Affinity Attention Graph Neural Network for Weakly Supervised Semantic Segmentation

Bingfeng Zhang, Jimin Xiao, *Member, IEEE,* Jianbo Jiao, *Member, IEEE,* Yunchao Wei, *Member, IEEE,* Yao Zhao, *Senior Member, IEEE*

Abstract—Weakly supervised semantic segmentation is receiving great attention due to its low human annotation cost. In this paper, we aim to tackle bounding box supervised semantic segmentation, *i.e.*, training accurate semantic segmentation models using bounding box annotations as supervision. To this end, we propose Affinity Attention Graph Neural Network (A^2*GNN*). Following previous practices, we first generate pseudo semantic-aware seeds, which are then formed into semantic graphs based on our newly proposed affinity Convolutional Neural Network (CNN). Then the built graphs are input to our A^2*GNN*, in which an affinity attention layer is designed to acquire the short- and long- distance information from soft graph edges to accurately propagate semantic labels from the confident seeds to the unlabeled pixels. However, to guarantee the precision of the seeds, we only adopt a limited number of confident pixel seed labels for A^2*GNN*, which may lead to insufficient supervision for training. To alleviate this issue, we further introduce a new loss function and a consistency-checking mechanism to leverage the bounding box constraint, so that more reliable guidance can be included for the model optimization. Experiments show that our approach achieves new state-of-the-art performances on Pascal VOC 2012 datasets (*val*: 76.5%, *test*: 75.2%). More importantly, our approach can be readily applied to bounding box supervised instance segmentation task or other weakly supervised semantic segmentation tasks, with state-of-the-art or comparable performance among almot all weakly supervised tasks on PASCAL VOC or COCO dataset. Our source code will be available at https://github.com/zbf1991/A2GNN.

Index Terms—Weakly supervised, semantic segmentation, graph neural network.

1 Introduction

Weakly supervised semantic segmentation aims to make a pixel-level semantic prediction using weak annotations as supervision. According to the level of provided annotations, the weak supervision can be divided into scribble level [1], [2], [3], bounding box level [4], [5], [6], [7], point level [8] and image level [9], [10], [11], [12]. In this paper, we mainly focus on bounding box supervised semantic segmentation (BSSS). The key challenge of BSSS lies in how to accurately estimate the pseudo object mask within the given bounding box so that reliable segmentation networks can be learned with the generated pseudo masks using current popular fully convolutional networks (FCN) [13], [14], [15], [16].

Most previous practices [4], [5], [6], [17] for the BSSS task use object proposals [18], [19] to provide some seed labels as supervision. These methods follow a common pipeline of employing object proposals [18], [19] and CRF [20] to produce pseudo masks, which are then adopted as ground-truth to train the segmentation network. However, such a pipeline often fails to generate accurate pseudo labels due to the gap between segmentation masks and object proposals. To overcome this limitation, graph-based learning was subsequently proposed to use the confident but a limited number of pixels mined from proposals as supervision. Compared to previous approaches, graph-based learning especially Graph Neural Network (GNN) can directly build long-distance edges between different nodes and aggregate information from multiple connected nodes, enabling to suppress the negative impact of the label noise. Besides, GNN performs well in semi-supervised tasks even with limited labels.

Recently, *GraphNet* [21] attempts to use Graph Convolutional Network (GCN) [22] for the BSSS task. They convert images to unweighted graphs by grouping pixels in a superpixel to a graph node [23]. Then the graph is input to a standard GCN with cross entropy loss to generate pseudo labels. However, there are two main drawbacks which limit its performance: (1) *GraphNet* [21] builds an unweighted graph as input, however, such a graph cannot accurately provide sufficient information since it treats all edges equally, with the edge weight being either 0 or 1, though in practice not all connected nodes expect the same affinity. (2) Using *GraphNet* [21] will lead to incorrect feature aggregation as input nodes and edges are not 100% accurate. For example, for an image that contains both dogs and cats, the initial node feature of dog fur and cat fur might be highly similar, which will produce some connected edges between them as edges are built based on feature similarity. Such edges will lead to a false positive case since *GraphNet* [21] only considers the initial edges for feature propagation. Thus, if the strong

B. Zhang is with University of Liverpool, UK, and also with School of Advanced Technology, Xi'an Jiaotong-Liverpool University, Suzhou, P.R. China (e-mail: bingfeng.zhang@liverpool.ac.uk).
J. Xiao is with School of Advanced Technology, Xi'an Jiaotong-Liverpool University, Suzhou, P.R. China (e-mail: jimin.xiao@xjtlu.edu.cn). (Corresponding author: Jimin Xiao).
J. Jiao is with the Department of Engineering Science, University of Oxford, UK (e-mail: jianbo.jiao@eng.ox.ac.uk).
Y. Wei is with University of Technology Sydney, Australia (e-mail: yunchao.wei@uts.edu.au).
Y. Zhao is with Institute of Information Science, Beijing Jiaotong University, Beijing, China (e-mail: yzhao@bjtu.edu.cn).

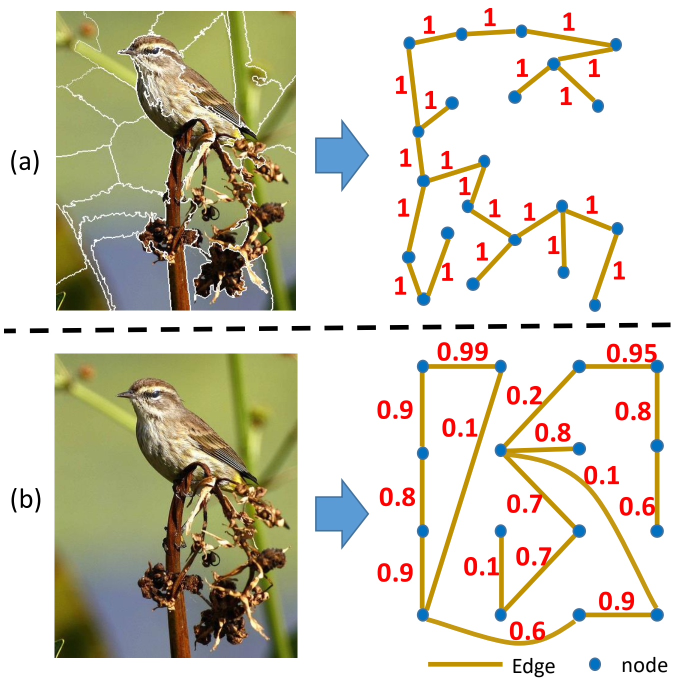


Fig. 1. The difference between our built graph and that of previous approach [21]. (a) Superpixel based approach [21]. (b) Our approach. The numbers along the edges indicate the edge values, soft edge allows any edge weights between 0 and 1.

correlations among pixels from different semantics can be effectively alleviated, a better propagation model can be acquired to generate more accurate pseudo object masks.

To this end, we design an Affinity Attention Graph Neural Network (A^2GNN) to address the above mentioned issues. Specifically, instead of using traditional method to build a unweighted graph, we propose a new affinity Convolutional Neural Network (CNN) to convert an image to a weighted graph. We consider that a weighted graph is more suitable than an unweighted one as it can provide different affinities for different node pairs. Fig. 1 shows the difference between our built graph and that of the previous approach [21]. It can be seen that the previous approach only considers locally connected nodes, and they build an unweighted graph based on superpixel [23], while we consider both local and long distance edges, and the built weighted graph views one pixel as one node.

Second, in order to produce accurate pseudo labels, we design a new GNN layer, in which both the attention mechanism and the edge weights are applied in order to ensure accurate propagation. So feature aggregation between pairwise nodes with weak/no edge connection or low attention can be significantly declined, and thus eliminating incorrect propagation accordingly. The node attention dynamically changes as training goes on.

However, to guarantee the accuracy of supervision, we only choose a limited number of confident seed labels as supervision, which is insufficient for the network optimization. For example, only around 40% foreground pixels are labeled in one image and none of them is 100% reliable. To further tackle this issue, we introduce a multi-point (MP) loss to augment the training of A^2GNN. Our MP loss adopts an online update mechanism to provide extra supervision from bounding box information. Moreover, in order to strengthen feature propagation of our A^2GNN, MP loss attempts to close up the feature distance of the same semantic objects, making the pixels of the same object distinguishable from others. Finally, considering that the selected seed labels may not perfectly reliable, we introduce a consistency-checking mechanism to remove those noisy labels from the selected seed labels, by comparing them with the labels used in the MP loss.

To validate the effectiveness of our A^2GNN, we perform extensive experiments on PASCAL VOC. In particular, we achieve a new mIoU score of 76.5% on the validation set. In addition, our A^2GNN can be further smoothly transferred to conduct the bounding box supervised instance segmentation (BSIS) task or other weakly supervised semantic segmentation tasks. According to our experiments, we achieve new state-of-the-art or comparable performances among all these tasks.

Our main contributions are summarized as:

- We propose a new framework that effectively combines the advantage of CNN and GNN for weakly supervised semantic segmentation. To the best of our knowledge, this is the first framework that can be readily applied to all existing weakly supervised semantic segmentation settings and the bounding box supervised instance segmentation setting.
- We design a new affinity CNN network to convert a given image to an irregular graph, where the graph node features and the node edges are generated simultaneously. Compared to existing approaches, the graphs built from our method are more accurate for various weakly supervised semantic segmentation settings.
- We propose a new GNN, A^2GNN, where we design a new GNN layer that can effectively mitigate inaccurate feature propagation through information aggregation based on edge weights and node attention. We further propose a new loss function (MP loss) to mine extra reliable labels using the bounding box constraint and remove existing label noise by consistency-checking.
- Our approach achieves state-of-the-art performance for BSSS on PASCAL VOC 2012 (*val*: 76.5%, *test*: 75.2%) as well as BSIS on PASCAL VOC 2012 ($\text{mAP}^r_{0.5}$: 59.1%, $\text{mAP}^r_{0.7}$: 35.5%, $\text{mAP}^r_{0.75}$: 27.4%) and COCO ($\text{mAP}^r_{0.5}$: 43.9%). Meanwhile, when applying the proposed approach to other weakly supervised semantic segmentation settings, new state-of-the-art or comparable performances are achieved as well.

2 Related Work

2.1 Weakly Supervised Semantic Segmentation

According to the definition of supervision signals, weakly supervised semantic segmentation can be generally divided into the following categories: based on scribble label [1], [2], [3], bounding box label [4], [5], [6], point label [8] and image-level class label [9], [10], [11]. Scribble, bounding box and point labels are stronger supervision signals compared to the image-level class label since both class and localization information are provided. Whereas image-level labels only provide image class tags with the lowest annotation cost.

Different supervisions are processed with different methods to generate pseudo labels. For the scribble supervision, Lin *et al.* [2] used superpixel based approach (*e.g.*, SLIC [23]) to expand the initial scribbles, and then used an FCN model [13] to get the final predictions. Tang *et al.* proposed two regularized losses [1], [3] using the constraint energy loss function to expand the scribble information. For the point supervision, Bearman *et al.* [8] directly incorporated a generic object prior in the loss function. For image-level supervision, class activation map (CAM) [24] is usually used as seeds to get pseudo labels. For example, Ahn and Kwak designed an affinity net [9] to obtain the transition probability matrix, and used random walk [25] to get pseudo labels. Huang *et al.* [26] proposed to use the seed region growing algorithm [27] to get pseudo labels from the initial confident class activation map. For the bounding box supervision, SDI [5] used the segmentation proposal by combining MCG [28] with GrabCut [29] to generate the pseudo labels. Song *et al.* proposed a box-driven method [6], using box-driven class-wise masking and filling rate guided adaptive loss to generate pseudo labels. Box2Seg [30] attempt to design a segmentation network which is suitable to utilize the noisy labels as supervision.

Following the same definition, there are different level sub-tasks for weakly supervised instance segmentation: image-level [12] and bounding box level [5], [7]. For the image-level task, Ahn *et al.* tried to generate the instance pseudo label using an affinity network [12]. For the bounding box task, SDI [5] produced pseudo labels using the segmentation proposal by combining MCG [28] with GrabCut [29] while Hsu *et al.* [7] tried to design a new loss function which samples positive and negative pixels relying on bounding box supervision.

2.2 Graph Neural Network

Generally, there are many different GNN [22], [31], [32], [33] methods designed for semi-supervised task and they have achieved satisfying performances. Recently, GNN has been successfully used in various computer vision tasks such as person search [34], image recognition [35], 3D pose estimation [36] and video object segmentation [37], etc.

For weakly supervised semantic segmentation, *GraphNet* [21] was proposed for bounding box task using GCN [22]. Specifically, based on the CAM technique [24], bounding box supervision was converted to initial pixel-level supervision firstly. And then the superpixel method [23] was applied to produce the graph nodes. They used a pre-trained CNN model to obtain the node feature, which was computed using average pooling for all its pixels. After that an adjacency matrix was computed based on the L1 distance between a node and its 8-neighbor nodes. Later, GCN was used to propagate the initial pixel-level labels to integral pseudo labels. Finally, all the pseudo labels were input into a semantic segmentation model for training. *GraphNet* proved that GNN was one possible solution for weakly supervised semantic segmentation. However, this method has some limitations. First of all, using superpixels as nodes introduce incorrect node labels, and building an adjacency matrix by a threshold loses some important detailed information. Secondly, the performance of this method is limited by the usage of GNN [22], which only considers the non-weight adjacency matrix. Finally, only cross entropy loss is used for *GraphNet*, which cannot mitigate the influence of incorrect nodes, edges and labels.

In order to overcome the limitations of previous graph-based learning approach, we design a new approach, A^2GNN, which takes a more accurate weighted graph as input and aggregates feature by considering both attention mechanism and edge weights. Meanwhile, we propose a new loss function to provide extra supervision and impose restrictions on the feature aggregation, thus our A^2GNN can generate high quality pseudo labels.

3 Generate Pixel-level Seed Label

The common practice to initialize weakly supervised task is to generate pixel-level seed labels from weak supervision [21], [38], [39]. For the BSSS task, both image-level and bounding box-level labels are available. We use both of them to generate the pixel-level seed labels since image-level label can generate foreground seeds while bounding box-level label can provide accurate background seeds. To convert the image-level label to pixel-level labels, we use a CAM-based method [9], [12], [24], [38]. To generate pixel-level labels from bounding box supervision, Grab-cut [29] is used to generate the initial labels, and the pixels which do not belong to any box are regarded as background labels. Finally, these two types of labels are fused together to generate the pixel-level seed labels.

Specifically, we use SEAM [38], which is a self-supervised classification network, to generate the pixel-level seed labels from image-level supervision. Suppose a dataset with category set $C = [c_0, c_1, c_2, ..., c_{N-1}]$, in which c_0 is background with the rest representing foreground categories. The pixel-level seed labels from image-level supervision are:

$$M_I = \text{Net}_{\text{SEAM}}(I), \tag{1}$$

where M_I is the generated seed labels. $\text{Net}_{\text{SEAM}}(\cdot)$ is the classification CNN used in SEAM [38].

For the BSSS task, as it provides bounding box-level label in addition to image-level label. We also generate pixel labels from the bounding box label as it can provide accurate background labels and object localization information. Given an image, suppose the bounding box set is $B = \{B_1, ..., B_M\}$. For a bounding box B_k with label L_{B_k}, its height and width are h and w, respectively. We use Grab-cut [29] to generate the seed labels from bounding box supervision, the seed labels for each bounding box are defined as:

$$M_{B_k}(i) = \begin{cases} \text{Grab}(i), & \text{if } i \in B_k \text{ and Grab}(i) \neq c_0 \\ 255, & \text{else} \end{cases}, \tag{2}$$

where $\text{Grab}(\cdot)$ is the Grab-cut operator and 255 means the pixel label is unknown.

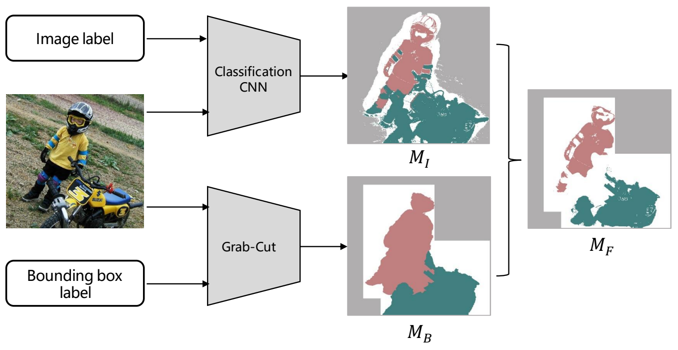


Fig. 2. An example of generating pixel-level seed labels. Given an image with its label, we firstly generate M_I from image-level label using a classification CNN and SEAM [38] method. Meanwhile, bounding box label is transferred to pixel-level label M_B using Grab-cut. Finally, M_I and M_B are integrated together to get the pixel-level seed label M_F. Each color represents one class and "white" means the pixel label is unknown.

Pixels not belonging to any bounding box are expressed as background, and the final seed labels generated from bounding box are:

$$M_B(i) = \begin{cases} c_0, & \text{if } i \notin B \\ M_{B_k}(i), & \text{if } i \in B \end{cases}. \tag{3}$$

For pixel i in the image, the final pixel-level seed label is defined as :

$$M_F(i) = \begin{cases} M_B(i), & \text{if } i \notin B \\ M_I(i), & \text{if } i \in B \text{ and } M_B(i) = M_I(i) \\ M_{B_k}(i), & \text{if } i \in B_k \text{ and } L_{B_k} \notin S(M_{I\text{-}B_k}) \\ 255, & \text{else} \end{cases}, \tag{4}$$

where $S(M_{I\text{-}B_k})$ is the set of predicted categories in M_I for bounding box B_k. $L_{B_k} \notin S(M_{I\text{-}B_k})$ indicates that there is no correct predicted label in M_I for bounding box B_k and we therefore use the prediction from M_{B_k} as the final seed labels.

In Fig. 2, an example is given to demonstrate the process to convert bounding box supervision to pixel-level seed labels. After combing M_I and M_B, we can get the pixel-level seed label.

4 The Proposed A^2GNN

4.1 Overview

In order to utilize GNN to generate the accurate pixel-level pseudo labels, there are three main problems: (1) How to provide useful supervision information and reduce the label noise as much as possible. (2) How to convert the image data to accurate graph data. (3) How to generate accurate pseudo labels based on the built graph and the supervision.

In this section, we will elaborate on the proposed A^2GNN to address the above mentioned three main problems. To generate an accurate graph, we propose a new affinity CNN to convert an image to a graph. To provide accurately labeled nodes for the graph, we select highly confident pixel-level seed labels as node labels, and at the same time, we introduce extra online updated labels based on the bounding box supervision, meanwhile, the pixel-level seed labels are further refined by consistency-checking. To generate accurate pseudo labels, we design a new GNN layer since the previous GNN, such as GCN [22] or AGNN [32] is designed based on the assumption that labels are 100% accurate, while in this case, there is no foreground pixel label being 100% reliable.

In Fig. 3, we show the main process of our approach, which can be divided into three steps:

(1) Generating confident seed labels. In this step, both image-level labels and bounding box-level labels are converted to initial pixel-level seed labels, as explained in section 3. Then the pixel-level seed labels with high confidence will be selected as confident seed labels (section 4.2).
(2) Converting images to graphs. In this step, we propose a new affinity CNN to generate the graph. Meanwhile, the selected confident seed labels will be converted to corresponding node labels.
(3) Generating final pixel-level pseudo labels. A^2GNN is trained using the converted graph as input, and it makes the prediction for all nodes in the graph. After converting node pseudo labels to pixel labels, we generate the final pixel-level pseudo labels.

After that, a FCN model such as Deeplab [14], [15] for BSSS or MaskR-CNN [16] for BSIS is trained using above pixel-level pseudo labels as supervision.

In the following section, we will first introduce how to provide useful supervision, and then we will give an explanation about how to build a graph from the image (section 4.3). Finally, we will introduce A^2GNN, including its affinity attention layer (section 4.4) and its loss function (section 4.5).

4.2 Confident Seed Label Selection

An intuitive solution is to use the pixel-level seed label M_F obtained from Eq. (4) as the seed labels. However, M_F is noisy and directly using it will be harmful to train a CNN/GNN. As a result, in this paper, we only select those highly confident pixel-level seed labels in M_F as the final seed labels. Specifically, we use a dynamic threshold to select top 40% confident pixel labels M_I' following [40] from the pixel label M_I in Eq. (1). Then the selected seed labels are defined as:

$$M_g(i) = \begin{cases} M_F(i), & \text{if } M_F(i) = M_B(i) \text{ or } M_F(i) = M_I'(i) \\ 255, & \text{else} \end{cases}, \tag{5}$$

where 255 means that the label is unknown. M_B and M_F are obtained from Eq. (3) and Eq. (4), respectively. Fig. 3 (top-right) illustrates the confident label selection.

Although noisy labels can be removed considerably, the confident label selection has two main limitations: 1) it also removes some correct labels, making the rest labels scarce and mainly focus on discriminative object parts (*e.g.*, human head) rather than uniformly distributed in the object; 2) there still exist non-accurate labels.

To tackle the label scarcity problem in the BSSS task, we propose to mine extra supervision information from the available bounding box. Assuming all bounding boxes are tight, for a random row or column pixels inside a bounding box, there is at least one pixel belonging to the object.

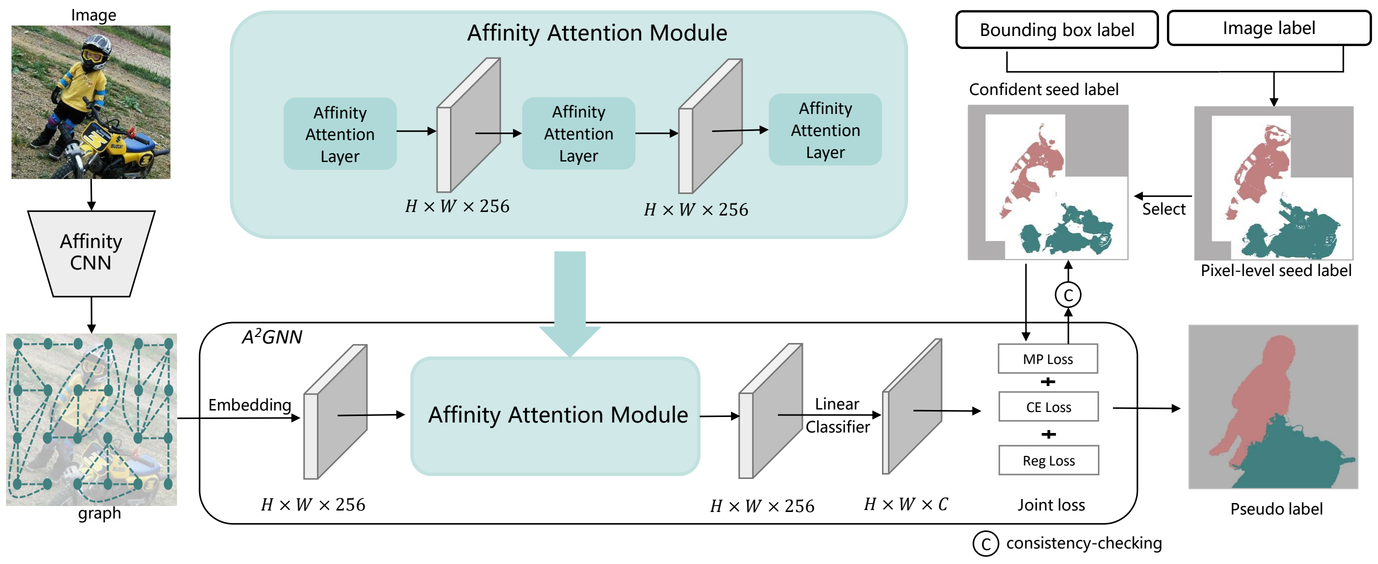


Fig. 3. The framework of our proposed A^2*GNN*. Firstly, we generate pixel-level seed labels using the bounding box and the image-level label. Then our affinity CNN is used to convert images to graphs. Meanwhile, we select confident labels from pixel-level seed labels as the node labels (The node labels in the white region are unknown). Finally, A^2*GNN* uses the graph data as input and the node labels as supervision to produce pseudo labels.

Identifying these nodes can provide extra foreground labels. And using the online updated labels, we introduce a new consistency-checking mechanism to further remove some noisy labels from M_g. We will describe the detailed process in section 4.5 since they rely on the output of our A^2GNN.

4.3 Graph Construction

4.3.1 Affinity CNN

We propose a new affinity CNN to produce an accurate graph from an image using the available affinity labels as supervision. This is because affinity CNN has the following merits. First, instead of regrading one superpixel as a node, it views a pixel as one node which introduces less noise. Second, the affinity CNN uses node affinity labels as training supervision, which ensures to generate suitable node features for this specific task, while previous *GraphNet* [21] uses classification supervision for training. Third, compared to the short distance unweighted graph (edges are only represented as 0 and 1) built in *GraphNet* [21], an affinity CNN can build a weighted graph with soft edges covering a long distance, which gives more accurate node relationship.

Different from prior works [9], [12], [41], [42] that use all noisy labels in Eq. (4) as supervision, our affinity CNN only uses the confident seed labels as defined in Eq. (5) as supervision to predict the relationship of different pixels.

In order to train our affinity CNN, we firstly generate class-agnostic labels from the confident pixel-level seed labels M_g from Eq. (5):

$$L_A(i,j) = \begin{cases} 1, & (i,j) \in R_{pair} \text{ and } M_g'(i) = M_g'(j) \\ 0, & (i,j) \in R_{pair} \text{ and } M_g'(i) \neq M_g'(j)\,, \\ 255, & else \end{cases} \quad (6)$$

where both i and j are pixel indices and 255 means that this pixel pair is not considered. M_g' is the down-sampled result of M_g in order to keep the same height and width with the feature map. R_{pair} is the pixel pair set to train the affinity CNN, and it satisfies the following formula:

$$\begin{aligned} R_{pair} = \{(i,j) | M_g'(i) \neq 255 \text{ and } M_g'(j) \neq 255 \\ \text{and } ||Pos(i) - Pos(j)||_2 \leqslant r\}\,, \end{aligned} \quad (7)$$

where $||\cdot||_2$ is an Euclidean distance operator, $Pos(\cdot)$ represents the coordinate of the pixel. r is the radius, which is used to restrict the selection of a pixel pair.

Given an image I, suppose the feature map from the affinity CNN is F_{A}, following [9], L1 distance is applied to compute the relationship of the two pixels i and j in F_{A}:

$$D(i,j) = exp(-\frac{\|F_A(i) - F_A(j)\|}{d_A}), \quad (8)$$

where d_A is the channel dimension of feature map F_A.

The training loss of affinity CNN is defined as:

$$\mathcal{L}_{\text{Aff}} = \mathcal{L}_{\text{Ac}} + \lambda \mathcal{L}_{\text{Ar}}. \quad (9)$$

In Eq. (9), $\mathcal{L}_{\text{Ac}}$ is a cross-entropy loss which focuses on using the annotated affinity labels as supervision:

$$\begin{aligned} \mathcal{L}_{\text{Ac}} = & -\frac{1}{|A^+|} \sum_{(i,j)\in A^+} L_A(i,j) log(D(i,j)) \\ & -\frac{1}{|A^-|} \sum_{(i,j)\in A^-} (1 - L_A(i,j)) log(1 - D(i,j)), \end{aligned} \quad (10)$$

where A^+ is the node pair set with $L_A(i,j) = 1$, A^- is the node pair set with $L_A(i,j) = 0$. Operator $|\cdot|$ defines the number of elements.

Note that only using the confident labels as supervision is insufficient to train a CNN when only considering $\mathcal{L}_{\text{Ac}}$ as loss function. In order to expand the labeled region to unlabeled region, we propose an affinity regularized loss $\mathcal{L}_{\text{Ar}}$ to encourage propagating from labeled pixels to its connected unlabeled pixels. In other words, instead of only

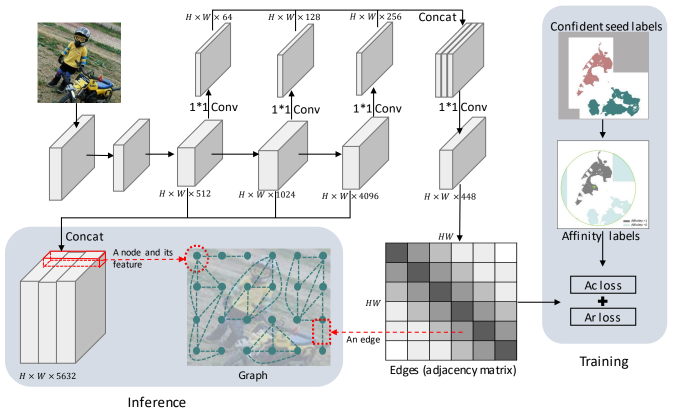


Fig. 4. Converting an image to a graph using our affinity CNN. During inference, the given image will be converted to a graph, in which a node is a pixel in the concatenated feature maps from the last three blocks and its feature is the corresponding pixel feature. The weight of graph edges is defined as the predicted affinity and they are represented as an adjacency matrix, in which each row corresponds to all edges between one node and all nodes.

considering pixel pairs in R_{pair}, we consider all pixel pairs which satisfy the following formula:

$$R_{Ar} = \{(i,j) | \, \|Pos(i) - Pos(j)\|_2 \leqslant r\} \, . \tag{11}$$

Then the affinity regularized loss is defined as:

$$\mathcal{L}_{Ar} = \sum_{i=1}^{\text{HW}} \sum_{(i,j) \in R_{Ar}} G(i,j) \frac{\|(F_A(i) - F_A(j))\|}{d_A}, \tag{12}$$

where $G(\cdot,\cdot)$ is a Gaussian bandwidth filter [1], which utilizes the color and spatial information:

$$G(i,j) = exp(-\frac{\|Pos(i)-Pos(j)\|_2}{2\sigma_{xy}^2} - \frac{\|Cor(i)-Cor(j)\|_2}{2\sigma_{rgb}^2}) \cdot [i \neq j] \, , \tag{13}$$

where $Pos(i)$ and $Pos(j)$ are the spatial positions of i and j, respectively. $Cor(\cdot)$ is the color information and $[\cdot]$ is Iverson bracket.

4.3.2 *Convert Image to Graph*

Usually, a graph is represented as $G = (V, E)$ where V is the set of nodes, and E is the set of edges. Let $v_i \in V$ denote a node and $E_{i,j}$ represents the edge between v_i and v_j. $X \in \mathbb{R}^{N_g * D_g}$ is a matrix representing all node features, where N_g is the number of nodes and D_g is the dimension of the feature. In X, the ith feature, represented as x_i, corresponds to the feature of node v_i. The set of all labeled nodes is defined as V^l, and the set of remaining nodes is represented as V^u, and $V = V^l \cup V^u$.

During training, our affinity CNN uses the class-agnostic affinity labels as supervision and learns to predict the relationship of pixels. During inference, given an image, our affinity CNN will output V, X and E simultaneously for a graph as shown in Fig. 4. Specifically, the node v_i and its feature x_i corresponds to ith pixel and its all-channel features in the concatenated feature map from the backbone. For two nodes v_i and v_j, their edge E_{ij} is defined as:

$$E_{ij} = \begin{cases} D(i,j), & \text{if } D(i,j) > \sigma \\ 0, & \text{else} \end{cases} , \tag{14}$$

where i and j are pixels in the feature map, and $D(i,j)$ is obtained from Eq. (8). Here we use a threshold σ (set as $1e$-3 in our experiment) to make some low affinity edges be 0. Finally, we generate the normalized features:

$$x_{i,j} = x_{i,j} / \sum_{j=1}^{D_g} (x_{i,j}), \tag{15}$$

where $x_{i,j}$ represents the jth value of feature x_i and D_g is the feature dimension.

4.4 Affinity Attention Layer

Effective GNN architectures have been studied in existing works [22], [32], where most of them are designed based on the assumption that the graph node and edge information is 100% accurate. However, in the BSSS task, it is not the case. We propose a new GNN layer with attention mechanism to mitigate this issue. As shown in Fig. 3, in the proposed A^2GNN, an affinity attention module is applied after the embedding layer. The affinity attention module includes three new GNN layers named affinity attention layers. Finally, an output layer is followed to predict class labels for all nodes.

Specifically, we use a feature embedding layer followed by a ReLU activation function in the first layer to map the initial node features to the same dimension of the assigned feature:

$$H^1 = ReLU(XW^0), \tag{16}$$

where X is the feature matrix defined in section 4.3.2 and W^0 is the parameter set of the embedding layer. Then we design several affinity attention layers to leverage the edge weights:

$$H^{l+1} = P^l H^l, \tag{17}$$

where $P^l \in \mathbb{R}^{N_G \times N_G}$, N_G is the number of nodes. For node v_i, the affinity attention $P^l(i,j)$ from node v_j is defined as:

$$\begin{aligned} P^l(i,j) &= \text{softmax}(w^l \cos(H^l(i), H^l(j)) + \beta[\cos(H^l(i), H^l(j)) > 0]E_{ij}) \\ &= \frac{\exp\{w^l \cos(H^l(i), H^l(j)) + \beta[\cos(H^l(i), H^l(j)) > 0]E_{ij}\}}{\sum\limits_{v_j \in S(i)} \exp\{w^l \cos(H^l(i), H^l(j)) + \beta[\cos(H^l(i), H^l(j)) > 0]E_{ij}\}}, \end{aligned} \tag{18}$$

where $l \in \{1, 2, ..., L\}$ is the layer index (L is set as 3 in our model) of A^2GNN and w^l is the learning parameter. $S(i)$ is the set of all the nodes connected with v_i (including itself). $[\cdot]$ equals 1 when $\cos(\cdot,\cdot) > 0$ and otherwise equals 0. $H^l(i)$ and $H^l(j)$ correspond to the features of v_i and v_j at layer l, respectively. $\cos(\cdot,\cdot)$ is used to compute the cosine similarity, which is a self-attention module. E_{ij} is the predicted edge in Eq. (14). β is a weighting factor. The final output is:

$$O = \text{softmax}(H^{L+1} W^{L+1}), \tag{19}$$

where W^{L+1} is the parameter set of the output layer.

Fig. 5 shows the flowchart of our affinity attention layer. Compared to GCN layer [22] and AGNN layer [43], our affinity attention layer makes full advantage of node similarity and edge weighting information.

4.5 Training of A^2GNN

As described in section 4.2, we only select confident labels as supervision, which is insufficient for the network optimization. In order to address this problem, we impose multiple supervision on our A^2GNN. Specifically, we design a new joint loss function, including a cross-entropy loss, a regularized shallow loss [3] and a multi-point (MP) loss:

$$\mathcal{L}_G = \mathcal{L}_{ce} + \mathcal{L}_{mp} + \lambda_1 \mathcal{L}_{reg}, \tag{20}$$

where $\mathcal{L}_{ce}$ is the cross-entropy loss to use the labeled nodes M_g generated in section 4.2. $\mathcal{L}_{reg}$ is the regularized loss using the shallow feature, *i.e.*, color and spatial position. $\mathcal{L}_{mp}$ is the newly proposed MP loss to use the bounding box supervision.

4.5.1 Cross Entropy Loss

L_{ce} is the cross entropy loss, which is used to optimize our A^2GNN based on the labeled nodes M_g:

$$\mathcal{L}_{ce} = -\frac{1}{|V^l|} \sum_{\substack{c_i \in C \\ v_j \in V^l}} [c_i = M_g(v_j)] log(O^{c_i}(v_j)), \tag{21}$$

where V^l is the set of all labeled nodes. $M_g(v_j)$ means the label of node v_j. $|\cdot|$ is used to compute the number of elements. $O^{c_i}(v_j)$ is the predicted probability of being class c_i for node v_j. V^l is the set of labeled nodes.

4.5.2 Regularized Loss

$\mathcal{L}_{reg}$ is the regularized loss which explores the shallow features of images. Here we use it to pose constraints based on the image-domain information (*e.g.*, color and spatial position).

$$\mathcal{L}_{reg} = \sum_{\substack{c_i \in C \\ v_a \in V}} \sum_{\substack{c_j \in C \\ v_b \in V}} G(v_a, v_b) O^{c_i}(v_a) O^{c_j}(v_b). \tag{22}$$

In Eq. (22), v_a and v_b represent two graph nodes. V is the set including all nodes, and $G(v_a, v_b)$ is defined in Eq. (13).

4.5.3 Multi-Point Loss

Inspired by [7], we design a new loss term named multi-point (MP) loss to acquire extra supervision from bounding boxes. This is because the labeled nodes generated in section 4.2 are scarce and not perfectly reliable, which could be complemented by the bounding box information. The MP loss is based on the following consideration. Assuming all bounding boxes are tight, for a random row or column pixels inside one bounding box, there is at least one pixel belonging to the object, if we can find out all these nodes, then we can label them with the object class and close up their distance in the embedding space. Thus, MP loss makes the object easy to be distinguished.

Specifically, for each row/column in the bounding box, the node with the highest probability to be classified to the bounding box class label is regarded as the selected node. Following the same definition in section 3, suppose the bounding box set in one image is B, then for arbitrary bounding box B_j in B, firstly we need to select the highest probability pixel for each row/column:

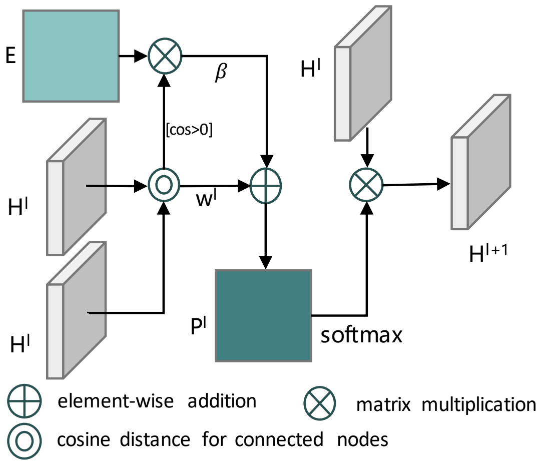


Fig. 5. Our proposed affinity attention layer. E is the adjacent matrix which provides soft edges information. H^l is the input feature of the layer and H^{l+1} is the output feature. P^l is the computed affinity attention matrix. w^l is the learning parameter and β is the weighting factor. With the attention mechanism and the soft edges, it can ensure accurate feature propagation.

$$i_{\text{max}}^{l_m} = \text{index}(\max_{i \in B_j^{l_m}} (O^{B_j}(i))), \tag{23}$$

where l_m means the mth row/column, $\max_{i \in B_j^{l_m}} (O^{B_j}(i))$ means that for each row/column, we select the node which has the highest probability to be classified as the same label with B_j, index $(\cdot)$ returns the index of selected node. $i_{\text{max}}^{l_m}$ is the index of the selected node in the mth row/column.

Then the set that contains all selected nodes for the bounding box B_j are defined as K_j:

$$K_j = \left\{ i_{\text{max}}^{l_1}, i_{\text{max}}^{l_2}, i_{\text{max}}^{l_3}, ..., i_{\text{max}}^{l_{(w+h)}} \right\}, \tag{24}$$

where w and h represents the width and height of B_j, respectively. Then all selected nodes for all bounding boxes are defined as K:

$$K = \{K_1, K_2, ..., K_M\}, \tag{25}$$

where M is the number of bounding boxes. Finally, the MP loss is defined as:

$$\begin{aligned} \mathcal{L}_{mp} = &-\frac{1}{N_p} \sum_{K_j \in K} \sum_{k_i \in K_j} log(O^{B_j}(k_i)) + \\ &\frac{1}{N_f} \sum_{K_j \in K} \sum_{\substack{k_m \in K_j \\ k_n \in K_j}} [k_m \neq k_n](d(H(k_m), H(k_n))). \end{aligned} \tag{26}$$

In Eq. (26), $d(\cdot,\cdot)$ is used to compute feature distance, where we set $d(\cdot,\cdot) = 1 - \cos(\cdot,\cdot)$. $H(k_m)$ and $H(k_n)$ correspond to the features from the last affinity attention layer H^{L+1} for node k_m and k_n, respectively. Both N_p and N_f are the number of sum items. MP loss tries to pull the selected nodes closer in the embedding space, while all other nodes connecting with them will benefit from this loss. This is because GNN layer can be regarded as a layer to aggregate features from the connected nodes, it will encourage the other connected nodes to share a similar feature with them.

In other words, MP loss will make the nodes belonging to the same object easy to be distinguished since they are assigned to a similar feature in the embedding space. In our model, we only enforce MP loss on K_j (Eq. (24)) rather than all labeled nodes. This is because other nodes from M_g still have noisy labels, and at the same time, the labeled foreground nodes in M_g focus more on discriminative parts of the object.

4.5.4 Consistency-Checking

As mentioned in section 4.2, although we select some confident seed labels as supervision, noisy labels are still inevitable. Considering that we provide some extra online labels in our MP loss, which selects the highest probability pixel in each row/column in the box as additional labels, we assume that most additional labels in MP loss are correct, then for each box, we first generate a prototype using the feature of all additional labels inside the box:

$$H_P^{K_j} = \frac{1}{N_{K_j}} \sum_{k_i \in K_j} H(k_i), \tag{27}$$

where $H_P^{K_j}$ represents the prototype of the jth bounding box and N_{K_j} is the number of the selected pixel.

Then for each bounding box, we compute the distance between all selected confident seed labels in Eq. (5) and the prototype, and finally the seed labels which are far away from the prototype are considered as noisy label and removed in each iteration:

$$M_g^{K_j}(i) = \begin{cases} M_g^{K_j}(i), & \text{if } d(H^{K_j}(i), H_P^{K_j}) > 0 \\ & \text{and } M_g^{K_j}(i) = L_{B_j} \\ 255, & \text{else} \end{cases}, \tag{28}$$

where $M_g^{K_j}$ is the selected confident label map of the jth bounding box (section 4.2), H^{K_j} is the corresponding feature map for the jth bounding box from the last affinity attention layer H^{L+1}, and $d(\cdot,\cdot)$ is the operator to compute the cosine distance.

5 Implement Details

To generate pixel-level seed label from image-level label, we use the same classification network as SEAM [38], which is a ResNet-38 [50]. All the parameters are kept the same as in [38].

Our affinity CNN adopts the same backbone with the above classification network. At the same time, dilated convolution is used in the last three residual blocks and their dilated rates are set as 2, 4 and 4, respectively. As in Fig. 4, the output channels of these three residual blocks are 512, 1024 and 4096. A node feature is a concatenated feature of these three outputs, so the feature dimension for one node is 5632. Since we need to use feature to compute distance, three 1×1 convolution kernels are used to reduce the feature dimensions of these three residual blocks and the output channels are set as 64, 128 and 256, respectively. Finally, a 1×1 convolution kernel with 448 channels is used to get the final feature map F_A. Following [9], we set $r = 5$ for both training and inference. λ in Eq. (9) is set to 3 and $\sigma_{xy} = 6$, $\sigma_{rgb} = 0.1$.

Our A^2GNN has five layers as mentioned in section 4.4, the output channel number for the first layer and three affinity attention layers are 256. λ_1 in Eq. (20) are set as 0.01. In $\mathcal{L}_{reg}$, we adopt the same parameters with Eq. (13). We use Adam as optimizer [51] with the learning rate being 0.03 and weight decay being 5×10^{-4}. During training, the epoch number is 100 and the dropout rate is 0.5. The training process will be divided into two stages: In the first stage (the first 50 epochs), L_{reg} and consistency-checking are not used while in the second stage, all losses and consistency-checking are used. We use dropout after the first layer. We use bilinear interpolation to achieve the original resolution during training and inference. CRF [20] is used as the post-processing method during inference. The unary potential of CRF uses the final output probability O in Eq. (19) while pair-wise potential corresponds to the color and spatial position of different nodes. All CRF parameters are the same as [9], [40]. Note that for the BSIS task, we need to convert the above pseudo labels to instance masks. Given a bounding box, we directly assign pixels which locate inside a bounding box and share the same class with it to one instance.

For the BSSS task, we take the Deeplab-Resnet101 [14], PSPNet [45] and Tree-FCN [49] as our fully supervised semantic segmentation models for fair comparison. For the BSIS task, MaskR-CNN [16] is taken as the final instance segmentation model and we use Resnet-101 as the backbone. Following the same post-processing with [7], we use CRF [20] to refine our final prediction.

All experiments are run on 4 Nvidia-TiTan X GPUs. For Pascal VOC 2012 dataset, generating the pixel-level seed label takes about 12 hours, training affinity CNN spends about 12 hours and generating the pseudo labels using A^2GNN takes about 16 hours.

6 Experiment

6.1 Datasets

We evaluate our method on PASCAL VOC 2012 [52] and COCO [53] dataset. For PASCAL VOC 2012, the augmented data SBD [54] is also used, and the whole dataset includes 10,582 images for training and 1,449 images for validating and 1,456 images for testing. For COCO dataset, we train on the default train split (80K images) and then test on the test-dev set.

For Pascal VOC 2012 dataset, mean intersection over union (mIoU) is applied as the evaluation criterion for weakly supervised semantic segmentation, and the mean average precision (mAP) [55] is adopted for weakly supervised instance segmentation. Following the same evaluation protocol as prior works, we reported mAP with three thresholds (0.5, 0.7, 0.75), denoting as mAP$^r_{0.5}$, mAP$^r_{0.7}$ and mAP$^r_{0.75}$, respectively. For COCO dataset, following [56], mAP, mAP$^r_{0.5}$, mAP$^r_{0.75}$, mAP$_s$, mAP$_m$ and mAP$_l$ are reported.

6.2 Comparison with State-of-the-Art

Weakly supervised semantic segmentation: In Table 1, we compare the performance between our method and other state-of-the-art approaches for BSSS. For using deeplab as

TABLE 1
Comparison with other approaches on PASCAL VOC 2012 *val* and *test* sets for BSSS. F: fully supervised. S: scribble supervised. B: bounding box supervised. Seg.: fully-supervised segmentation model

Method	Pub.	Seg.	Sup.	val mIoU (%)		test mIoU (%)	
				w/o CRF	w/ CRF	w/o CRF	w/ CRF
Deeplab-V1 [44]	-	-	F	62.3	67.6	-	70.3
Deeplab-Vgg [14]	TPAMI'18	-	F	68.8	71.5	-	72.6
Deeplab-Resnet101 [14]	TPAMI'18	-	F	75.6	76.8	-	79.7
PSPNet [45]	CVPR'17	-	F	79.2	-	82.6	-
ScribbleSup [2]	CVPR'16	Deeplab-Vgg	S	-	63.1	-	-
RAWKS [46]	CVPR'17	Deeplab-V1	S	-	61.4	-	-
Regularized Loss [3]	ECCV'18	Deeplab-Resnet101	S	73.0	75.0	-	-
Box-Sup [4]	CVPR'15	Deeplab-V1	B	-	62.0	-	64.6
WSSL [17]	CVPR'15	Deeplab-V1	B	-	60.6	-	62.2
GraphNet [21]	ACMM'18	Deeplab-Resnet101	B	61.3	65.6	-	-
SDI [5]	CVPR'17	Deeplab-Resnet101	B	-	69.4	-	-
BCM [6]	CVPR'19	Deeplab-Resnet101	B	-	70.2	-	-
Lin *et al.* [47]	ECCV'18	PSPNet	B	-	74.3	-	-
Box2Seg [30]	ECCV'20	UperNet [48]	B	74.9	76.4	-	-
Box2Seg-CEloss [30]	ECCV'20	UperNet [48]	B	72.7	-	-	-
A^2GNN (ours)	TPAMI'21	Deeplab-Resnet101	B	**72.2**	**73.8**	**72.8**	**74.4**
A^2GNN (ours)	TPAMI'21	PSPNet	B	**74.4**	**75.6**	**73.9**	**74.9**
A^2GNN (ours)	TPAMI'21	Tree-FCN [49]	B	**75.1**	**76.5**	**74.5**	**75.2**

TABLE 2
Comparison with other approaches on PASCAL VOC 2012 *val* dataset for BSIS.

Method	Pub.	Sup.	$\text{mAP}^r_{0.5}$	$\text{mAP}^r_{0.7}$	$\text{mAP}^r_{0.75}$
SDS [55]	ECCV'14	F	49.7	25.3	-
MaskR-CNN [16]	ICCV'17	F	67.9	52.5	44.9
PRM [57]	CVPR'18	I	26.8	-	9.0
IRN [12]	CVPR'19	I	46.7	23.5	-
SDI [5]	CVPR'17	B	44.8	-	16.3
BBTP [7]	NeurIPS'19	B	58.9	30.4	21.6
A^2GNN (ours)	TPAMI'21	B	**59.1**	**35.5**	**27.4**

the segmentation model, it can be seen that our approach obtains 96.1% of our upper-bound with pixel-level supervision (Deeplab-Resnet101 [14] with CRF). Compared to the other approaches, our approach gives a new state-of-the-art performance. Specifically, our approach with deeplab-resnet101 [49] outperforms Box-Sup [4], WSSL [17] by big margins, approximately 11.8% and 13.2%, respectively. Besides, compared to *GraphNet* [21], the only graph learning solution, our method with Deeplab-Resnet101 performs much better than it, with an improvement of 10.9% for mIoU (without CRF). We can also observe that our performance is even better than SDI [5], which uses MCG [18] and BSDS [19] as extra pixel-level supervision. When using PSPNet [45] as the segmentation model, our approach obtains 74.4% mIoU without CRF as post-processing, which is even higher than the results in [47] with CRF. Finally, our method with Tree-FCN [49] outperforms the state-of-the-art Box2Seg [30] in this task. Note that Box2Seg focused on designing a segmentation network using noisy label from bounding box, thus our performance could be further improved using their network as the final segmentation network.

Weakly supervised instance segmentation: In Table 2, we compare our approach to other state-of-the-art approaches on BSIS. It can be seen that our approach achieves a new state-of-the-art performance among all evaluation criteria. Specifically, our approach performs much better than SDI [5], increasing 14.3% and 11.1% on $\text{mAP}^r_{0.5}$ and $\text{mAP}^r_{0.75}$, respectively. It can also be found that compared to BBTP [7], which is the state-of-the-art approach on this task, our approach significantly outperforms it by large margins, around 5.1% on $\text{mAP}^r_{0.7}$ and 5.8% on $\text{mAP}^r_{0.75}$. The performance is increased more on $\text{mAP}^r_{0.75}$ than $\text{mAP}^r_{0.7}$ and $\text{mAP}^r_{0.5}$, which also indicates that our approach can produce masks that preserve the object structure details. One interesting observation is that our approach even achieves better performance than the fully supervised method SDS [55].

In Fig. 6, we compare some qualitative results between our approach and other state-of-the-art approaches for which the source code is publicly available. Specifically, we compare our results with SDI [5][1] for the BSSS task and BBTP [7][2] for the BSIS task. It can be seen that compared to other approaches, our approach produces better segmentation masks covering object details.

In Table 3, we make a comparison between our approach and others on COCO test-dev dataset. It can be seen that our approach performs much better than LIID [61], with an increase of 16.8% on mAP^r_{50}. Furthermore, our approach even performs competitive with fully-supervised approach MNC [60], which also indicates the effectiveness of our approach.

6.3 Ablation Studies

Since the pseudo labels for BSIS are generated from the BSSS task, in this section, we will conduct ablation studies only on the BSSS task. We simply evaluate the pseudo label mIoU on the *training* set, without touching the *val* and *test* set.

In Fig 7, we make a comparison between our A^2GNN and others for BSSS. It can be seen that our A^2GNN performs much better than other GNNs, with an improvement of 1.9% mIoU over AGNN [32] when only using the cross-entropy loss, and the full A^2GNN outperforms AGNN [32] by a large margin (6.9%).

In Table 4(a), we explore the influence of different modules in our approaches to generate pseudo labels. *Baseline*

1. we use a re-implement code from: github.com/johnnylu305
2. github.com/chengchunhsu/WSIS_BBTP

TABLE 3
Comparison with other approaches on COCO test-dev dataset for weakly supervised instance segmentation. E: extra dataset [58] with instance-level annotation. S^4Net: salient instance segmentation model [59].

Method	Pub.	Sup.	mAP	$mAP^r_{0.5}$	$mAP^r_{0.75}$	AP_s	AP_m	AP_l
MNC [60]	CVPR'16	F	24.6	44.3	24.8	4.7	25.9	43.6
Mask-RCNN [16]	ICCV'17	F	37.1	60.0	39.6	35.3	35.3	35.3
Fan *et.al.* [56]	ECCV'18	I+E+S^4Net	13.7	25.5	13.5	0.7	15.7	26.1
LIID [61]	TPAMI'20	I+E+S^4Net	16.0	27.1	16.5	3.5	15.9	27.7
A^2GNN (ours)	TPAMI'21	B	**20.9**	**43.9**	**17.8**	**8.3**	**20.1**	**31.8**

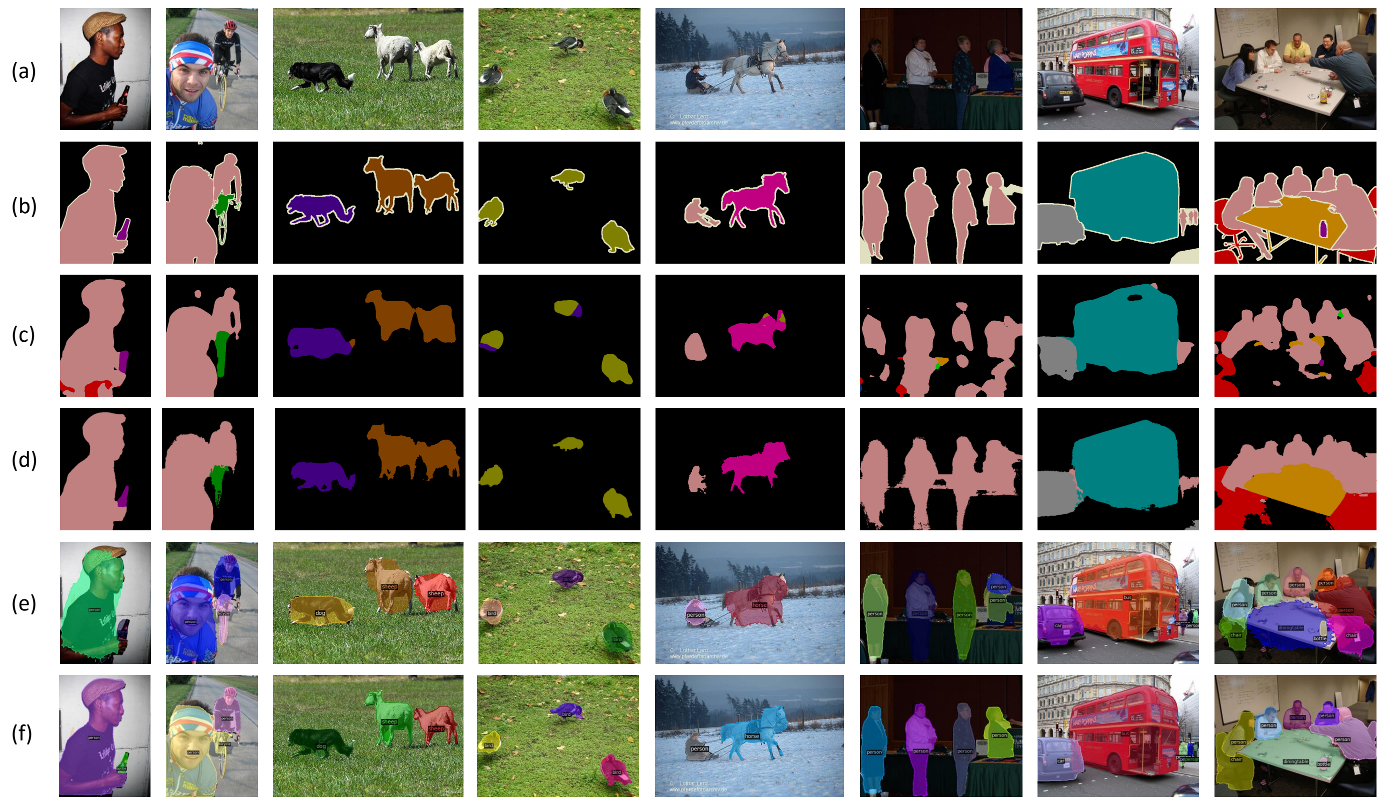


Fig. 6. Qualitative results of our *A^2GNN* and other state-of-the-art approaches on PASCAL VOC 2012 *val* dataset. (a) Original image. (b) Ground truth of semantic segmentation. (c) SDI [5] for BSSS. (d) Our results for BSSS. (e) BBTP [7] for BSIS. (f) Our results for BSIS.

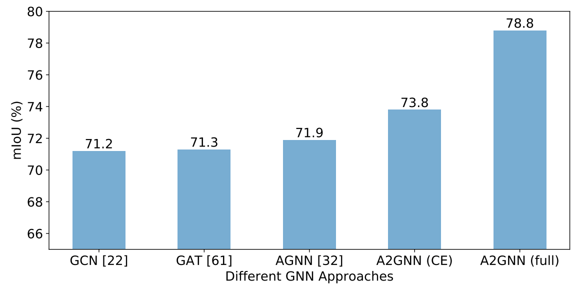


Fig. 7. Comparison between our *A^2GNN* and other GNNs (GCN [22], GAT [62], AGNN [32]) on Pascal VOC 2012 *training* set. "CE" means only cross-entropy loss is used.

means that we use SEAM [38] to generate the foreground seed labels and then use bounding box supervision to generate the background. *RW* means that we follow SEAM [38] to use random walk for pseudo label generation. It can be seen that the proposed approach outperforms the baseline by a large margin. And each module significantly improves the performance.

In Table 4(b), we study the effectiveness of our joint loss function. It can be seen that compared to *A^2GNN* which only adopts cross-entropy loss, our MP loss can improve its performance by 2.1%, validating the effectiveness of our MP loss. With consistency-checking, the performance is improved to 77.2%, indicating the effectiveness of our proposed consistency-checking mechanism. When jointly optimized by these three losses with our consistency-checking mechanism, the performance is further improved to 78.8%.

In Table 4(c), we study different ways to build our graph. *Superpixel (S.P.)* means that we adopt [21] to produce graph nodes and their features. *Distance (Dis.)* means that we build the graph edge using L_1 distance of feature map [21]. It can be seen that the performance is improved when directly using pixel in the feature map as the node, suggesting that it is more accurate than using superpixel. When we use our affinity CNN to build the graph, the performance is significantly improved by 4.1%, which shows that our approach

TABLE 4
Ablation studies on Pascal VOC 2012 *training* dataset for BSSS.

(a) Evaluation for different modules in our approach. *RW*:random walk [38]. *H*: affinity attention layer. *C.C.*: consistency-checking.

Baseline	affinity CNN		*RW*	A^2GNN				mIoU
	$\mathcal{L}_{Ac}$	$\mathcal{L}_{Ar}$		*H*	$\mathcal{L}_{reg}$	$\mathcal{L}_{mp}$	*C.C.*	
✓								62.3
✓	✓		✓					70.3
✓	✓	✓	✓					71.3
✓	✓	✓		✓				73.8
✓	✓	✓		✓	✓			74.9
✓	✓	✓		✓	✓	✓		78.1
✓	✓	✓		✓	✓	✓	✓	**78.8**

(b) Evaluation for the loss functions of our A^2GNN. *C.C.*: consistency-checking.

$\mathcal{L}_{ce}$	$\mathcal{L}_{reg}$	$\mathcal{L}_{mp}$	C.C.	mIoU (%)
✓				73.8
✓	✓			74.9
✓		✓		75.9
✓		✓	✓	77.2
✓	✓	✓		78.1
✓	✓	✓	✓	**78.8**

(c) Evaluation for different methods to build the graph. S.P.: superpixel. Feat.: feature map. Dis.: distance. Aff: affinity CNN.

S.P.	Feat.	Dis.	Aff.	mIoU(%)
✓		✓		73.3
	✓	✓		74.7
	✓		✓	**78.8**

(d) Evaluation of the affinity attention layer in A^2GNN.

A^2GNN layer		mIoU(%)
cos(·)	edge	
✓		73.1
	✓	77.3
✓	✓	**78.8**

(e) Performance comparison for using different seed labels on affinity CNN and the loss functions.

$\mathcal{L}_{Ac}$	$\mathcal{L}_{Ar}$	M_F	M_g	mIoU
✓		✓		74.5
✓			✓	71.7
✓	✓	✓		76.1
✓	✓		✓	**78.8**

can build a more accurate graph than other approaches.

In Table 4(d), we study the effectiveness of our affinity attention layer. It can be found that if we use either the attention module or the affinity module separately, the mIoU score is lower than that of the full A^2GNN, which indicates the effectiveness of our designed GNN layer.

In Table 4(e), we show the joint influence of the loss functions and labels for our proposed affinity CNN. It can be seen that when only using L_{Ac}, M_F labels perform better than M_g. This is because that M_g only provides limited pixels and these pixels are usually located at the discriminative part of an object (such as the human head). Such limited labels are not sufficient when only using L_{Ac}. When we use both L_{Ac} and L_{Ar}, M_g performs much better than M_F, indicating that L_{Ar} can accurately propagate the labeled regions to unlabeled regions.

In addition, we also analyze the influence of supervision for our A^2GNN. Specifically, we make a comparison of the results when using M_F (in Eq. (4)) and M_g (in Eq. (5)) as supervision for our A^2GNN, respectively. Compared to M_F, M_g has fewer annotated nodes but each annotation is more reliable. The mIoU score on Pascal VOC 2012 *training* set is 73.2% and 78.8% for M_F and M_g, respectively. This result validates the effectiveness of the leverage of the high-confident labels.

7 Application to Other Weakly Supervised Semantic Segmentation Tasks

In order to use our approach on other weakly supervised semantic segmentation tasks, *e.g.*, scribble, point and image-level, we need to ignore our proposed MP loss (section 4.5.3) and the consistency-checking (section 4.5.4) as they rely on bounding box supervision. Besides, we need to convert different weak supervised signals to pixel-level seed labels. All other steps and parameters are the same as that in the BSSS task. In the following section, we will introduce how to convert the different weakly supervised signal to pixel-level seed labels, and then we will report experimental results on these tasks.

TABLE 5
Comparison with other state-of-the-arts on PASCAL VOC 2012 *val* and *test* datasets. Sup.: Segmentation model. F: fully supervised. S: scribble. P: point. I: image-level label. E: extra salient dataset. "highlight" means the best performance for a specific task.

Method	Pub.	Seg.	Sup.	*val*	*test*
(1) Deeplab-V1 [44]	-	-	F	67.6	70.3
(2) Deeplab-Vgg [14]	TPAMI'18	-	F	71.5	72.6
(3) Deeplab-Resnet [14]	TPAMI'18	-	F	76.8	79.7
(4) WiderResnet38 [50]	PR'19	-	F	80.8	82.5
(5) Tree-FCN [49]	NeurIPS'19	-	F	80.9	-
RAWKS [46]	CVPR'17	(1)	S	61.4	-
ScribbleSup [2]	CVPR'16	(2)	S	63.1	-
GraphNet [21]	ACMM'18	(3)	S	73.0	-
Regularized loss [3]	ECCV'18	(3)	S	75.0	-
A^2GNN (ours)	TPAMI'21	(3)	S	74.3	74.0
A^2GNN (ours)	TPAMI'21	(5)	S	**76.2**	**76.1**
What's the point [8]	ECCV'16	(2)	P	43.4	43.6
Regularized loss [3]	ECCV'18	(3)	P	57.0	-
A^2GNN(ours)	-	(3)	P	**66.8**	**67.7**
AE-PSL [63]	CVPR'17	(1)	I+E	55.0	55.7
DSRG [26]	CVPR'18	(3)	I+E	61.4	63.2
FickleNet [64]	CVPR'19	(3)	I+E	64.9	65.3
Zhang *et.al* [65]	ECCV'20	(3)	I+E	66.6	66.7
ICD [39]	CVPR'20	(3)	I+E	67.8	68.0
EME [66]	ECCV'20	(3)	I+E	67.2	66.7
MCIS [67]	ECCV'20	(3)	I+E	66.2	66.9
ILLD [61]	TPAMI'20	(3)	I+E	66.5	67.5
ILLD [61]	TPAMI'20	(3)†	I+E	**69.4**	**70.4**
A^2GNN(ours)	TPAMI'21	(3)	I+E	68.3	68.7
A^2GNN(ours)	TPAMI'21	(3)†	I+E	69.0	69.6
PSA [9]	CVPR'18	(4)	I	61.7	63.7
SEAM [38]	CVPR'20	(4)	I	64.5	65.7
ICD [39]	CVPR'20	(3)	I	64.1	64.3
BES [68]	ECCV'20	(3)	I	65.7	66.6
SubCat [69]	CVPR'20	(3)	I	66.1	65.9
CONTA [70]	NeurIPS'20	(4)	I	66.1	66.7
A^2GNN(ours)	TPAMI'21	(3)	I	**66.8**	**67.4**

† means using Res2Net [71] as the backbone.

7.1 Pixel-level Seed Label Generation

As mentioned in section 3, the common practice to initialize weakly supervised task is to generate pixel-level seed labels from the given weak supervision. For different weakly supervision, we use different approaches to convert them

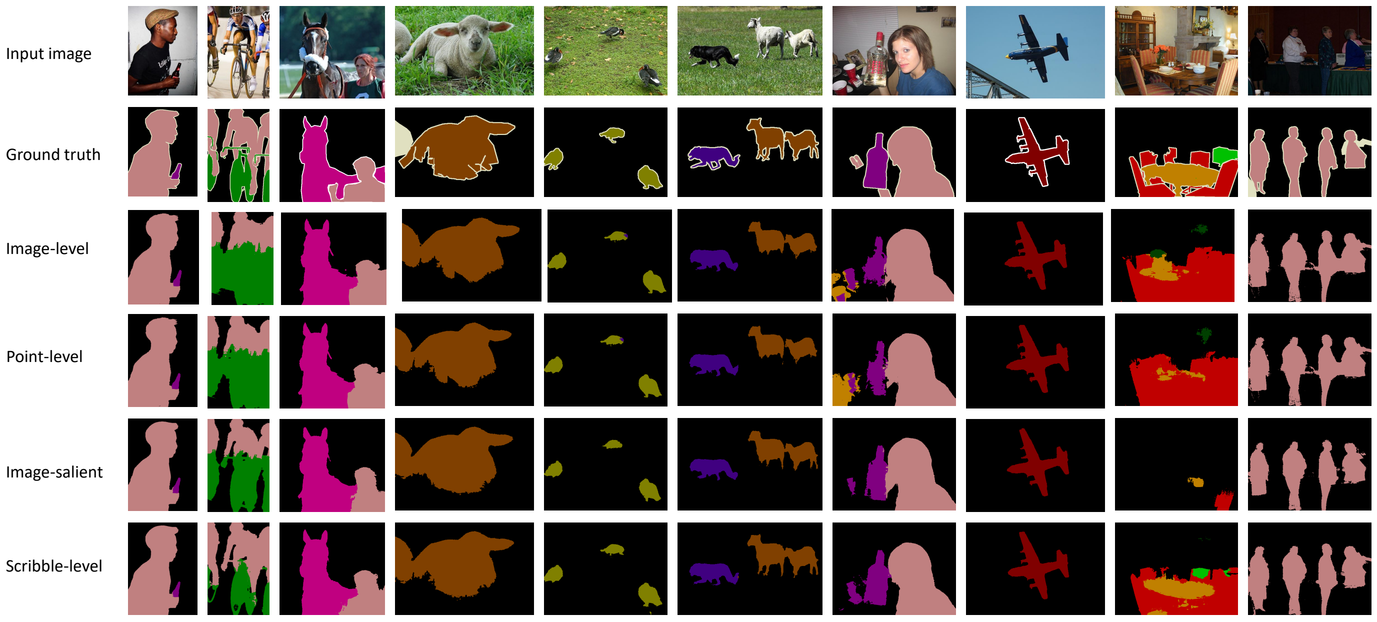


Fig. 8. Qualitative results of our A^2GNN on PASCAL VOC 2012 *val* dataset. We show the results from different levels of supervision signals (3rd – 6th rows). Stronger supervision signals (*e.g.*, scribble) produce more accurate results than weaker signals (*e.g.*, point, image-level label).

to pixel-level seed labels.

Image-level supervision: we directly use M_I' defined in Eq. (5) to train our affinity CNN and use it as M_g to train our A^2GNN. The final pseudo labels are generated using the ratio (1:3) to fuse our results and the results of random walk.

Scribble supervision: For the scribble supervised semantic segmentation task, for each class in an image (including background), it provides one or more scribbles as labels. Superpixel method [23] is used to get the expanded labels M_S from the initial scribbles. To get seed label to train our affinity CNN, we merge M_S with M_I' using the following rule: if the pixel label in M_S is known (not 255), the corresponding label in M_g will be the same label as M_S. Otherwise, the pixel label will be treated as the same label as M_I'. To generate the node labels for A^2GNN, we directly use $M_g = M_S$ since it provides accurate labels for around 10% pixels in an image.

Point supervision: For point supervised semantic segmentation, for each object in an image, it provides one point as supervision and there is no annotation for background. To train our affinity CNN, we used M_I' directly. To generate node supervision for A^2GNN, we use a superpixel method [23] to get the expanded label M_P from initial point labels. Then M_g is generated using the same setting with the scribble task.

For our affinity CNN and our A^2GNN, we use the same setting with our bounding box task.

7.2 Experimental Evaluations

In Table 5, we compare the performance between our method and other state-of-the-art weakly supervised semantic segmentation approaches.

For point supervision, our method achieves state-of-the-art performance with 66.8% and 67.7% mIoU on the *val* and *test* set of PASCAL VOC, respectively. Compared to other two approaches [8] and [3], our method increases 23.4% and 9.8% in mIoU on PASCAL VOC 2012 *val* dataset, respectively.

TABLE 6
Performance comparison in mIoU (%) for evaluating the pseudo labels on the PASCAL VOC training data set.

Method	Pub.	Sup.	mIoU (%)
PSA [9]	CVPR'18	I	58.4
ICD [39]	CVPR'20	I	62.2
SubCat [69]	CVPR'20	I	63.4
SEAM [38]	CVPR'20	I	63.6
A^2GNN (ours)	TPAMI'21	I	**65.3**
A^2GNN (ours)	TPAMI'21	I+E	**66.5**
Box2Seg [30]	ECCV'20	B	73.6*
A^2GNN (ours)	TPAMI'21	B	**78.8**

* Reproduce by ourself.

For the image-level supervision task, our A^2GNN achieves mIoU of 66.8% and 67.4% on *val* and *test* set, respectively. It should be noticed that PSA [9], SEAM [38] and CONTA [70] apply Wider ResNet-38 [50] as segmentation model, which has a higher upper-bound than Deeplab-Resnet101 [14]. Using Deeplab-Resnet101 [14] as the segmentation, Subcat [69] is the state-of-the-art approach on this task, but it require multi-round training processes. Moreover, our method achieves 66.8% mIoU using Deeplab-Resnet101 [14], being 87.0% of our upper-bound (76.8% mIoU score with Deeplab-Resnet [14]) on *val* set.

Besides, for the image-level supervision, some approaches [26], [39], [61], [64] used salient model with extra pixel-level salient dataset [72] or instance pixel-level salient dataset [58] to generate more accurate pseudo labels. Follow these approaches, we also use saliency models. Specifically, we use the saliency approach [73] following ICD [39] to produce the initial seed labels, and then use our approach to produce the final pseudo labels. It can be seen from Table 5 that our approach outperforms other approaches (using ResNet101 as the backbone).Following ILLD [61], we also evaluate our approach using Res2Net [71] as the segmentation backbone, and our performance is further improved to 69.0% and 69.6%. For this setting, we have not designed any specific denoising scheme for the seed labels. Nevertheless, our performance is comparable with other state-of-the-art methods, *e.g.*, [61], which also proves that our method can be well generalized to all weakly supervised tasks.

For the scribble supervision task, our method also achieves a new state-of-the-art performance.

In Table 6, we present a comparison to evaluate the pseudo labels on the PASCAL VOC training set. It can be seen that our approach outperforms other approaches. Compared to the state-of-the-art approach SEAM [38], our approach obtains 1.7% mIoU improvement. We also compare the quality of the pseudo labels between our approach and Box2Seg [30]. It can be seen that our method outperforms Box2Seg [30] by a large margin, with 5.2% mIoU improvement.

In Fig. 8, we also present more qualitative results for the above three tasks. It can be seen that stronger supervision leads to better performance and preserves more segmentation details.

8 Conclusion

We have proposed a new system, A^2GNN, for the bounding box supervised semantic segmentation task. With our proposed affinity attention layer, features can be accurately aggregated even when noise exists in the input graph. Besides, to mitigate the label scarcity issue, we further proposed a MP loss and a consistency-checking mechanism to provide more reliable guidance for model optimization. Extensive experiments show the effectiveness of our proposed approach. In addition, the proposed approach can also be applied to bounding box supervised instance segmentation and other weakly supervised semantic segmentation tasks. As future work, we will investigate how to generate more reliable seed labels and more accurate graph, so that the noise level in the input graph can be alleviated and therefore our A^2GNN can produce more accurate pseudo labels.

References

[1] M. Tang, A. Djelouah, F. Perazzi, Y. Boykov, and C. Schroers, "Normalized cut loss for weakly-supervised cnn segmentation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 1818–1827.

[2] D. Lin, J. Dai, J. Jia, K. He, and J. Sun, "Scribblesup: Scribble-supervised convolutional networks for semantic segmentation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 3159–3167.

[3] M. Tang, F. Perazzi, A. Djelouah, I. Ben Ayed, C. Schroers, and Y. Boykov, "On regularized losses for weakly-supervised cnn segmentation," in *Proceedings of the European Conference on Computer Vision*, 2018, pp. 507–522.

[4] J. Dai, K. He, and J. Sun, "Boxsup: Exploiting bounding boxes to supervise convolutional networks for semantic segmentation," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 1635–1643.

[5] A. Khoreva, R. Benenson, J. Hosang, M. Hein, and B. Schiele, "Simple does it: Weakly supervised instance and semantic segmentation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 876–885.

[6] C. Song, Y. Huang, W. Ouyang, and L. Wang, "Box-driven class-wise region masking and filling rate guided loss for weakly supervised semantic segmentation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 3136–3145.

[7] C.-C. Hsu, K.-J. Hsu, C.-C. Tsai, Y.-Y. Lin, and Y.-Y. Chuang, "Weakly supervised instance segmentation using the bounding box tightness prior," in *Advances in Neural Information Processing Systems*, 2019, pp. 6582–6593.

[8] A. Bearman, O. Russakovsky, V. Ferrari, and L. Fei-Fei, "What's the point: Semantic segmentation with point supervision," in *Proceedings of the European Conference on Computer Vision*. Springer, 2016, pp. 549–565.

[9] J. Ahn and S. Kwak, "Learning pixel-level semantic affinity with image-level supervision for weakly supervised semantic segmentation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 4981–4990.

[10] Y. Wei, X. Liang, Y. Chen, X. Shen, M.-M. Cheng, J. Feng, Y. Zhao, and S. Yan, "Stc: A simple to complex framework for weakly-supervised semantic segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 11, pp. 2314–2320, 2016.

[11] Q. Hou, P. Jiang, Y. Wei, and M.-M. Cheng, "Self-erasing network for integral object attention," in *Proceedings of the Advances in Neural Information Processing Systems*, 2018, pp. 549–559.

[12] J. Ahn, S. Cho, and S. Kwak, "Weakly supervised learning of instance segmentation with inter-pixel relations," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 2209–2218.

[13] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 3431–3440.

[14] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 40, no. 4, pp. 834–848, 2017.

[15] L.-C. Chen, G. Papandreou, F. Schroff, and H. Adam, "Rethinking atrous convolution for semantic image segmentation," *arXiv preprint arXiv:1706.05587*, 2017.

[16] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask r-cnn," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 2961–2969.

[17] G. Papandreou, L.-C. Chen, K. Murphy, and A. Yuille, "Weakly-and semi-supervised learning of a dcnn for semantic image segmentation (2015)," *arXiv preprint arXiv:1502.02734*.

[18] P. Arbeláez, J. Pont-Tuset, J. T. Barron, F. Marques, and J. Malik, "Multiscale combinatorial grouping," in *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 2014, pp. 328–335.

[19] D. Martin, C. Fowlkes, D. Tal, and J. Malik, "A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics," in *Proceedings of the Eighth IEEE International Conference on Computer Vision.*, vol. 2. IEEE, 2001, pp. 416–423.

[20] P. Krähenbühl and V. Koltun, "Parameter learning and convergent inference for dense random fields," in *Proceedings of the International Conference on Machine Learning*, 2013, pp. 513–521.

[21] M. Pu, Y. Huang, Q. Guan, and Q. Zou, "Graphnet: Learning image pseudo annotations for weakly-supervised semantic segmentation," in *Proceedings of the 26th ACM International Conference on Multimedia*, 2018, pp. 483–491.

[22] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," *arXiv preprint arXiv:1609.02907*, 2016.

[23] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Süsstrunk, "Slic superpixels compared to state-of-the-art superpixel methods," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 11, pp. 2274–2282, 2012.

[24] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba, "Learning deep features for discriminative localization," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 2921–2929.

[25] L. Lovász, "Random walks on graphs: A survey," *Combinatorics, Paul erdos is eighty*, vol. 2, no. 1, pp. 1–46, 1993.

[26] Z. Huang, X. Wang, J. Wang, W. Liu, and J. Wang, "Weakly-supervised semantic segmentation network with deep seeded

region growing," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 7014–7023.

[27] R. Adams and L. Bischof, "Seeded region growing," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 16, no. 6, pp. 641–647, 1994.

[28] J. Pont-Tuset, P. Arbelaez, J. T. Barron, F. Marques, and J. Malik, "Multiscale combinatorial grouping for image segmentation and object proposal generation," *IEEE transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 1, pp. 128–140, 2016.

[29] C. Rother, V. Kolmogorov, and A. Blake, "Grabcut: Interactive foreground extraction using iterated graph cuts," in *ACM Transactions on Graphics (TOG)*, vol. 23, no. 3, 2004, pp. 309–314.

[30] V. Kulharia, S. Chandra, A. Agrawal, P. Torr, and A. Tyagi, "Box2seg: Attention weighted loss and discriminative feature learning for weakly supervised segmentation," in *Proceedings of the European Conference on Computer Vision*. Springer, 2020, pp. 290–308.

[31] S. Abu-El-Haija, A. Kapoor, B. Perozzi, and J. Lee, "N-gcn: Multi-scale graph convolution for semi-supervised node classification," *arXiv preprint arXiv:1802.08888*, 2018.

[32] Z. Guo, Y. Zhang, and W. Lu, "Attention guided graph convolutional networks for relation extraction," *arXiv preprint arXiv:1906.07510*, 2019.

[33] S. Harada, H. Akita, M. Tsubaki, Y. Baba, I. Takigawa, Y. Yamanishi, and H. Kashima, "Dual convolutional neural network for graph of graphs link prediction," *arXiv preprint arXiv:1810.02080*, 2018.

[34] Y. Yan, Q. Zhang, B. Ni, W. Zhang, M. Xu, and X. Yang, "Learning context graph for person search," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 2158–2167.

[35] Z.-M. Chen, X.-S. Wei, P. Wang, and Y. Guo, "Multi-label image recognition with graph convolutional networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 5177–5186.

[36] Y. Cai, L. Ge, J. Liu, J. Cai, T.-J. Cham, J. Yuan, and N. M. Thalmann, "Exploiting spatial-temporal relationships for 3d pose estimation via graph convolutional networks," in *Proceedings of the IEEE International Conference on Computer Vision*, 2019, pp. 2272–2281.

[37] W. Wang, X. Lu, J. Shen, D. J. Crandall, and L. Shao, "Zero-shot video object segmentation via attentive graph neural networks," in *Proceedings of the IEEE International Conference on Computer Vision*, 2019, pp. 9236–9245.

[38] Y. Wang, J. Zhang, M. Kan, S. Shan, and X. Chen, "Self-supervised equivariant attention mechanism for weakly supervised semantic segmentation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2020, pp. 12 275–12 284.

[39] J. Fan, Z. Zhang, C. Song, and T. Tan, "Learning integral objects with intra-class discriminator for weakly-supervised semantic segmentation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2020, pp. 4283–4292.

[40] B. Zhang, J. Xiao, Y. Wei, M. Sun, and K. Huang, "Reliability does matter: An end-to-end weakly supervised semantic segmentation approach," *arXiv preprint arXiv:1911.08039*, 2019.

[41] C. Song and J. Xiao, "Cian: Cross-image affinity net for weakly supervised semantic segmentation," 2020.

[42] B. Wu, S. Zhao, W. Chu, Z. Yang, and D. Cai, "Improving semantic segmentation via dilated affinity," *arXiv preprint arXiv:1907.07011*, 2019.

[43] K. K. Thekumparampil, C. Wang, S. Oh, and L.-J. Li, "Attention-based graph neural network for semi-supervised learning," *arXiv preprint arXiv:1803.03735*, 2018.

[44] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "Semantic image segmentation with deep convolutional nets and fully connected crfs," *arXiv preprint arXiv:1412.7062*, 2014.

[45] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia, "Pyramid scene parsing network," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017.

[46] P. Vernaza and M. Chandraker, "Learning random-walk label propagation for weakly-supervised semantic segmentation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 7158–7166.

[47] Q. Li, A. Arnab, and P. H. Torr, "Weakly-and semi-supervised panoptic segmentation," in *Proceedings of the European Conference on Computer Vision*, 2018, pp. 102–118.

[48] T. Xiao, Y. Liu, B. Zhou, Y. Jiang, and J. Sun, "Unified perceptual parsing for scene understanding," in *Proceedings of the European Conference on Computer Vision*, 2018, pp. 418–434.

[49] L. Song, Y. Li, Z. Li, G. Yu, H. Sun, J. Sun, and N. Zheng, "Learnable tree filter for structure-preserving feature transform," in *Advances in Neural Information Processing Systems*, 2019, pp. 1711–1721.

[50] Z. Wu, C. Shen, and A. Van Den Hengel, "Wider or deeper: Revisiting the resnet model for visual recognition," *Pattern Recognition*, vol. 90, pp. 119–133, 2019.

[51] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[52] M. Everingham and J. Winn, "The pascal visual object classes challenge 2012 (voc2012) development kit," *Pattern Analysis, Statistical Modelling and Computational Learning, Tech. Rep*, 2011.

[53] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft coco: Common objects in context," in *Proceedings of the European Conference on Computer Vision*. Springer, 2014, pp. 740–755.

[54] B. Hariharan, P. Arbeláez, L. Bourdev, S. Maji, and J. Malik, "Semantic contours from inverse detectors," in *Proceedings of the IEEE International Conference on Computer Vision*, 2011, pp. 991–998.

[55] B. Hariharan, P. Arbeláez, R. Girshick, and J. Malik, "Simultaneous detection and segmentation," in *Proceedings of the European Conference on Computer Vision*. Springer, 2014, pp. 297–312.

[56] R. Fan, Q. Hou, M.-M. Cheng, G. Yu, R. R. Martin, and S.-M. Hu, "Associating inter-image salient instances for weakly supervised semantic segmentation," in *Proceedings of the European Conference on Computer Vision*, 2018, pp. 367–383.

[57] Y. Zhou, Y. Zhu, Q. Ye, Q. Qiu, and J. Jiao, "Weakly supervised instance segmentation using class peak response," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 3791–3800.

[58] G. Li, Y. Xie, L. Lin, and Y. Yu, "Instance-level salient object segmentation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 2386–2395.

[59] R. Fan, M.-M. Cheng, Q. Hou, T.-J. Mu, J. Wang, and S.-M. Hu, "S4net: Single stage salient-instance segmentation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 6103–6112.

[60] J. Dai, K. He, and J. Sun, "Instance-aware semantic segmentation via multi-task network cascades," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 3150–3158.

[61] Y. Liu, Y.-H. Wu, P. Wen, Y. Shi, Y. Qiu, and M.-M. Cheng, "Leveraging instance-, image- and dataset-level information for weakly supervised instance segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020.

[62] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio, "Graph attention networks," *arXiv preprint arXiv:1710.10903*, 2017.

[63] Y. Wei, J. Feng, X. Liang, M.-M. Cheng, Y. Zhao, and S. Yan, "Object region mining with adversarial erasing: A simple classification to semantic segmentation approach," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 1568–1576.

[64] J. Lee, E. Kim, S. Lee, J. Lee, and S. Yoon, "Ficklenet: Weakly and semi-supervised semantic image segmentation using stochastic inference," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 5267–5276.

[65] T. Zhang, G. Lin, W. Liu, J. Cai, and A. Kot, "Splitting vs. merging: Mining object regions with discrepancy and intersection loss for weakly supervised semantic segmentation," in *Proceedings of the European Conference on Computer Vision*, 2020.

[66] J. Fan, Z. Zhang, and T. Tan, "Employing multi-estimations for weakly-supervised semantic segmentation," in *Proceedings of the European Conference on Computer Vision*. Springer, 2020.

[67] G. Sun, W. Wang, J. Dai, and L. Van Gool, "Mining cross-image semantics for weakly supervised semantic segmentation," in *Proceedings of the European Conference on Computer Vision*. Springer, 2020, pp. 347–365.

[68] L. Chen, W. Wu, C. Fu, X. Han, and Y. Zhang, "Weakly supervised semantic segmentation with boundary exploration," in *Proceedings of the European Conference on Computer Vision*. Springer, 2020, pp. 347–362.

[69] Y.-T. Chang, Q. Wang, W.-C. Hung, R. Piramuthu, Y.-H. Tsai, and M.-H. Yang, "Weakly-supervised semantic segmentation via

sub-category exploration," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2020, pp. 8991–9000.

[70] D. Zhang, H. Zhang, J. Tang, X. Hua, and Q. Sun, "Causal intervention for weakly-supervised semantic segmentation," *arXiv preprint arXiv:2009.12547*, 2020.

[71] S.-H. Gao, M.-M. Cheng, K. Zhao, X.-Y. Zhang, M.-H. Yang, and P. Torr, "Res2net: A new multi-scale backbone architecture," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021.

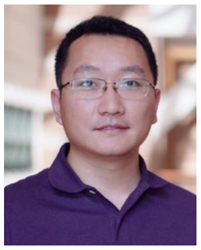
[72] Q. Hou, M.-M. Cheng, X. Hu, A. Borji, Z. Tu, and P. H. Torr, "Deeply supervised salient object detection with short connections," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 3203–3212.

[73] J.-J. Liu, Q. Hou, M.-M. Cheng, J. Feng, and J. Jiang, "A simple pooling-based design for real-time salient object detection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 3917–3926.

Bingfeng Zhang received the B.S. degree in electronic information engineering from China University of Petroleum (East China), Qingdao, PR China, in 2015, the M.E. degree in systems, control and signal processing from University of Southampton, Southampton, U.K., in 2016. He is now a Ph.D student in the University of Liverpool, Liverpool, U.K., and also a Ph.D student in the school of the advanced technology of the Xi'an Jiaotong-Liverpool University, Suzhou, PR China. His current research interest is weakly supervised semantic segmentation and few-shot segmentation.

Jimin Xiao received the B.S. and M.E. degrees in telecommunication engineering from the Nanjing University of Posts and Telecommunications, Nanjing, China, in 2004 and 2007, respectively, and the Ph.D. degree in electrical engineering and electronics from the University of Liverpool, Liverpool, U.K., in 2013. From 2013 to 2014, he was a Senior Researcher with the Department of Signal Processing, Tampere University of Technology, Tampere, Finland, and an External Researcher with the Nokia Research Center, Tampere. Since 2014, he has been a Faculty Member with Xi'an Jiaotong-Liverpool University, Suzhou, China. His research interests include image and video processing, computer vision, and deep learning.

Jianbo Jiao is a Postdoctoral Researcher in the Department of Engineering Science at the University of Oxford. He obtained his Ph.D. degree in Computer Science from City University of Hong Kong in 2018. He was the recipient of the Hong Kong PhD Fellowship. He was a visiting scholar with the Beckman Institute at the University of Illinois at Urbana-Champaign from 2017 to 2018. His research interests include computer vision and machine learning.



Yunchao Wei is currently an Assistant Professor at the University of Technology Sydney. He received his PhD degree from Beijing Jiaotong University in 2016. Before joining UTS, he was a Postdoc Researcher in Prof. Thomas Huang's Image Formation and Professing (IFP) group at Beckman Institute, UIUC, from 2017 to 2019. He has published over 60 papers on top-tier journals and conferences (e.g., T-PAMI, CVPR, ICCV, etc.), Google citation 3900+. He received the Excellent Doctoral Dissertation Award of CIE in 2016, ARC Discovery Early Career Researcher Award in 2019, 1st Prize in Science and Technology awarded by China Society of Image and Graphics in 2019. His research interests mainly include Deep learning and its applications in computer vision, e.g., image classification, video/image object detection/segmentation, and learning with imperfect data. He has organized multiple Workshops and Tutorials in CVPR, ICCV, ECCV and ACM MM.

Yao Zhao received the B.S. degree from the Radio Engineering Department, Fuzhou University, Fuzhou, China, in 1989, the M.E. degree from the Radio Engineering Department, Southeast University, Nanjing, China, in 1992, and the Ph.D. degree from the Institute of Information Science, Beijing Jiaotong University (BJTU), Beijing, China, in 1996, where he became an Associate Professor and a Professor in 1998 and 2001, respectively. From 2001 to 2002, he was a Senior Research Fellow with the Information and Communication Theory Group, Faculty of Information Technology and Systems, Delft University of Technology, Delft, The Netherlands. In 2015, he visited the Swiss Federal Institute of Technology, Lausanne (EPFL), Switzerland. From 2017 to 2018, he visited University of Southern California. He is currently the Director with the Institute of Information Science, BJTU. His current research interests include image/video coding, digital watermarking and forensics, video analysis and understanding, and artificial intelligence. Dr. Zhao is a Fellow of the IET. He serves on the Editorial Boards of several international journals, including as an Associate Editor for the IEEE TRANSACTIONS ON CYBERNETICS, a Senior Associate Editor for the IEEE SIGNAL PROCESSING LETTERS, and an Area Editor for Signal Processing: Image Communication. He was named a Distinguished Young Scholar by the National Science Foundation of China in 2010 and was elected as a Chang Jiang Scholar of Ministry of Education of China in 2013.