mask annotations and other 20 classes (*i.e.*, unseen classes) are only with box annotations.

We first train the model with the 20 classes mask annotations. As shown in Table 6 (1st row), if our proposed loss terms are not used, where the mask loss is only computed for the instances with mask annotations and other instances are discarded during the mask learning, the model can only achieve 19.9% mask AP on the unseen categories. This low performance suggests that the model is difficult to generalize to unseen classes. If we use the L_{proj} term for the 60 classes without the mask annotations during training, as shown in the table (2nd row), the performance can be dramatically improved to 29.7%. If we further apply the pairwise term $L_{pairwise}$, the performance can be boosted to 30.9%. Moreover, compared to the setting only using the box annotations (last row in Table 6), the performance on the unseen classes is also improved from 29.6% to 30.9%, which suggests that the box-supervised model can benefit from the partial mask annotations. Additionally, the experimental results with the 60 classes masks are shown in Table 7, and the same conclusions can be drawn.

4. Conclusions

In this work, we have proposed BoxInst that can achieve high-quality instance segmentation with only box annotations. The core idea of BoxInst is to replace the original pixelwise mask loss with the proposed projection and pairwise affinity mask loss. With the proposed mask loss, we show excellent instance segmentation performance without using any mask annotations on COCO and Pascal VOC, significantly improving the state-of-the-art.

References

[1] https://github.com/chengchunhsu/WSIS_BBTP/issues/11. 8

[2] Aditya Arun, C. V. Jawahar, and Pawan Kumar. Weakly supervised instance segmentation by learning annotation consistent instances. In *Proc. Eur. Conf. Comp. Vis.*, 2020. 2

[3] Amy Bearman, Olga Russakovsky, Vittorio Ferrari, and Li Fei-Fei. What's the point: Semantic segmentation with point supervision. In *Proc. Eur. Conf. Comp. Vis.*, pages 549–565, 2016. 2

[4] Daniel Bolya, Chong Zhou, Fanyi Xiao, and Yong Jae Lee. YOLACT: Real-time instance segmentation. In *Proc. IEEE Int. Conf. Comp. Vis.*, 2019. 3, 7

[5] Hao Chen, Kunyang Sun, Zhi Tian, Chunhua Shen, Yongming Huang, and Youliang Yan. Blendmask: Top-down meets bottom-up for instance segmentation. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, pages 8573–8581, 2020. 1

[6] Tianheng Cheng, Xinggang Wang, Lichao Huang, and Wenyu Liu. Boundary-preserving mask R-CNN. In *Proc. Eur. Conf. Comp. Vis.*, 2020. 1

[7] Jifeng Dai, Kaiming He, and Jian Sun. Boxsup: Exploiting bounding boxes to supervise convolutional networks for semantic segmentation. In *Proc. IEEE Int. Conf. Comp. Vis.*, pages 1635–1643, 2015. 2, 3

[8] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. ImageNet: A large-scale hierarchical image database. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, pages 248–255, 2009. 6

[9] Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The Pascal visual object classes (VOC) challenge. *Int. J. Comput. Vision*, 88(2):303–338, 2010. 2, 3, 5

[10] Bharath Hariharan, Pablo Arbeláez, Lubomir Bourdev, Subhransu Maji, and Jitendra Malik. Semantic contours from inverse detectors. In *Proc. IEEE Int. Conf. Comp. Vis.*, pages 991–998, 2011. 5

[11] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask R-CNN. In *Proc. IEEE Int. Conf. Comp. Vis.*, pages 2961–2969, 2017. 1, 7

[12] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, pages 770–778, 2016. 6

[13] Cheng-Chun Hsu, Kuang-Jui Hsu, Chung-Chi Tsai, Yen-Yu Lin, and Yung-Yu Chuang. Weakly supervised instance segmentation using the bounding box tightness prior. In *Proc. Advances in Neural Inf. Process. Syst.*, 2019. Note that the mask AP results on COCO are in the supplementary, available at https://tinyurl.com/yyjovxn6. 1, 2, 3, 5, 6, 7, 8

[14] Ronghang Hu, Piotr Dollár, Kaiming He, Trevor Darrell, and Ross Girshick. Learning to segment every thing. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, pages 4233–4241, 2018. 8

[15] Zhaojin Huang, Lichao Huang, Yongchao Gong, Chang Huang, and Xinggang Wang. Mask scoring R-CNN. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, pages 6409–6418, 2019. 1

[16] Xu Jia, Bert De Brabandere, Tinne Tuytelaars, and Luc Van Gool. Dynamic filter networks. In *Proc. Advances in Neural Inf. Process. Syst.*, pages 667–675, 2016. 3

[17] Anna Khoreva, Rodrigo Benenson, Jan Hosang, Matthias Hein, and Bernt Schiele. Simple does it: Weakly supervised instance and semantic segmentation. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, pages 876–885, 2017. 2, 3, 5, 8

[18] Philipp Krähenbühl and Vladlen Koltun. Efficient inference in fully connected CRFs with Gaussian edge potentials. In *Proc. Advances in Neural Inf. Process. Syst.*, pages 109–117, 2011. 2, 3

[19] Viveka Kulharia, Siddhartha Chandra, Amit Agrawal, Philip Torr, and Ambrish Tyagi. Box2Seg: Attention weighted loss and discriminative feature learning for weakly supervised segmentation. In *Proc. Eur. Conf. Comp. Vis.* 2, 3

[20] Weicheng Kuo, Anelia Angelova, Jitendra Malik, and Tsung-Yi Lin. ShapeMask: Learning to segment novel objects by refining shape priors. In *Proc. IEEE Int. Conf. Comp. Vis.*, pages 9207–9216, 2019. 8

[21] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and Lawrence Zitnick. Microsoft COCO: Common objects in context. In *Proc. Eur. Conf. Comp. Vis.*, pages 740–755, 2014. 2, 5

[22] George Papandreou, Liang-Chieh Chen, Kevin Murphy, and Alan Yuille. Weakly- and semi-supervised learning of a deep convolutional network for semantic image segmentation. In *Proc. IEEE Int. Conf. Comp. Vis.*, pages 1742–1750, 2015. 2

[23] Jordi Pont-Tuset, Pablo Arbelaez, Jonathan Barron, Ferran Marques, and Jitendra Malik. Multiscale combinatorial grouping for image segmentation and object proposal generation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 39(1):128–140, 2016. 2, 3

[24] Martin Rajchl, Matthew Lee, Ozan Oktay, Konstantinos Kamnitsas, Jonathan Passerat-Palmbach, Wenjia Bai, Mellisa Damodaram, Mary Rutherford, Joseph Hajnal, Bernhard Kainz, et al. Deepcut: Object segmentation from bounding box annotations using convolutional neural networks. *IEEE Trans. Medical Imaging*, 36(2):674–683, 2016. 2

[25] Carsten Rother, Vladimir Kolmogorov, and Andrew Blake. GrabCut: interactive foreground extraction using iterated graph cuts. *ACM Trans. Graphics*, 23(3):309–314, 2004. 2, 3, 8

[26] Chunfeng Song, Yan Huang, Wanli Ouyang, and Liang Wang. Box-driven class-wise region masking and filling rate guided loss for weakly supervised semantic segmentation. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, pages 3136–3145, 2019. 2, 3

[27] Mingxing Tan, Ruoming Pang, and Quoc V. Le. EfficientDet: Scalable and efficient object detection. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, pages 10778–10787, 2020. 7

[28] Zhi Tian, Chunhua Shen, and Hao Chen. Conditional convolutions for instance segmentation. In *Proc. Eur. Conf. Comp. Vis.*, 2020. 1, 2, 3, 4, 6, 7

[29] Zhi Tian, Chunhua Shen, Hao Chen, and Tong He. FCOS: Fully convolutional one-stage object detection. In *Proc. IEEE Int. Conf. Comp. Vis.*, 2019. 1

[30] Xinlong Wang, Tao Kong, Chunhua Shen, Yuning Jiang, and Lei Li. SOLO: Segmenting objects by locations. In *Proc. Eur. Conf. Comp. Vis.*, 2020. 1

[31] Xinlong Wang, Rufeng Zhang, Tao Kong, Lei Li, and Chunhua Shen. SOLOv2: Dynamic and fast instance segmentation. In *Proc. Advances in Neural Inf. Process. Syst.*, 2020. 1

[32] Enze Xie, Peize Sun, Xiaoge Song, Wenhai Wang, Xuebo Liu, Ding Liang, Chunhua Shen, and Ping Luo. Polarmask: Single shot instance segmentation with polar representation. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, pages 12193–12202, 2020. 3, 7

[33] Yanzhao Zhou, Xin Wang, Jianbin Jiao, Trevor Darrell, and Fisher Yu. Learning saliency propagation for semi-supervised instance segmentation. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, 2020. 8

[34] Xizhou Zhu, Han Hu, Stephen Lin, and Jifeng Dai. Deformable convnets v2: More deformable, better results. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, pages 9308–9316, 2019. 7