Center-context-gap Refinement for Weakly Supervised Instance Segmentation

Anonymous Authors

Abstract

This paper proposes a weakly supervised instance segmentation method that only requires image-level labels for training. The proposed architecture consists of three branches, the multiple instance learning branch, the semantic segmentation branch, and the instance segmentation branch. The multiple instance learning branch selects bounding boxes indicating locations of instances. The semantic segmentation branch produces semantic segmentation results of delineating shapes of instances. By combining the selected boxes and the generated segmentation results, we produce pseudo instance segmentation labels for training the instance segmentation branch. For further improving the quality of the generated pseudo labels, we propose the center-context-gap refinement module and it significantly boosts the performance. The experiments show that the proposed method achieves favorable performance against the state-of-the-art weakly supervised methods.

Introduction

Instance segmentation (He et al. 2017; Novotny et al. 2018) aims at jointly detecting and segmenting each individual instance in an image. It has become a key component of scene understanding because it provides detailed information for scene analysis, such as object locations, object shapes, and numbers of object instances. As such, instance segmentation is valuable and beneficial to many high-level vision applications, e.g., autonomous driving (Zhang, Fidler, and Urtasun 2016), visual question answering (Gan et al. 2017) and image and sentence matching (Huang, Wang, and Wang 2017).

Instance segmentation is a challenging task because both correct detection and precise segmentation are required. Recently, significant progress on instance segmentation (He et al. 2017; Novotny et al. 2018) has been made with *convolutional neural networks* (CNNs) (Krizhevsky, Sutskever, and Hinton 2012) which enables joint visual features and nonlinear classifier learning, but the sufficient instance-level pixel-wise training annotations are required. It is expensive to collect such data as typically these annotations are manually drawn or delineated by tools with intensive user interaction.

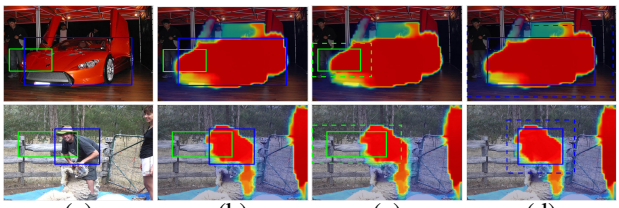

Figure 1: The effect of the proposed CCGR module. Each row gives an example. Green and blue rectangles are respectively the bounding boxes before and after refinement. In (a) and (b), we show the input images and the predicted segmentation score maps with the bounding boxes, respectively. It is clear that the initial (green) boxes often focus on the discriminating regions and cannot cover the whole objects. After refinement, the refined (blue) boxes better cover the whole objects. In (c) and (d), dotted rectangles show the enlarged boxes indicating the surrounding context of the bounding boxes. In (c), both the unrefined boxes and their context regions contain high scores in the segmentation maps. In (d), after refinement, the refined boxes contains high scores while their context regions contain much lower scores, meaning the boxes cover instances more tightly.

Therefore, the heavy annotation cost of collecting training data becomes a barrier from its application in potential tasks.

To alleviate the heavy cost, this paper focuses on weakly supervised instance segmentation with only image-level annotations provided. Image-level annotations can be collected more easily, thus significantly reducing the annotation cost. To achieve this goal, we proposes a CNN architecture containing three branches: the *multiple instance learning* (MIL) branch, the *semantic segmentation* (SS) branch, and the *instance segmentation* (IS) branch shown in Figure 2. The MIL branch selects the bounding boxes that best cover the inferred object instances. The SS branch produces semantic segmentation results. The selected boxes from the MIL branch indicate the locations of instances while the semantic segmentation results from the SS branch delineates the shapes of instances. By combining them together, we can generate pseudo instance segmentation labels for training

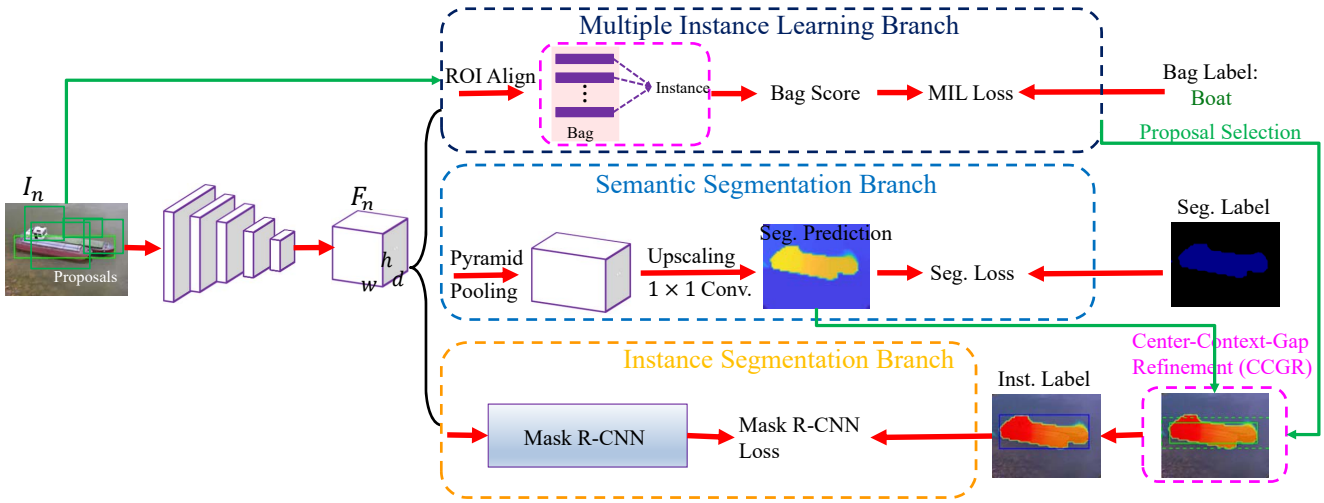

Figure 2: Overview of the proposed network. We first extract the features of an input with the shared convolutional layers. Then, these extracted features are fed into the multiple instance learning (MIL) branch and the semantic segmentation (SS) branch for selecting the object proposals and generating the segmentation score maps respectively. Based on the selected proposals and the score maps, center-context-gap refinement (CCGR) further refines the box proposal and generates the pseudo instance-level labels for training the instance segmentation (IS) branch.

the IS branch. The optimization is iterative, and the results of IS branch can be further fed to the MIL branch and segmentation branch for generating pseudo ground truth with better quality, consequently improving instance segmentation.

The quality of the pseudo ground truth is key to the performance. Therefore, we propose a novel refinement scheme, *center-context-gap refinement* (CCGR), to refine the bounding boxes based on the segmentation results, and then generate better pseudo instance-level labels for training the IS branch. In CCGR, we would maximize the segmentation scores within the box to encourage it only contain the whole instance. At the same time, we would minimize the segmentation scores of the surrounding context of the box to urge it cover less instance regions. By maximizing inside the box and minimizing within its surrounding, the gap between the center box and its context is enlarged. Thus, we call it as the center-context-gap refinement. The proposed CCGR is differentiable, and it can be integrated into our model and optimized by simply using gradient descent. Figure 1 gives examples motivating the use of the CCGR module.

This paper makes the following contributions. First, we propose a CNN architecture to address weakly supervised instance segmentation. The proposed method trains an instance segmentation model with only the image-level supervision. It is simple and flexible because the architecture in each branch could be replaced with any method. Second, an iterative optimization scheme is adopted to efficiently optimize the proposed architecture, and a novel refinement scheme, center-context-gap refinement (CCGR), is proposed to repair the poor bounding boxes. CCGR is differentiable, and the gradient descent method can be employed for searching the optimal bounding box. Finally, we comprehensively evaluate our method on two standard benchmarks for instance segmentation, PASCAL VOC 2012 (Everingham et al. 2010) and the newly collected COCO-VOC (Hsu, Lin, and Chuang 2019). The results show that our method can achieve better than the state-of-the-art weakly supervised instance segmentation methods.

Related Work

Weakly supervised semantic segmentation. State-of-the-art semantic segmentation methods (Lin et al. 2019; Zhang et al. 2018; Chen et al. 2018) are built upon CNNs (Krizhevsky, Sutskever, and Hinton 2012). Despite the effectiveness, training CNNs for semantic segmentation requires a vast amount of manually-drawn pixel-wise object annotations. To reduce the burden of pixel-wise annotations, various types of coarse annotations are employed as the supervision, such as bounding boxes (Khoreva et al. 2017), scribbles (Tang et al. 2018), points (Qian et al. 2019), and image-level labels (Hou et al. 2018; Wei et al. 2018; Ahn and Kwak 2018; Shen et al. 2018; Wang et al. 2018). We focus on the image-level supervision whose annotation cost is the least. However, different from these methods, the paper works on instance segmentation which is a more challenging task than semantic segmentation.

Instance segmentation with pixel-wise annotations. Conventional researches on instance segmentation can be divided into two categories, namely the proposal-based methods (He et al. 2017; Novotny et al. 2018) and the proposal-free methods (Liu et al. 2018). The former first detects bounding boxes and then segments the instances within the bounding boxes with *region proposal network* (RPN) (Ren et al. 2015) while the latter directly segment each instance without the detection results. Despite the effectiveness and efficiency, these methods for instance segmentation rely on a

large number instance-aware pixel-wise annotations to learn their models, but different from them, the proposed method can be trained with only the image-level supervision.

Instance segmentation with low annotation cost. To reduce the annotation cost, few methods address instance segmentation with the coarse supervision including bounding boxes (Khoreva et al. 2017) and image-level labels (Zhou et al. 2018; Zhu et al. 2019; Cholakkal et al. 2019). PRM (Zhou et al. 2018) is the first deep method using the image-level labels for instance segmentation. In PRM, the peak in the classifier response map are regarded as the potential object instance. Thus, the peak propagation is proposed to exploit the peak cue, and then produce the instance-aware cue to search the top-ranked proposal as final results during inference. Although PRM is effective, it has two issues: 1) It only recognizes the discriminative object part resulting in the higher classifier response. 2) The inference cost is high because its inference involves back propagation, proposal generation and proposal ranking. Recently, to address the first limitation, IAM (Zhu et al. 2019) leverages incomplete region responses from PRM and the corresponding matching proposals to learn a CNN-based extent module. This module can fill each pixel with the value from its neighbors to make the object region more complete. However, the additional parameters in this module should be learned. Our CCGR is learning-free, so no additional parameters are included. Cholakkal et al. (Cholakkal et al. 2019) integrate the addition density branch into the PRM framework with the additional object count information, and this branch can predict the global object count and preserve the spatial distribution of objects. However, they require additional information of the object count, but our method doesn't.

Proposed Method

In this section, we first give an overview of the proposed method, and then describe the network architecture with three branches: the *multiple instance learning (MIL) branch*, *semantic segmentation (SS) branch* and *instance segmentation (IS) branch* in detail. Next, we explain the proposed center-context-gap refinement. Finally, we describe the optimization strategy and provide implementation details.

Overview

In our weakly supervised setting, the training set $D = \{I_n, \mathbf{L}_n\}_{n=1}^{N}$ contains a set of images I_n and their corresponding labels $\mathbf{L}_n$. The label $\mathbf{L}_n$ is a C-d vector indicating which class the image I_n contains, where C is the number of classes. Given the training set D, our goal for weakly-supervised instance segmentation is to learn a model that can jointly detect and segment each instance in an image.

Figure 2 depicts the proposed architecture consisting of three branches, MIL, SS and IS, with the shared convolutional layers, and an additional refinement module, CCGR. Following (Zhou et al. 2018; Cholakkal et al. 2019; Zhu et al. 2019), we use the multi-scale combinatorial grouping (MCG) (Pont-Tuset et al. 2017) to extract proposals. For an image I_n, we use $\mathbb{P}_n$ to denote its set of object proposals. The MIL branch is designed to select box proposals that potentially correspond to instances from $\mathbb{P}_n$. The SS branch is designed to generate segmentation score maps indicating the likelihood that each pixel belongs to a specific class. By combining a selected box and the score map, we can obtain the pseudo instance segmentation label for an instance. With the pseudo labels, the IS branch built on Mask R-CNN (He et al. 2017) can be trained for instance segmentation. For improving the quality of the pseudo labels, the CCGR module is designed to refine a selected box proposal with the help of the corresponding segmentation score map.

The network is trained by minimizing the following loss:

$$\mathcal{L}(\mathbf{w}) = \mathcal{L}_{MIL}(\mathbf{w}) + \mathcal{L}_{SS}(\mathbf{w}) + \mathcal{L}_{IS}(\mathbf{w}), \qquad (1)$$

where $\mathcal{L}_{MIL}$, $\mathcal{L}_{SS}$, $\mathcal{L}_{IS}$ are respectively losses for the MIL, SS and IS branches. However, direct optimization of Eq. (1) is difficult due to the limited GPU memory. Instead, we adopt the iterative optimization strategy and alternate optimization among branches. Once the model is learned, given a test image, the inference only needs to take a forward pass through the IS branch to generate results. Neither other branches nor additional proposals are required.

Proposed network architecture

This section describes the details of the three branches.

Multiple instance learning branch. Multiple instance learning (MIL) is widely used in weakly supervised object localization methods (Bilen and Vedaldi 2016; Wan et al. 2018). In these methods, an image is often regarded as a *bag*, and the bounding boxes inside it are taken as *instances*. For each image, only the box that most likely belongs to a specific class is retained to train the classifier for this class. Since there are often multiple instances in an image, for our task, it is necessary to retain a box for an instance and multiple boxes in an image, instead of a box per image. For that purpose, following the observation in (Zhou et al. 2018) that an object instance often covers a peak in the classifier response map, we take a peak as a *bag* and regard all MCG box proposals containing only this peak as its corresponding *instances*. Each bag is assigned the same class label as the peak. With the bags and their associated labels, we employ the weakly supervised deep detection network (WSDDN) (Bilen and Vedaldi 2016) in our MIL branch for its effectiveness. Note that any MIL method can be used as the MIL branch as long as it is end-to-end trainable.

As shown in Figure 2, for each image I_n and its box proposals $\mathbb{P}_n$, the proposal features are first extracted with ROIAlign (He et al. 2017) from the feature maps of the shared convolutional layers. Then, given R proposal features inside the bag B and its associated bag label $\mathbf{L}_B$, we fed them into two streams, classification and detection. The former maps the proposal features to a matrix $\mathbf{x}^{cls} \in \mathbb{R}^{C\times R}$ by fully connected layers, where C is the number of image classes. The latter maps the proposal features to $\mathbf{x}^{det} \in \mathbb{R}^{C\times R}$. The two matrices are then passed through two individual *softmax* operators along with different dimensions,

$$\left[\sigma_{cls}\left(\mathbf{x}^{cls}\right)\right]_{ij} = \frac{e^{x_{ij}^{cls}}}{\sum_{k=1}^{C} e_{kj}^{x^{cls}}}; \left[\sigma_{det}\left(\mathbf{x}^{det}\right)\right]_{ij} = \frac{e^{x_{ij}^{det}}}{\sum_{k=1}^{R} e_{ik}^{x^{det}}}.$$

By *softmax* along columns (among classes), σ_{cls} can be viewed as performing classification for each box as it finds the most likely class for each box. Similarly, by *softmax* along rows, σ_{det} can be viewed as performing detection.

The proposal scores $\mathbf{x}^B$ for the bag B can then be produced by the element-wise product $\mathbf{x}^B = \sigma_{cls}\left(\mathbf{x}^{cls}\right) \odot \sigma_{det}\left(\mathbf{x}^{det}\right)$. The class score ϕ_B of the bag B is then obtained by summing over all proposals of B. With ϕ_B and $\mathbf{L}_B$, the standard multi-class cross entropy loss can be used to measure the loss for the bag B,

$$\mathcal{L}(B) = -\sum_{c=1}^{C} \mathbf{L}_B(c) \log \phi_B(c) + (1\text{–}\mathbf{L}_B(c)) \log(1\text{–}\phi_B(c)).$$

The sum of all bag losses serves as the loss $\mathcal{L}_{MIL}$ for training the MIL branch.

Semantic segmentation branch. A fully convolutional network (Long, Shelhamer, and Darrell 2015) is adopted for the semantic segmentation branch with a pyramid pooling module (Zhao et al. 2017) which extracts multiple-scale representations among different sub-regions. This module fuses features at different pyramid scales. At the coarsest level, global pooling is performed to generate a single bin output. At the following levels, the feature map is separated into several sub-regions and forms pooled representations for different locations. The formed feature maps at different scales have different sizes. All features at different pyramid levels are upsampled to the original size by the bilinear interpolation, and then concatenated as the final pyramid pooling feature. The segmentation score map is generated by a 1×1 convolution layer with the pyramid pooling feature. The multi-class cross entropy loss between the score maps and the pseudo semantic segmentation labels is adopted as $\mathcal{L}_{SS}$. The pseudo labels are generated with a weakly-supervised semantic segmentation method (Ahn and Kwak 2018) for the first iteration and obtained from the outputs of the instance segmentation branch for the succeeding iterations.

Instance segmentation branch. We adopt Mask R-CNN (He et al. 2017), the state-of-the-art supervised instance segmentation network, as the IS branch. Mask R-CNN extends Faster R-CNN (Ren et al. 2015), which is designed for object detection. The output of Mask R-CNN contains the bounding box, the instance mask and the corresponding class label. We follow Mask R-CNN (He et al. 2017) and define the instance segmentation loss as $\mathcal{L}_{IS}(\mathbf{w}) = \mathcal{L}_{cls}(\mathbf{w}) + \mathcal{L}_{box}(\mathbf{w}) + \mathcal{L}_{mask}(\mathbf{w})$ in Eq. (1), where $\mathcal{L}_{cls}(\mathbf{w})$ assesses accuracy of the classification task; $\mathcal{L}_{box}(\mathbf{w})$ measures goodness of the box coordinate regression; and the last term $\mathcal{L}_{mask}(\mathbf{w})$ evaluates the effectiveness of the instance mask prediction. The ground truth used in $\mathcal{L}_{mask}(\mathbf{w})$ is generated by the proposed CCGR module. Thus, no manually annotated mask is required for training.

Center-context-gap refinement

The quality of pseudo instance segmentation labels is crucial to the performance. A pseudo instance label is generated by fusing the segmentation score map and a bounding box, where the former delineates the specific class regions and the latter indicates the object location. The bounding box is initially selected by the MIL branch. For improving the quality of the bounding box and consequently enhancing the generated pseudo labels, we propose the center-context-gap refinement module (CCGR) for the box refinement.

An ideal bounding box for an instance would exactly cover the whole instance, and it would have two properties. First, the ratio of instance pixels in the box should be high. Thus, one could maximize the average segmentation score within the box, but doing this prefers small boxes containing few pixels with the highest segmentation scores. To avoid it, the second property encourages that the box surrounding should contain as fewer instance pixels as possible. Thus, minimizing the average segmentation score of the surrounding would hold this property. By combining the two properties together, we maximize the gap between the average segmentation scores of the box (center) and its surrounding (context). As shown in Figure 1, the bounding box selected by the MIL branch usually contains only the discriminative object region. By maximizing the center-context gap, the refined box covers the whole instance much better.

The inputs to CCGR are the selected bounding box $\mathcal{B}$ and the segmentation score map $\mathcal{S}$. Assume that the upper-left and lower-right coordinates of $\mathcal{B}$ are (x_1, y_1) and (x_2, y_2) respectively. The surrounding box $\tilde{\mathcal{B}}$ is the box having the same center as $\mathcal{B}$ but a larger size with 1.5 times $\mathcal{B}$'s width and height. Let $(\tilde{x}_1, \tilde{y}_1)$ and $(\tilde{x}_2, \tilde{y}_2)$ denote the upper-left and lower-right coordinates of $\tilde{\mathcal{B}}$, as shown in Figure 3. We define the context region $\mathcal{C}$ as the surrounding of $\mathcal{B}$ within $\tilde{\mathcal{B}}$, i.e., $\mathcal{C} = \tilde{\mathcal{B}} - \mathcal{B}$. Assume that the areas of $\mathcal{B}$ and $\tilde{\mathcal{B}}$ are respectively $A_{\mathcal{B}}$ and $A_{\tilde{\mathcal{B}}}$. The area of the context $\mathcal{C}$ is thus $A_{\mathcal{C}} = A_{\tilde{\mathcal{B}}} - A_{\mathcal{B}}$. CCGR adjusts the selected bounding box by maximizing the gap between the average segmentation scores of $\mathcal{B}$ and $\mathcal{C}$. Given $\mathcal{S}$, the average segmentation score $\Phi(\mathcal{B})$ of the box $\mathcal{B}$ is calculated as

$$\Phi(\mathcal{B}) = \frac{\int_{y_1}^{y_2} \int_{x_1}^{x_2} \mathcal{S}(x, y)\, dx\, dy}{A_{\mathcal{B}}}. \tag{2}$$

The average segmentation score of $\mathcal{C}$ can then be calculated as $\Phi(\mathcal{C}) = \Phi(\tilde{\mathcal{B}}) - \Phi(\mathcal{B})$. The CCGR loss for a box is defined as the gap between $\Phi(\mathcal{B})$ and $\Phi(\mathcal{C})$, i.e.,

$$\mathcal{L}_{CCGR} = \Phi(\mathcal{C}) - \Phi(\mathcal{B}). \tag{3}$$

When minimizing Eq. (3), high segmentation score in the context region is penalized by the first term, and the high segmentation score inside the bounding box is encouraged by the second term. The CCGR loss in Eq. (3) is differentiable, so the bounding box coordinates can be optimized with gradient descent. After CCGR, we directly crop the segmentation map with the refined boxes as the pseudo instance labels for training the IS branch.

Figure 3 shows an example of box refinement. In the first row, it is obvious that although the pixels inside the selected bounding box have high segmentation scores, the context region also contain pixels with high scores. After refinement, in the second row, almost all pixels in the context region have low segmentation scores, and the refined box covers the object more tightly.

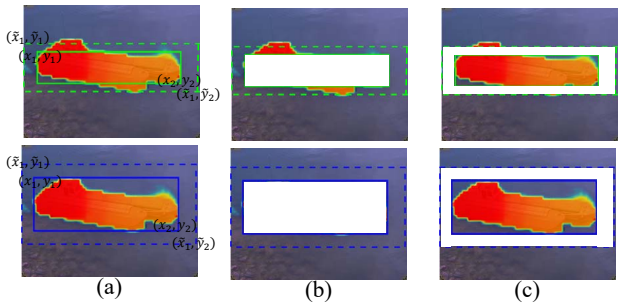

Figure 3: The example of CCGR visualization. The first row shows the boxes before refinement. The solid rectangles represent the bounding boxes while the dotted ones represent the context. The second row shows the boxes after refinement. (a) segmentation score maps. (b) segmentation scores inside the context region. (c) segmentation scores inside the bounding box.

Iterative optimization and its initialization

Due to limited memory, it is infeasible to train the whole network in Figure 2. Thus, we adopt an iterative strategy for the optimization. In each iteration, the MIL, SS, and IS branches are sequentially trained. More specifically, we first train the MIL branch with the given bags. Next, the SS branch is trained with the provided pseudo class-level segmentation labels. With the boxes from the MIL branch and the segmentation score maps generated by the SS branch, CCGR refines all bounding boxes. With the refined boxes, we crop the segmentation map using the boxes for generating pseudo instance-level labels that are fed to the IS branch for training. After each iteration, the bags and the class-level pseudo segmentation labels are updated according to the instance segmentation results of Mask R-CNN. We found empirically that $3 \sim 5$ iterations are sufficient for training.

The peak in the response maps of the classifiers (Zhou et al. 2018) can indicate a potential object instance. Therefore, for the MIL branch, a peak is regarded as a bag, and its class label is assigned to the bag. Then, we generate the MCG box proposals (Pont-Tuset et al. 2017), and collect the first 10 top-ranked proposals containing the peak. A box proposal $\mathcal{B}$ is ranked by the following ranking function,

$$\mathcal{R}\left(\mathcal{B}\right) = \mathcal{B}_S - 0.1 \times \frac{\mathcal{B} * (1 - O)}{|\mathcal{B}|}, \tag{4}$$

where $\mathcal{B}_S$ is $\mathcal{B}$'s objectness score from (Pont-Tuset et al. 2017), O is the segmentation score map from (Ahn and Kwak 2018), and $*$ is the Frobenius inner product between two matrices. The first term encourages proposals with high objectness scores while the second term penalizes the proposals containing more background. For the later iterations, the bounding boxes from the IS branch are used as the bags. For a predicted box, we first compute the box overlap ratio between it and all MCG box proposals, and select the top 10 proposals with the highest overlap ratios as the corresponding instances for the bag.

Training the SS branch requires semantic segmentation labels. For the first iteration, we use the results generated by a weakly-supervised semantic segmentation method (Ahn and Kwak 2018) as the class-level pseudo labels. For the successive iterations, the unions of all instance masks of the same class from the IS branch serve as the pseudo labels.

method	year	Sup.	MCG	$\text{mAP}^r_{0.25}$	$\text{mAP}^r_{0.5}$
FRCNN (Ren et al. 2015)	NeurIPS 2015	FS	×	70.7	62.1
FRCNN (Ren et al. 2015)	NeurIPS 2015	FS*	×	47.8	25.0
MELM (Wang et al. 2018)	CVPR 2018	I	√	36.9	22.9
SPN (Zhou et al. 2016)	ICCV 2017	I	√	26.4	12.7
CAM (Zhou et al. 2016)	CVPR 2016	I	√	20.4	7.8
PRM (Zhou et al. 2018)	CVPR 2018	I	√	44.3	26.8
IAM (Zhu et al. 2019)	CVPR 2019	I	×	45.9	28.8
ILC (Cholakkal et al. 2019)	CVPR 2019	I*	√	48.5	30.2
Ours	-	I	×	53.6	35.3

Table 1: Performance of instance segmentation on the PASCAL VOC 2012 dataset. The numbers in red and green indicate the best and the second best results, respectively. Sup. indicates the supervision type, while I, I*, FS, FS* indicate the image-level label, the image-level label and object counts, the fully supervised setting, and the fully supervised setting with the same annotation cost of the image-level label, respectively.

Implementation details

We implement the proposed method using *PyTorch*. ResNet-50 (He et al. 2016) is adopted as the backbone network. For the MIL branch and the SS branch, the batch size is set to 1 and a single GPU is sufficient for training. The learning rate, weight decay, and momentum are set to 10^{-4}, 0.0001, and 0.9, respectively. For the instance segmentation branch, the batch size is 16 and learning rate is set to 10^{-2}. Four GPUs are used for its training. The optimization procedures of the MIL branch, the SS branch and the IS branch stop after $55k$, $80k$, $24k$ epochs respectively. We choose ADAM (Kingma and Ba 2014) as the optimization solver.

Experimental Results

Our method is evaluated in this section. We first describe the datasets and evaluation metrics, then compare our method with the state-of-the-art methods, and finally conduct ablation studies to assess the effect of each component.

Datasets and evaluation metrics

PASCAL VOC 2012 **dataset.** This dataset (Everingham et al. 2010) contains 20 object categories. It has $5,717$ training images and $5,823$ validation images for object classification and detection, while $1,464$ training images and $1,449$ validation images are used for segmentation. Following (Zhou et al. 2018), we use the $5,717$ training images of the classification task as training data, and evaluate our method on the validation images for the segmentation task.

COCO-VOC dataset. This dataset is collected in (Hsu, Lin, and Chuang 2019) based on the MS COCO (Lin et al. 2014) dataset. It contains 12 object categories commonly covered by PASCAL VOC 2012. It has $1,281$ images with total $3,151$ instances. We only use this dataset for inference.

method	year	input type	MCG	$mAP^r_{0.25}$	$mAP^r_{0.5}$
CLRW (Tang et al. 2014)	CVPR 2014	Multiple	√	33.3	13.7
UODL (Cho et al. 2015)	CVPR 2015	Multiple	√	9.6	2.2
DDT (Wei et al. 2017)	IJCAI 2017	Multiple	√	31.4	10.1
DFF (Collins, Achanta, and Susstrunk 2018)	ECCV 2018	Multiple	√	31.7	10.6
DDT+ (Wei et al. 2019)	PR 2019	Multiple	√	30.8	11.6
PRM (Zhou et al. 2018)	CVPR 2018	Single	√	44.9	14.6
Ours	-	Single	×	60.1	26.0

Table 2: Performance of instance segmentation on the COCO-VOC dataset. The numbers in red and green indicate the best and the second best results, respectively.

Figure 4: Progressive box refinement by CCGR on five examples (rows). From top to bottom, the object classes are *dining table*, *bike*, *bus*, *sofa* and *cat* respectively. Blue boxes are the ground truth. Green rectangles are (a) the boxes predicted by the MIL branch and (b) $\sim$ (f) the boxes refined by CCGR at the 40th, 80th, 120th, 160th, and 200th optimization iterations, respectively.

Evaluation mthoetrics. The mean average precision with an IoU (intersection over union) threshold r (mAP^r) (Hariharan et al. 2014) is used as the performance measure for instance segmentation. Following (Zhou et al. 2018; Hsu, Lin, and Chuang 2019), mAP^r with $r \in \{0.25, 0.5\}$ is computed in this work. Besides, for the evaluation of the predicted bounding box, we use the same metrics but replace the segments with the bounding boxes. We only use the threshold, 0.5, and denote it by $mAP^b_{0.5}$.

Competing methods

We select the most representative and the state-of-the-art methods as the competing methods. In the following, we describe them based on the datasets.

Methods on the PASCAL VOC 2012 **dataset.** Our approach is compared with six weakly supervised state-of-the-art methods, including three instance segmentation methods, ILC (Cholakkal et al. 2019), IAM (Zhu et al. 2019), PRM (Zhou et al. 2018), and three object localization methods, MELM (Wang et al. 2018), CAM (Zhou et al. 2016), and SPN (Zhu et al. 2017). Image-level labels are used as training data in all methods except for ILC using additional object counts. Following PRM, the MCG (Pont-Tuset et al. 2017) proposals are employed to convert the predicted boxes

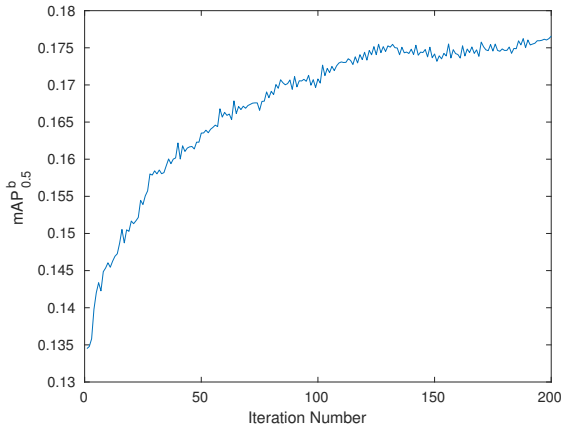

Figure 5: Performance in $mAP^b_{0.5}$ along the CCGR iterative optimization process.

in MELM, CAM, and SPN into segments. For each predicted bounding box, the proposal with the highest IoU is considered as the results of instance segmentation.

Methods on the COCO-VOC dataset. Two types of methods are compared on this dataset, as reported in (Hsu, Lin, and Chuang 2019). The first type is object co-localization, and we adopt CLRW (Tang et al. 2014), UODL (Cho et al. 2015), DDT (Wei et al. 2017), DDT+ (Wei et al. 2019), and DFF (Collins, Achanta, and Susstrunk 2018) as our competing methods. Similarly, the MCG proposals are used to convert the box-level results into segment-level ones. The second type is the weakly supervised instance segmentation, PRM (Zhou et al. 2018). Although the object co-localization method requires the multiple images containing the same object categories, we still include these competing methods as our baselines to enrich the comparison.

Comparison with the state-of-the-art methods

In Table 1 and Table 2, we report the results of our method and all competing methods on the PASCAL VOC 2012 and COCO-VOC datasets, respectively. As shown in Table 1, our method outperforms the state-of-the-art weakly localization method, MELM (Wang et al. 2018) by a large margin 12.4% in $mAP^r_{0.5}$ and the state-of-the-art weakly supervised instance segmentation method, ICL (Cholakkal et al. 2019) by a large margin 5.1% in $mAP^r_{0.5}$. However, ILC requires the additional object counts for training, and thus has more annotation cost. Besides, all competing methods except for IAM in Table 1 require the MCG proposals, so their results depend on the quality of object proposals. However, we don't require any proposals for inference, so our method can run faster and don't depend on any external proposal results. In Table 2, the object co-localization methods belong to the online optimization methods, and require the MCG proposals, so our method can run faster than these methods and has higher performance.

Annotation cost analysis. In Table 1, we also compare to our upper bound, Mask R-CNN, training on the pixel-level masks. Besides, we also compare the fully-supervised Mask R-CNN trained on the pixel-level mask with the same annotation cost of the image-level labels, as discussed by (Bear-

Seg.	Det.	CCGR	Ins.	$mAP^r_{0.25}$	$mAP^r_{0.5}$
√	×	×	√	38.3	19.3
√	√	×	√	44.0	25.0
√	√*	√	√	39.4	19.8
√	√	√	√	49.1	31.2

Table 3: Ablation studies of our proposed methods. * indicates that the proposals is generated by the pretrained RPN.

man et al. 2016). Our upper bound is 62.1 in $mAP^b_{0.5}$, but our method outperforms Mask R-CNN with the same annotation cost by a large margin about 10.3% in $mAP^b_{0.5}$ due to more training images.

Ablation studies

We conduct the convergence analysis of CCGR. Figure 5 plots the performance in $mAP^b_{0.5}$ along the CCGR iterative optimization process on the training images of the PASCAL VOC 2012 classification task. We observe that CCGR converges smoothly, and the performance is gradually improved along with optimization. To gain insight into the quantitative analysis, Figure 4 displays the bounding boxes gradually refined by CCGR. It is observed that the output boxes can be greatly improved by CCGR even under unfavorable conditions, such as clutter background (*dinning table* and *sofa*), large intra-object variations (*bus*), complex object shapes (*bike*), and low figure-ground distinctness (*cat*).

In Table 3, we also show performance without the detection branch and the CCGR. It is noted that we only show the results of the first training iteration. Without the detection branch, we simply generate the bounding box with four corners of the segmentation mask. This result is 19.3 in $mAP^b_{0.5}$, and falls behind our method by a margin 11.9% since the masks may contain multiple instances of the same category. On the other hand, we stop the CCGR process if the refined bounding box contains more than one peak, and thus the proposed CCGR can prevent this situation. The result without GGCR is 25.0 in $mAP^b_{0.5}$, and it is also lower than our method. It can prove that CCGR can produce the pseudo ground truth with higher quality. Besides, we also conduct the results with the RPN pre-trained on the external dataset, and thus the external proposal algorithm, MCG, is not adopted. In Table 3, the result using RPN pre-trained on COCO dataset without fine-tuning is 19.8 in $mAP^b_{0.5}$, which is lower than our method. Compared to the pre-trained RPN, the end-to-end learning manner can learn more reliable RPN parameters, and thus the better result can be produced.

Qualitative results

We show several examples of the segmentation results in Figure 6. Even with challenging variations, such as closeness, occlusions, and multiple scales of instances, our method can still deal with these variations well. In Figure 6 (a) and (b), we present multiple instances belonging to the same category, i.e., cow, and our method can produce the promising results. Moreover, in Figure 6 (c) and (d), multiple instances of different categories, i.e., train and person,

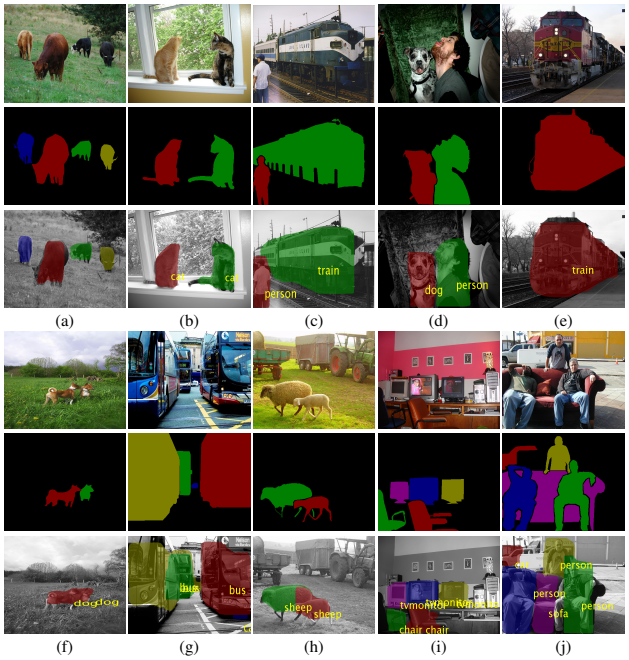

Figure 6: In the first and last three rows, the successful and failure results are shown. For each case, from the first row to the third row, the input images, the ground truth masks, and the predicted results are shown, respectively.

are shown. The occlusion between instances also appears in this image. Nevertheless, our method works well.

Limitation. In Figure 6, some typical failure cases are also shown. In (f), our method wrongly merges the two different instances of the same categories because of the similar pattern. In (g) and (h), our method fails to detect accurate boundaries of objects because of the instance closeness. In (i) and (j), false positives and false negatives appear since noisy patterns in clutter background are presented.

Conclusion

This paper proposes a weakly supervised instance segmentation method requiring only image-level class annotations. To address this task, we introduce a new deep architecture containing three branches, including the MIL branch, the SS branch, and the IS branch. The bounding boxes containing instances are selected in the MIL branch, and the segmentation results offering the shapes of instances are provided in the SS branch. By combining two types of results, we produce pseudo instance masks as the training data to train the IS branch. To further improve the quality of the generated pseudo labels, we propose CCGR, which significantly enhance the performance. In the experiments, the proposed method outperforms the state-of-the-art weakly supervised methods. In the future, we will integrate the proposed architecture and CCGR into the more difficult task, panoptic segmentation, which unifies instance segmentation for thing classes and semantic segmentation for stuff classes.

References

Ahn, J., and Kwak, S. 2018. Learning pixel-level semantic affinity with image-level supervision for weakly supervised semantic segmentation. In *CVPR*.

Bearman, A.; Russakovsky, O.; Ferrari, V.; and Fei-Fei, L. 2016. Whats the point: Semantic segmentation with point supervision. In *ECCV*.

Bilen, H., and Vedaldi, A. 2016. Weakly supervised deep detection networks. In *CVPR*.

Chen, L.-C.; Zhu, Y.; Papandreou, G.; Schroff, F.; and Adam, H. 2018. Encoder-decoder with atrous separable convolution for semantic image segmentation. In *ECCV*.

Cho, M.; Kwak, S.; Schmid, C.; and Ponce, J. 2015. Unsupervised object discovery and localization in the wild: Part-based matching with bottom-up region proposals. In *CVPR*.

Cholakkal, H.; Sun1, G.; Khan, F. S.; and Shao, L. 2019. Object counting and instance segmentation with image-level supervision. In *CVPR*.

Collins, E.; Achanta, R.; and Susstrunk, S. 2018. Deep feature factorization for concept discovery. In *ECCV*.

Everingham, M.; Gool, L. V.; Williams, C. K. I.; Winn, J.; and Zisserman, A. 2010. The pascal visual object classes (VOC) challenge. *IJCV*.

Gan, C.; Li, Y.; Li, H.; Sun, C.; and Gong, B. 2017. VQS: Linking segmentations to questions and answers for supervised attention in vqa and question-focused semantic segmentation. In *ICCV*.

Hariharan, B.; Arbelaez, P.; Girshick, R.; and Malik., J. 2014. Simultaneous detection and segmentation. In *ECCV*.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep residual learning for image recognition. In *CVPR*.

He, K.; Gkioxari, G.; Dollar, P.; and Girshick, R. 2017. Mask R-CNN. In *ICCV*.

Hou, Q.; Jiang, P.-T.; Wei, Y.; and M.-M.-Cheng. 2018. Self-erasing network for integral object attention. In *NeurIPS*.

Hsu, K.-J.; Lin, Y.-Y.; and Chuang, Y.-Y. 2019. Deep instance co-segmentation by co-peak search and co-saliency detectio. In *CVPR*.

Huang, Y.; Wang, W.; and Wang, L. 2017. Instance-aware image and sentence matching with selective multimodal LSTM. In *CVPR*.

Khoreva, A.; Benenson, R.; Hosang, J.; Hein, M.; and Schiele, B. 2017. Simple does it: Weakly supervised instance and semantic segmentation. In *CVPR*.

Kingma, D., and Ba, J. 2014. ADAM: A method for stochastic optimization. In *ICLR*.

Krizhevsky, A.; Sutskever, I.; and Hinton, G. 2012. Imagenet classification with deep convolutional neural networks. In *NeurIPS*.

Lin, T.-Y.; Maire, M.; Belongie, S.; Hays, J.; Perona, P.; Ramanan, D.; Dollar, P.; and Zitnick, C. L. 2014. Microsoft COCO: Common objects in context. In *ECCV*.

Lin, G.; Liu, F.; Milan, A.; Shen, C.; and Reid, I. 2019. Refinenet: Multi-path refinement networks for dense prediction. *TPAMI*.

Liu, Y.; Yang, S.; Li, B.; Zhou, W.; Xu, J.; Li, H.; and Lu, Y. 2018. Affinity derivation and graph merge for instance segmentation. In *ECCV*.

Long, J.; Shelhamer, E.; and Darrell, T. 2015. Fully convolutional models for semantic segmentation. In *CVPR*.

Novotny, D.; Novotny, D.; Larlus, D.; and Vedaldi, A. 2018. Semi-convolutional operators for instance segmentation. In *ECCV*.

Pont-Tuset, J.; Arbelaez, P.; Barron, J.; Marques, F.; and Malik, J. 2017. Multiscale combinatorial grouping for image segmentation and object proposal generation. *TPAMI*.

Qian, R.; Wei, Y.; Shi, H.; Li, J.; Liu, J.; and Huang, T. 2019. Weakly supervised scene parsing with point-based distance metric learning. In *AAAI*.

Ren, S.; He, K.; Girshick, R.; and Sun, J. 2015. Faster R-CNN: Towards real-time object detection with region proposal networks. In *NeurIPS*.

Shen, T.; Lin, G.; Shen, C.; and Reid, I. 2018. Bootstrapping the performance of webly supervised semantic segmentation. In *CVPR*.

Tang, K.; Joulin, A.; Li, L.-J.; and Li, F.-F. 2014. Co-localization in real-world images. In *CVPR*.

Tang, M.; Perazzi, F.; Djelouah, A.; Ayed, I.; Schroers, C.; and Boykov, Y. 2018. On regularized losses for weakly-supervised cnn segmentation. In *ECCV*.

Wan, F.; Wei, P.; Jiao, J.; Han, Z.; ; and Ye, Q. 2018. Min-entropy latent model for weakly supervised object detection. In *CVPR*.

Wang, X.; You, S.; Li, X.; and Ma, H. 2018. Weakly-supervised semantic segmentation by iteratively mining common object features. In *CVPR*.

Wei, X.-S.; Zhang, C.-L.; Li, Y.; Xie, C.-W.; Wu, J.; Shen, C.; and Zhou, Z.-H. 2017. Deep descriptor transforming for image co-localization. In *IJCAI*.

Wei, Y.; Xiao, H.; Shi, H.; Jie, Z.; Feng, J.; and Huang, T. 2018. Revisiting dilated convolution: A simple approach for weakly- and semi-supervised semantic segmentation. In *CVPR*.

Wei, X.-S.; Zhang, C.-L.; Wu, J.; Shen, C.; and Zhou, Z.-H. 2019. Unsupervised object discovery and co-localization by deep descriptor transforming. *PR*.

Zhang, Z.; Zhang, X.; Peng, C.; Xue, X.; and Sun, J. 2018. ExFuse: Enhancing feature fusion for semantic segmentation. In *ECCV*.

Zhang, Z.; Fidler, S.; and Urtasun, S. 2016. Instance-level segmentation for autonomous driving with deep densely connected mrfs. In *CVPR*.

Zhao, H.; Shi, J.; Qi, X.; Wang, X.; and Jia, J. 2017. Pyramid scene parsing network. In *CVPR*.

Zhou, B.; Khosla, A.; Lapedriza, A.; Oliva, A.; and Torralba, A. 2016. Learning deep features for discriminative localization. In *CVPR*.

Zhou, Y.; Zhu, Y.; Ye, Q.; Qiu, Q.; and Jiao, J. 2018. Weakly supervised instance segmentation using class peak response. In *CVPR*.

Zhu, Y.; Zhou, Y.; Ye, Q.; Qiu, Q.; and Jiao, J. 2017. Soft proposal networks for weakly supervised object localization. In *ICCV*.

Zhu, Y.; Zhou, Y.; Xu, H.; Ye, Q.; Doermann, D.; and Jiao, J. 2019. Learning instance activation maps for weakly supervised instance segmentation. In *CVPR*.